IC3K 2024

16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management

PROCEEDINGS

Volume 1: KDIR

Porto, Portugal

17 - 19 November, 2024

EDITORS

Ana Fred
Frans Coenen
Jorge Bernardino

https://ic3k.scitevents.org/

SPONSORED BY

INSTICC

PAPERS AVAILABLE AT

SCITEPRESS DIGITAL LIBRARY

IC3K 2024

Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management

Volume 1: KDIR

Porto - Portugal

November 17 - 19, 2024

Sponsored by
INSTICC - Institute for Systems and Technologies of Information, Control and Communication

ACM In Cooperation
SIGAI - ACM Special Interest Group on Artificial Intelligence

In Cooperation with
AAAI - Association for the Advancement of Artificial Intelligence
APPIA - Portuguese Association for Artificial Intelligence

Edited by Frans Coenen, Ana Fred and Jorge Bernardino

Printed in Portugal
ISSN: 2184-3228
ISBN: 978-989-758-716-0
DOI: 10.5220/0000193400003838
Depósito Legal: 536889/24

https://kdir.scitevents.org
kdir.secretariat@insticc.org

BRIEF CONTENTS

INVITED SPEAKERS

Carlo Sansone
University of Naples Federico II
Italy

Nirmalie Wiratunga
Robert Gordon University, Aberdeen
United Kingdom

João Gama
University of Porto
Portugal

Organizing Committees

Conference Chair

Jorge Bernardino, Polytechnic of Coimbra - ISEC, Portugal

Program Co-chairs

Frans Coenen, University of Liverpool, United Kingdom
Ana Fred, Instituto de Telecomunicações and Instituto Superior Técnico (University of Lisbon), Portugal

Secretariat

Ana Rita Paciência, INSTICC, Portugal

Graphics Production and Webdesigner

Inês Teles, INSTICC, Portugal

Webmaster

João Francisco, INSTICC, Portugal
Carolina Ribeiro, INSTICC, Portugal

PROGRAM COMMITTEE

Amir Ahmad, United Arab Emirates University, United Arab Emirates

Mayer Aladjem, Ben-Gurion University of the Negev, Israel

Eva Armengol, IIIA CSIC, Spain

Mohamed Ben Aouicha, University of SFax, Tunisia

Marko Bohanec, Jožef Stefan Institute, Slovenia

Mohamed-Rafik Bouguelia, Halmstad University, Sweden

Alina Campan, Northern Kentucky University, United States

Erion Çano, Independent Researcher, Czech Republic

Jesús Carrasco-Ochoa, INAOE, Mexico

Luigi Cerulo, University of Sannio, Italy

Sharma Chakravarthy, University of Texas at Arlington, United States

Chih-Ming Chen, National Chengchi University, Taiwan, Republic of China

Chong Chen, Beijing Normal University, China

Zhiyuan Chen, University of Maryland Baltimore County, United States

Patrick Ciarelli, Universidade Federal do Espírito Santo, Brazil

Justin Dauwels, Nanyang Technological University, Singapore

Tai Dinh, The Kyoto College of Graduate Studies for Informatics, Japan

Mihaela Dinsoreanu, Technical University of Cluj-Napoca, Romania

Thanh-Nghi Do, College of Information Technology, Can Tho University, Vietnam

Bilel Elayeb, Liwa College of Technology, United Arab Emirates

Iaakov Exman, School of Computer Science, HIT = Holon Institute of Technology, Israel

Dayne Freitag, SRI International, United States

Susan Gauch, University of Arkansas, United States

Josephine Griffith, National University of Ireland, Galway, Ireland

Gerhard Heyer, Leipzig University, Germany

Dorit Hochbaum, University of California-Berkeley, United States

Victoria Hodge, Department of Computer Science, University of York, United Kingdom

Beatriz de la Iglesia, University of East Anglia, United Kingdom

Arti Jain, Jaypee Institute of Information Technology, India

Yogan Jaya Kumar, Universiti Teknikal Malaysia Melaka, Malaysia

Uzay Kaymak, Eindhoven University of Technology, Netherlands

Ron Kenett, Technion, Israel

Roman Kern, Know-Center GmbH, Austria

Ikuo Keshi, Fukui University of Technology, Japan

Margita Kon-Popovska, Ss Cyril and Methodius University, North Macedonia

Constantine Kotropoulos, Aristotle Univ. of Thessaloniki, Greece

Jean-Charles Lamirel, LORIA, University of Strasbourg, France

Jie Liu, WOU, United States

Jake Luo, University of Wisconsin-Milwaukee, United States

Xiao Luo, Oklahoma State University, United States

Christos Makris, University of Patras, Greece

Saadia Malik, King AbdulAziz University, Saudi Arabia

J. Martínez-Trinidad, Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, Mexico

Dulani Meedeniya, University of Moratuwa, Sri Lanka

Enza Messina, University of Milano Bicocca, Italy

Manuel Montes y Gómez, INAOE, Mexico

Agnieszka Mykowiecka, Institute of Computer Science Polish Academy of Sciences, Poland

Simon O'Keefe, University of York, United Kingdom

Elias Oliveira, Universidade Federal do Espírito Santo, Brazil

Farhad Oroumchian, University of Wollongong in Dubai, United Arab Emirates

Gaurav Pradhan, Mayo Clinic College of Medicine and Science, United States

Alfredo Pulvirenti, University of Catania, Italy

Marcos Quiles, Federal University of Sao Paulo - UNIFESP, Brazil

Chotirat Ratanamahatana, Chulalongkorn University, Thailand

Georg Rehm, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Germany

Leonardo Ribeiro, Federal University of Goiás, Brazil

Marcela Ribeiro, Universidade Federal de São Carlos (UFSCar), Brazil

Farag Saad, FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Germany

Diana Santos, University of Oslo, Norway

Rui Santos, Polytechnic Institute of Leiria, CEAUL - Center of Statistics and Applications, Portugal

Alessia Sarica, Magna Graecia University of Catanzaro, Italy

Abeed Sarker, Emory University, United States

Claudio Sartori, Universita' di Bologna, Italy

Milos Savic, University of Novi Sad, Serbia

Filippo Sciarrone, Universitas Mercatorum, Italy

Asadollah Shahbahrami, University of Guilan, Iran, Islamic Republic of

Dana Shapira, Ariel University, Israel

Marek Sikora, Silesian University of Technology, Poland

Wei Song, North China University of Technology, China

Julian Szymanski, Gdansk University of Technology, Poland

Maguelonne Teisseire, INRAE, France

Marco Temperini, Sapienza University of Rome, Italy

Kar Toh, Yonsei University, Korea, Republic of

Suppawong Tuarob, Mahidol University, Thailand

Domenico Ursino, Università Politecnica delle Marche, Italy

Ishan Verma, Tata Consultancy Services, India

Manuel Vilares Ferro, University of Vigo, Spain

Jie Wang, University of Massachusetts Lowell, United States

Xing Wei, Pinterest Inc., United States

Albert Weichselbraun, University of Applied Sciences of the Grisons, Switzerland

Nicola Zeni, University of Trento, Italy

Yan Zhang, California State University, San Bernardino, United States

AUXILIARY REVIEWERS

Roberto Asín, Departamento de Informatica, Universidad Tecnica Federico Santa Maria, Chile

Abdelhalim Dahou, Independent Researcher, Germany

Zheng Fang, Pinterest, United States

Giacomo Frisoni, Universita di Bologna, Italy

Andrew Mackey, University of Arkansas, United States

Quan Mai, University of Arkansas, United States

Uttamasha Oyshi, University of Arkansas, United States

Juliana Wolf Pereira, UFMS, Brazil

SELECTED PAPERS BOOK

A number of selected papers presented at KDIR 2024 will be published by Springer in a CCIS Series book. This selection will be done by the Conference Chair and Program Co-chairs, among the papers actually presented at the conference, based on a rigorous review by the KDIR 2024 Program Committee members.

FOREWORD

This book contains the proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. This year, IC3K was held in Porto, Portugal, from 17 to 19 November, 2024. It was sponsored by the Institute for Systems and Technologies of Information, Control and Communication (INSTICC). IC3K 2024 was also organized in collaboration with the ACM Special Interest Group on Artificial Intelligence, the Association for the Advancement of Artificial Intelligence and the Portuguese Association for Artificial Intelligence.

The purpose of the IC3K series is to bring together researchers, engineers, and practitioners in the areas of Knowledge Discovery and Information Retrieval (KDIR), Knowledge Engineering and Ontology Development (KEOD) and Knowledge Management and Information Systems (KMIS).

IC3K this year, as in previous years, was composed of 3 subconferences, each specializing in one of the aforementioned knowledge areas, namely KDIR, KEOD, and KMIS.

IC3K 2024 received 175 paper submissions from 47 countries of which 21.14% were accepted and published as full papers. A double-blind paper review was performed for each submission by at least 2 but usually 3 or more members of the International Program Committee, consisting of established researchers and domain experts.

The high quality of the IC3K 2024 program is enhanced by the keynote lectures delivered by distinguished speakers who are renowned experts in their fields: Carlo Sansone (University of Naples Federico II, Italy), Nirmalie Wiratunga (The Abderdeen Robert Gordon University, United Kingdom) and João Gama (University of Porto, Portugal).

The conference is complemented by a Special Session on Ontologies for Digital Twin modelling, chaired by Fatma Chamekh. In addition, some tutorials on relevant topics have been offered to the conference audience.

All papers presented will be available in the SCITEPRESS Digital Library and will be submitted for evaluation for indexing by SCOPUS, Google Scholar, The DBLP Computer Science Bibliography, Semantic Scholar, Engineering Index and Web of Science / Conference Proceedings Citation Index.

In recognition of the best papers, several awards will be presented at the closing session of the conference, based on the combined scores of the paper reviewers, as assessed by the Programme Committee, and the quality of the presentation, as assessed by the session chairs at the conference venue.

Authors of selected papers will be invited to submit extended versions for inclusion in a forthcoming book of IC3K Selected Papers to be published by Springer, as part of the CCIS Series. Some papers will also be selected for publication of extended and revised versions in the special issue of the Springer Nature Computer Science Journal.

The program for this conference required the dedicated efforts of many people. Firstly, we must thank the authors, whose research efforts are herewith recorded. Next, we thank the members of the Program Committee and the auxiliary reviewers for their diligent and professional reviewing. We would also like to express our sincere gratitude to the invited speakers for their invaluable contributions and for taking the time to prepare their presentations. Finally, a word of appreciation for the hard work of the INSTICC team; organizing a conference of this level is a task that can only be achieved by the collaborative effort of a dedicated and highly competent team.

We wish you all an exciting and inspiring conference. We hope to have contributed to the development of our research community, and we look forward to presenting more research at the next edition of IC3K, details of which can be found at https://ic3k.scitevents.org.

Frans Coenen
University of Liverpool, United Kingdom

Ana Fred
Instituto de Telecomunicações and Instituto Superior Técnico (University of Lisbon), Portugal

Jorge Bernardino
Polytechnic of Coimbra - ISEC, Portugal

Contents

INVITED SPEAKERS

Keynote Speakers

PAPERS

Full Papers

SHORT PAPERS

INVITED SPEAKERS

KEYNOTE SPEAKERS

Multimodal Deep Learning in Medical Imaging

Carlo Sansone
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione, University of Naples Federico II, Naples, Italy

Abstract: In this talk, we will consider how Deep Learning (DL) approaches can profitably exploit the presence of multiple data sources in the medical domain. First, the need to be able to use information from multimodal data sources is addressed. Starting from an analysis of different multimodal data fusion techniques, an innovative approach will be proposed that allows the different modalities to influence each other. However, in medical applications it is often very difficult to obtain high quality and balanced labelled datasets due to privacy and sharing policy issues. Therefore, several applications have leveraged DL approaches in data augmentation techniques, proposing models that can create new realistic and synthetic samples. Consequently, a new data source can be identified, namely a synthetic data source. In this context, a data augmentation method based on deep learning, specifically designed for the medical domain, will be presented. It exploits the biological characteristics of images by implementing a physiologically-aware synthetic image generation process.

BRIEF BIOGRAPHY

Carlo Sansone is currently Full Professor of Computer Engineering at the Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione of the University of Naples Federico II. His basic interests cover the areas of image analysis, pattern recognition and machine and deep learning. From an applicative point of view, his main contributions were in the fields of biomedical image analysis, biometrics, intrusion detection in computer networks and image forensics. He coordinated several projects in the areas of artificial intelligence, biomedical images interpretation and network intrusion detection. Prof. Sansone is a member of the IEEE and of the International Association for Pattern Recognition (IAPR). In 2012 he was elected Vice-President of the GIRPR (the Italian Association affiliated to the IAPR) for two terms (four years).

Sansone, C.
Multimodal Deep Learning in Medical Imaging.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, page 5
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Intelligent Reuse of Explanation Experiences: The Role of Case-Based Reasoning in Promoting Best Practice in Explainable AI

Nirmalie Wiratunga
RGU's School of Computing, United Kingdom

Abstract: The EU now requires that machine learning models provide an explanation of their decisions. Different stakeholders may have different backgrounds, competencies, and goals, which may require different types of explanations. Interpreting and explaining machine learning (ML) models can be done in various ways, and there are many options available. However, it's difficult to know which method or combination of methods to use for different AI models and different deployment situations. The iSee project is trying to tackle this question. In this talk we will discuss why Case-Based Reasoning (CBR) is well placed to promote best practices in Explainable AI (XAI). We will also explore how CBR can be used to reason about end-users' XAI experiences and enable the sharing and reusing of such experiences through the iSee platform (https://isee4xai.com/). The talk will present the key components that facilitate reasoning in iSee – an ontology to model experiences, cases to capture experiences, a retrieval engine to identify best practice, and an interactive interface to engage with end-users

BRIEF BIOGRAPHY

Nirmalie Wiratunga is a Professor in Intelligent Systems at RGU's School of Computing, and the Associate Dean for Research in the school, with over two decades of experience in computer science and AI research. She has held positions such as postdoctoral researcher on EPSRC funded projects, and was appointed Readership in 2009, and Professorship in 2016. Nirmalie lead's the Artificial Intelligence & Reasoning Research Group (AIR) in the School of Computing. She has been involved in numerous funded AIR projects, including the development of platforms for reusable explainable AI experiences and initiatives in healthcare.

Wiratunga, N.
Intelligent Reuse of Explanation Experiences: The Role of Case-Based Reasoning in Promoting Best Practice in Explainable AI.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, page 7
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Recent Advances in Learning from Data Streams

João Gama
University of Porto, Portugal

Abstract: Learning from data streams is a hot topic in machine learning and data mining. This talk presents two problems and discusses streaming techniques to solve them. The first problem is the application of data stream techniques to predictive maintenance. We propose a two-layer neuro-symbolic approach to explain black-box models. The explanations are oriented toward equipment failures. For the second problem, we present a streaming algorithm for online hyperparameter tuning. The Self hyper-parameter Tuning (SPT) algorithm is an optimisation algorithm for online hyper-parameter tuning from non-stationary data streams. SPT is a wrapper over any streaming algorithm and can be used for classification, regression, and recommendation.

BRIEF BIOGRAPHY

João Gama is a Full Professor at the Faculty of Economy, University of Porto. He is a researcher and vice-director of LIAAD, a group belonging to INESC TEC. He got the PhD degree from the University of Porto, in 2000. He is a Senior member of IEEE.He has worked on several National and European projects on Incremental and Adaptive learning systems, Ubiquitous Knowledge Discovery, Learning from Massive, and Structured Data, etc. He served as Co-Program chair of ECML 2005, DS 2009, ADMA 2009, IDA 2011, and ECML/PKDD 2015. He served as track chair on Data Streams with ACM SAC from 2007 till 2016. He organized a series of Workshops on Knowledge Discovery from Data Streams with ECML/PKDD, and Knowledge Discovery from Sensor Data with ACM SIGKDD. He is the author of several books on Data Mining (in Portuguese) and authored a monograph on Knowledge Discovery from Data Streams. He authored more than 250 peer-reviewed papers in areas related to machine learning, data mining, and data streams. He is a member of the editorial board of international journals ML, DMKD, TKDE, IDA, NGC, and KAIS. He (co-)supervised more than 12 PhD students and 50 MSc students.

Gama, J.
Recent Advances in Learning from Data Streams.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, page 9
ISBN: 978-989-758-716-0; ISSN: 2184-3228

PAPERS

FULL PAPERS

Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search

Colin Daly[1,2][a] and Lucy Hederman[1,2][b]
[1]*The ADAPT SFI Research Centre, Ireland*
[2]*School of Computer Science and Statistics, Trinity College Dublin, Ireland*
{dalyc24, hederman}@tcd.ie

Keywords: Enterprise Search, Learning to Rank, Query Auto-Complete, Language Modelling.

Abstract: Query Auto-Completion (QAC) is of particular importance to the field of Enterprise Search, where query suggestions can steer searchers to use the appropriate organisational jargon/terminology and avoid submitting queries that produce no results. The order in which QAC candidates are presented to users (for a given prefix) can be influenced by signals, such as how often the prefix appears in the corpus, most popular completions, most frequently queried, anchor text and other of a document, or what queries are currently trending in the organisation. We measure the individual contribution of each of these heuristic signals and supplement them with a feature based on Large Language Modelling (LLM) to detect jargon/terminology. We use Learning To Rank (LTR) to combine the weighted features to create a QAC ranking model for a live Enterprise Search service. In an online A/B test over a 12-week period processing 100,000 queries, our results show that the addition of our jargon/terminology detection LLM feature to the heuristic LTR model results in a Mean Reciprocal Rank score increase of 3.8%.

1 INTRODUCTION

Query Auto-Completion (QAC) presents users with a ranked list of suggested queries in a drop-down box as they start typing their query in a search box. QAC facilitates the search process by making it easier for users to finish entering their queries without typing all the letters.

The user's query prefix can be reformulated inline to use specific 'wording' that ensures relevance or semantic matching. This reformulation is particularly useful for Enterprise Search (ES), where organisations have their own jargon and terminology. QAC can also help avoid users submitting queries that produce no results (a potentially common occurrence, as an ES corpus is small compared to WS) (Kruschwitz and Hull, 2017).

A consequence of the ubiquity of commercial/Internet Web Search (WS) is that users have high expectations when it comes to interacting with search engines (Davis et al., 2011). Cleverley and Burnett refer to this as 'Google Habitus' (Cleverley and Burnett, 2019). Ranked query suggestions are an expected characteristic of every interactive search service (White, 2018).

The simple definition of ES is finding the information needed within an organisation (Bentley, 2011). Many employees or members of an organisation may not be proficient in using their organisation's jargon and terminology. QAC for ES can educate new staff members about the range of selections available to them and assist in narrowing that selection even before the user has finished typing a query. For commercial WS services, the search box typically occupies the centre of an otherwise blank page. Major providers such as Google, Yahoo, Bing, and Baidu offer ten auto-complete suggestions. For ES, the search box is less prominent and has limited real estate to present suggestions. For this reason, many ES services present fewer suggestion candidates. Additionally, more suggestions could cause users (especially on mobile devices) to either begin to ignore suggestions (at which point the additional suggestions become mere noise) or spend an inordinate amount of time reading suggestions (interrupting or even halting the flow of their search session) (Scott, 2022). Where an ES service presents a restricted number of suggestions, their ranking is more important.

Learning to Rank (LTR) is the application of su-

[a] https://orcid.org/0000-0001-7218-7765
[b] https://orcid.org/0000-0001-6073-4063

Daly, C. and Hederman, L.
Learning to Rank for Query Auto-Complete with Language Modelling in Enterprise Search.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 15-26
ISBN: 978-989-758-716-0; ISSN: 2184-3228

pervised machine learning techniques to train a model to list the best ranking order (Li, 2011; Xu et al., 2020). In the context of QAC, this involves combining signals to present the best order of query candidates for a given query prefix. LTR computes the optimum 'weight' (importance) of signals, which are extracted from the ES corpus, and query log files.

The challenge of deciphering enterprise jargon/terminology within a corpus lends itself to the fields of natural language processing (NLP) and large language models (LLM). LLMs, such as OpenAI's GPT (Generative Pre-trained Transformer), are trained on large datasets containing vast amounts of text from diverse sources. Word embeddings can capture semantic relationships between words in text data. While LLMs and embeddings are regularly used in e-commerce (Singh et al., 2023a) and commercial search engines (Li et al., 2017), their application for ES has not been sufficiently explored.

In this paper, we introduce a ranking feature explicitly designed for ES. We call this 'QACES' (Query Auto-Complete for Enterprise Search). This feature is centred on the relative unusualness of words, such as those used in organisational jargon/terminology. QACES is scalable and can be applied to any ES service. Our hypothesis is that adding the QACES feature to a heuristic LTR ranking model will significantly increase the QAC ranking performance, as measured by Mean Reciprocal Rank (MRR). The major contribution of this research is the introduction of our new feature designed to detect, understand and suggest jargon/terminology specific to organisations. For performance context, we undertake an offline ablation study to measure the individual improvement of each ranking feature. Subsequently, we perform an A/B test on a live ES service of a large third-level academic institution to confirm that the addition of the QACES feature to our heuristic model can significantly improve QAC ranking performance.

2 RELATED WORK

2.1 QAC Components

Depending on a researcher's field of study, the terminology used to describe QAC varies widely. A user's incomplete input is often referred to as a 'query prefix'. The generated query suffixes or suggestions are 'query candidates' or 'query completions'. Lesser used terminologies used to describe the same or similar functionality include typeahead, query transformation (Croft, 2010), 'search as you type' (Turnbull and Berryman, 2016), predictive search, auto-suggest, real-time query expansion (RTQE) (White and Marchionini, 2007), query modification suggestions (Kruschwitz and Hull, 2017) and subword completion (Kim, 2019). A closely related and overlapping concept to QAC is 'auto-complete suggestion', which is a more general term that often encompasses a broader range of features aimed at assisting users in formulating queries that do not necessarily contain the same starting string of characters. Google differentiates these two concepts by using the word 'predictions' rather than 'suggestions'. For predictions, the priority is to faithfully 'help people complete a search they were intending to do, not to suggest new types of searches to be performed.'[1]. In practice, since QAC may also include the related tasks of suggestion, correction (e.g. spelling reformulation) and expansion, the terms QAC and query suggestion are usually used interchangeably (Yadav et al., 2021; Li et al., 2017), as is the case in this study.

2.2 Approaches to QAC Ranking

There are two principal approaches to ranking QAC candidates (Cai and De Rijke, 2016). The first, more traditional approach is heuristic and combines domain-specific signals from a corpus and query logs. This approach, where ranking signals are handcrafted based on relevance or popularity, typically uses experimental or trial-and-error methods to apply weightings to features. The heuristic approach produces ranking models that are relatively transparent. The second approach employs NLP or Language Modelling to produce context-aware suggestions (Singh et al., 2023a; Kim, 2019). Learning to Rank can combine or 'fuse' the two approaches with any number of features (Rahangdale and Raut, 2019; Guo et al., 2016) as outlined in §3.

2.3 Historical QAC Data

Relevance judgements in the form of annotated *query-document* pairs are typically required to train a ranking model for documents on the Search Engine Results Page (SERP) (Joachims, 2002; Daly, 2023). QAC researchers rarely rely on the same editorial effort to manually annotate *prefix-candidate pairs*. More often, QAC relies on the collection of large-scale historical data for QAC tasks (Chang and Demg, 2020). Previously recorded behaviour and queries provide useful information for any user's intent and can be leveraged to suggest completions that are more

[1] https://blog.google/products/search/how-google-autocomplete-works-search/, accessed 29th June 2023

relevant while adhering to the user's prefix (Yadav et al., 2021).

Most Popular Completions (MPC) is a ranking feature that proposes completion candidates based on user preferences recorded in historical search data. For this reason, and because of a reluctance of users to provide explicit feedback (Kruschwitz and Hull, 2017), MPC is typically considered the main indicator of historical relevance (Li et al., 2017) and thus considered to be a credible proxy for ground truth in many datasets.

A similar ranking feature extracted from historical logs is Most Frequent Queries (MFQ) (Yadav et al., 2021). These queries represent the topics or keywords searched for most frequently by the user community. MFQ simply ranks the most frequent suggestions matching the input prefix and is particularly useful when MPC data is sparse.

Much research for QAC focuses on improving the relevance of suggested queries using a ranking model trained and evaluated with data generated by commercial search engines, such as the 2006 AOL WS dataset (Pass et al., 2006), which includes over 10 million queries. ES differs from WS insofar as the content may be indexed from multiple databases (e.g. corporate directories) and intranet document repositories. ES may also include searches for explicitly indexed usernames, course codes, tracking numbers, purchasing codes or any datum specific to the organisation (Craswell et al., 2005). In §3.3, we demonstrate that the AOL query history is sweeping, centred around popular culture and often archaic. While the AOL QAC dataset is not ideal, we could not find a more relevant ES benchmark. A test collection or dataset based on Enterprise Search is hard to come by, as organisations are not inclined to open their intranet to public distribution, even for research purposes (Craswell et al., 2005; Cleverley and Burnett, 2019).

Personalisation of an individual user's session or recent history enables 'contextual suggestions', which has proved very effective for completing a user's query prefix in Web Search (Fiorini and Lu, 2018). For example, if a particular user submits a Google search for "Past American Presidents", then if his/her next query prefix starts with an 'N', the suggestion will be 'Richard Nixon'. The use of personalisation for QAC in the domain of Enterprise Search seems to be rare, possibly because members of the organisation may not wish to be 'profiled'.

2.4 Trending Queries

Queries that have been popular in a recent time period merit a ranking feature to capture temporal behavioural trends. In 1999, the operators of the Lycos commercial Web Search engine began publishing a weekly list of the 50 most popular queries submitted. The query term 'Britney Spears' was number two on the weekly list. The popularity of that term endured, and 'Britney Spears' never fell off the list over the next eight years. This meant that the list was quite static, and emergent topics were volumetrically drowned out. This has sometimes been referred to as 'The Britney Spears Problem' (Hayes, 2008). A more dynamic list tells us what topic or query is *up and coming* or generating a *buzz* as measured by a sudden abnormal burst of interest. This concept is known as trending and is based on the relative spike in the volume of clustered topic searches in relation to the absolute volume of searches. Unlike the popularity list, the trend list would exclude the constantly popular 'Britney Spears'.

2.5 Terms Extracted from the Corpus

The preceding sections describe how historical log data can be harnessed to create candidates. Separately, candidates can also be retrieved from the corpus. Enterprise Search engines like Apache Solr (The Apache Software Foundation., 2004) are designed for the explicit retrieval of Term Frequency (TF) within a particular field of a schema, such as title, content body, anchors, footers, etc. A candidate indexed from anchor text within a corpus is likely to be more important than a candidate from the content field.

2.6 Jargon/Terminology

Jargon is enterprise-specific vocabulary that employees/members can understand. It encompasses words, phrases, expressions, and idioms that are not universally familiar or properly understood. The same term can have a different meaning outside of the organisation. A search for 'timetable' in WS will probably return bus or train times. Krushwitz gives the example that the same search in a third-level education institution might be aimed at lecture timetables (in Autumn) or exam timetables (in Spring) (Kruschwitz et al., 2013).

Although excessive use of jargon and terminology in organisations is often perceived as exclusionary, we use the terms here in a positive context for conveying complex ideas, processes, or services among employees/members who share common knowledge of the enterprise. In this context, jargon and terminology facilitate efficient communication.

As we will see in §3, the task of detecting enterprise jargon/terminology terms within a corpus lends itself to the fields of NLP and LLM. Once the terms have been detected and scored, they can be used as another feature in our model.

2.7 Learning to Rank for QAC

Users' search history is generally the most likely indicator of current search intent (Chang and Demg, 2020). The use of MPC and MFQ to extract suggestions from historical log data assumes that current and future query popularity distribution will remain the same as previously observed. A generalised ranking model is therefore required to handle *as-yet-unseen* queries. LTR can be used to combine multiple features with the optimum weighting contribution to a ranking model. When a user types a query prefix, Apache Solr can retrieve and dynamically score suggestions from multiple sources using a 'weightExpression' in real-time.

2.8 Metrics for QAC

The evaluation of QAC performance has two general approaches (Chang and Demg, 2020), each of which has its own metric:

1. The MRR metric focuses on the quality of ranking.
2. The Minimum Keystroke Length (MKS) metric that focuses on savings of a user's keystroke effort (Duan and Hsu, 2011).

Since this study focuses on ranking rather than keystroke effort, we compute MRR, which is widely accepted as the principal metric for evaluating QAC ranking performance (Li et al., 2017; Cai and De Rijke, 2016). The MRR metric is appropriate whenever evaluating a list of possible responses to a sample of queries, ordered by the probability of correctness. The Reciprocal Rank (RR) of a query response is the multiplicative inverse of the rank of the first correct answer: 1 for first place, 1⁄3 for third place, or zero if the submitted query is not present in the ranked list (Singh et al., 2023a). The mean reciprocal rank is the average of the RR results for a sample of queries Q:

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

where $rank_i$ refers to the rank position of the first relevant document for the i^{th} query. MRR only considers the rank of the first relevant candidate (if there are further relevant candidates, they are ignored).

3 METHODS

This section describes how logging is configured to capture users' autocomplete session behaviour. This historical behaviour is converted into a QAC dataset, which is used to train a baseline ranking model using LTR-weighted features. We describe each feature and analyse its contribution. We describe how our QACES concept can supplement the baseline heuristic features. The subsequent ranking models are then evaluated using the MRR metric.

3.1 Log Collection for QAC

To enable detailed capture of users' session behaviour, the log4j module[2] of the Apache Solr Enterprise Search platform (The Apache Software Foundation., 2004) is modified to record the suggestion candidates for a given query prefix, the selected candidate (if any), and finally, the submitted query. This session data enables the calculation of an RR score (§2.8). Table 1 lists the recorded parameters for each query session.

Table 1: QAC session logging parameters to capture suggestion candidates, record the user's selection, and the submitted query.

Parameter	ES QAC Dataset example
user id (anon)	qtp1209411469-21
session id	33418(rid)
time stamp	2024-01-14 16:25:37.697
prefix	"aca"
top suggestion candidates	academic registry, academic calendar, academic year structure, academic registry fees, academic year, academic practice, academic resources
submitted query	"academic registry"

The QAC session id commences as soon as the first letter is typed into the search box and ends when the user selects a suggestion candidate or hits submit with the fully typed query (Li et al., 2017).
This enhanced logging has no discernible impact on the responsiveness or latency of our live ES service.

[2] https://cwiki.apache.org/confluence/display/solr/SolrLogging, accessed 16th Jan 2024

Storage was added to the backend linux servers to host both the enhanced logging file size and file retention for 180 days.

3.2 QAC Dataset Construction

3.2.1 Ground Truth

Yossef et al. claim that MPC carries ‘the wisdom of the crowds’ (Bar-Yossef and Kraus, 2011). It can be regarded as an approximate maximum likelihood estimator (Li et al., 2017; Yadav et al., 2021). In our study, we use MPC as a surrogate for judgements of prefix-candidate pairs. For a given prefix, enumerated judgements {2-5} are allocated to the candidates, representing {irrelevant, moderately relevant, relevant, highly relevant}. The judgement scores are computed as a percentile of the MPC score. Figure 1 shows an extract of our dataset, which has been formatted for use with an LTR framework (§3.5).

3.2.2 Sensical Suggestions

An important pre-processing step applied to any QAC dataset for ES is the removal of suggestions that would produce no search results if selected. An incorrect suggestion may be considered more damaging than no suggestion, as it would undermine users' confidence in the ES service. This step is sometimes referred to as producing ‘plausible completions‘ or filtering out of ‘nonsensical’ suggestions (Yadav et al., 2021). For example, many suggestions extracted from the 2006 AOL WS dataset (Table 2) cannot be used. Even the top AOL queries, such as ‘mapquest’ and ‘myspace’ do not produce search results within our ES corpus.

3.2.3 Pre-Processing

Further pre-processing steps involved converting all queries, prefixes, and completions to lowercase. Full stops, other punctuation, and diacritics were removed. Queries and suggestions with less than three characters, more than eight words, or 50 characters (whichever was the bigger) were removed or truncated. All non-English queries were removed.

3.3 Heuristic Ranking Features

We describe the list of ranking functions below as ‘heuristic’ because they are carefully hand-crafted based on domain knowledge and information retrieval principles.

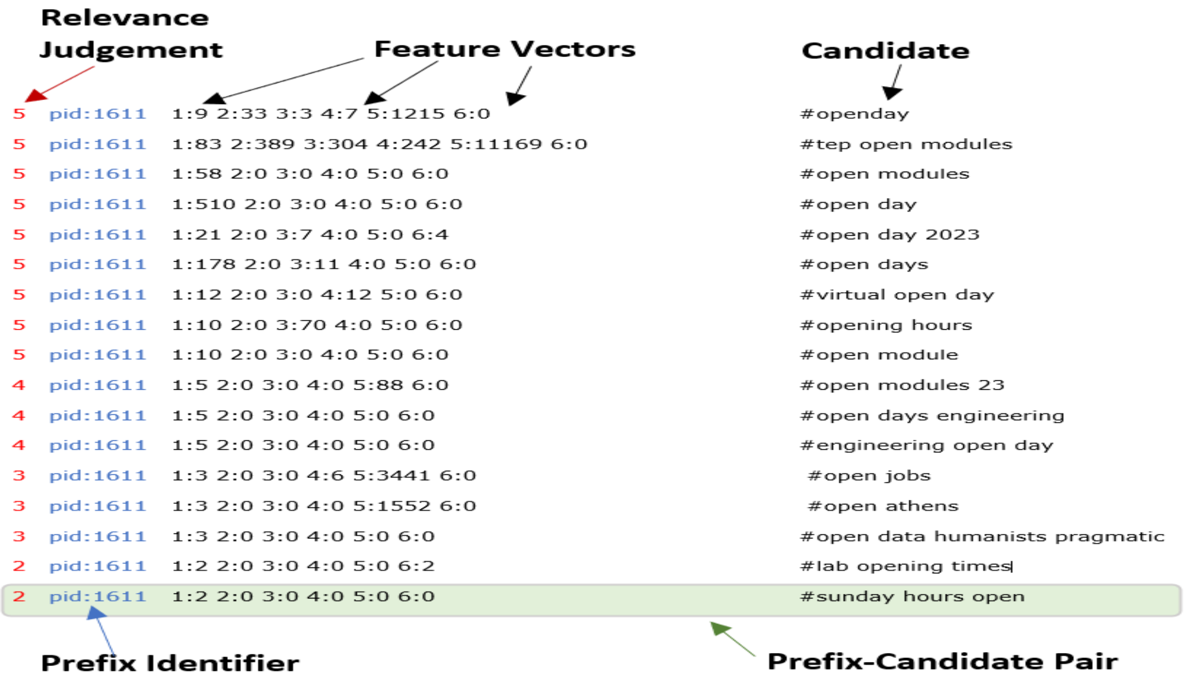
- *MFQ*. The ‘Most Frequent Queries’ feature consists of a table of queries with their frequency of occurrence extracted from the query logs. An example is ”data science 410”, which tells us that the query ‘data science’ has been submitted to the search engine 410 times. The Search Demand Curve (Figure 2) shows the outsize impact that a small number of popular queries has on the overall volume of search activity. According to Kritzinger et al, popular search terms make up 30% of the overall searches performed on commercial Web Search engines. Using zoning norms devised by the SEO community in 2011, the 18.5% of searchers with the highest occurrence is known as the Fat Head. The next 11.5% is termed the Chunky Middle (Kritzinger and Weideman, 2013). The Long Tail (x-axis) in Figure 2 has been limited for presentation purposes but actually represents 70% of the search volume. In our ES query logs, we see that 65 queries account for 18.5% of all search volume[3].
- *aolFeature* This feature extracts pertinent queries from the AOL WS dataset (20 million search queries from about 650,000 users collected between May and July 2006). Since our ES service must only offer ‘sensical’ candidates, our extracted list contains just 1740 candidates, which is 0.009% of the total. The feature consists of a table of queries with their frequency of occurrence extracted from the AOL dataset. An example is ”music downloads 517”, which tells us that the query ‘music downloads’ has been submitted to the AOL search engine 517 times. The AOL queries differ markedly from the types of query we expect for ES, which has a much narrower focus, as shown in Table 2.

Table 2: The top 10 most popular query terms for the 2006 AOL WS compared with our ES query history.

Top 10	AOL WS query terms	ES query terms
1	google	scholarship
2	ebay	fees
3	yahoo	library
4	mapquest	phd
5	yahoo.com	medicine
6	google.com	pshychology
7	myspace.com	erasmus
8	internet	courses esc
9	myspace	vacancies
10	www.google.com	law

- *trendingFeature*. The new or nearly new terms that have been queried in the past 24 hours. To measure an abnormal spike, we must first deter-

[3] https://github.com/colindaly75/QAC_LTR_for_ES



```
5  pid:1611  1:9 2:33 3:3 4:7 5:1215 6:0              #openday
5  pid:1611  1:83 2:389 3:304 4:242 5:11169 6:0       #tep open modules
5  pid:1611  1:58 2:0 3:0 4:0 5:0 6:0                 #open modules
5  pid:1611  1:510 2:0 3:0 4:0 5:0 6:0                #open day
5  pid:1611  1:21 2:0 3:7 4:0 5:0 6:4                 #open day 2023
5  pid:1611  1:178 2:0 3:11 4:0 5:0 6:0               #open days
5  pid:1611  1:12 2:0 3:0 4:12 5:0 6:0                #virtual open day
5  pid:1611  1:10 2:0 3:70 4:0 5:0 6:0                #opening hours
5  pid:1611  1:10 2:0 3:0 4:0 5:0 6:0                 #open module
4  pid:1611  1:5 2:0 3:0 4:0 5:88 6:0                 #open modules 23
4  pid:1611  1:5 2:0 3:0 4:0 5:0 6:0                  #open days engineering
4  pid:1611  1:5 2:0 3:0 4:0 5:0 6:0                  #engineering open day
3  pid:1611  1:3 2:0 3:0 4:6 5:3441 6:0                #open jobs
3  pid:1611  1:3 2:0 3:0 4:0 5:1552 6:0                #open athens
3  pid:1611  1:3 2:0 3:0 4:0 5:0 6:0                  #open data humanists pragmatic
2  pid:1611  1:2 2:0 3:0 4:0 5:0 6:2                  #lab opening times
2  pid:1611  1:2 2:0 3:0 4:0 5:0 6:0                  #sunday hours open
```

Figure 1: An extract of our LTR formatted dataset including a sample of the prefix-candidate pairs for the "open" prefix. Each candidate has an associated judgement for a particular prefix (generated using MPC). The candidates also have an prefix identifier (pid) and a series of feature vectors.

mine what would be a normal baseline score. This type of calculation lends itself to z-scores, which consider the burst of popularity against the backdrop of the historical average (including its standard deviation). This feature consists of a table of queries with their computed z-score (e.g. "graduate studies 22.81").[4] A higher z-score indicates that the query is more 'trending'.

- *anchorTextFeature*. The 'anchorText' is the link label that content providers use to describe a document. This feature was generated using the LinkRank algorithm (similar to Google's PageRank). In the field of Web Search, this feature can be expected to have a high weighting co-efficient as it both tags meaningful descriptive text and also adds context to a document. In ES, LinkRank may be less effective, as many documents are created without publishing intent (e.g. MS Word documents placed on an intranet drive). This feature consists of a table of terms with their frequency of occurrence. An example is "research support system 24", which tells us that the phrase 'research support system' is encoded into anchors 24 times in our corpus. Considerable filtering was required to remove non-descriptive terms and repetitive labels such as 'Next page', 'Previous Page', 'Home', etc.
- *titleFeature*. This TF feature is a list of candidates with their corresponding frequency retrieved from the field of our enterprise corpus. An example is "communication 333", which tells us that the term 'communication' occurs 333 times in the title field of our corpus.
- *contentFeature*. This TF feature is a list of candidates with their corresponding frequency retrieved from the content field of our enterprise corpus.

Although MPC is typically considered the main indicator of historical relevance (Li et al., 2017), it cannot be included as a feature here (since we already use it as a proxy for ground truth (i.e. the target variable) in our LTR dataset). Similarly, while Personalisation has proved very effective for completing a user's query prefix in WS, we exclude it as part of this ES research as our organisation does not permit the profiling of individual user data.

3.4 QACES LLM Feature for ES

The task of detecting enterprise jargon/terminology terms within a corpus lends itself to the fields of NLP and LLM. We call this feature 'QACES' (Query Auto-Complete for Enterprise Search). It is computed using the synonyms of terms in the real world and the closest terms in an enterprise corpus. Where these differ significantly, the term is considered jargon. Once

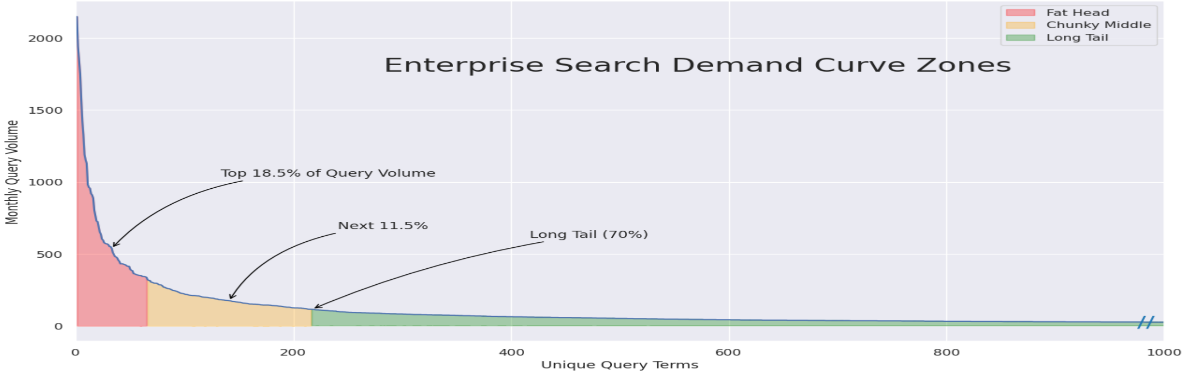


Figure 2: Search Demand Curve for our Enterprise Search query history, showing the so-called 'Fat Head', 'Chunky Middle' and 'Long Tail' zones. For our ES service, the most popular 65 queries represent 18.5% of all search volume.

the jargon terms have been detected, they can be used as a feature in our ranking model.

3.4.1 LLM Synonyms

LLMs are trained on enormous datasets containing vast amounts of text from diverse sources. We use the 'client.chat.completions.create' API response of OpenAI's GPT-4 model (OpenAI 2023, 2023) to produce a list of ten English language 'LLM nearest' terms for each query in the 'fat head' zone of our Search Curve (this is the second column of Table 3). An alternative tool to GPT-4 would have been WordNet (Princeton University, 2010). We opted not to use WordNet, however, as frequency data are not independently available, making it impossible to determine the *nearest* terms.

Use of the GPT-4 API prevents data leaking (Balloccu et al., 2024). We tested the temperature parameter at both 0.5 and 1.0 and observed no obvious changes in the retrieved nearest terms. The prompt was "create a flat, json-formatted, sorted, unnumbered list of the top 10 nearest (semantically) words or phrases for each of the words in the following array". We struggled to achieve repeatable lists of near-synonyms on each run. The detailed wording of the prompt was necessary to achieve repeatable results. The array included all of the query terms in the 'fat head' of our Search Demand Curve.

3.4.2 ES Corpus Vectorisation

Word embeddings represent terms as dense vectors where similar words are closer together in vector space. We use Word2Vec (Mikolov et al., 2013) to learn representations based on their contextual usage in our ES corpus. This produces word and phrase vectors where vectors close together in vector space have similar meanings based on context. We produce a list of 'Corpus Nearest' terms for each query in the 'fat head' zone of our Search Curve (this is the third column of Table 3).

3.4.3 Detecting Jargon/Terminology

Jargon/terminology consists of enterprise-specific words, phrases, expressions, and idioms that diverge from those universally familiar or understood outside of the organisation. In Set Theory, these divergent terms can represented by the set of elements both in Y and not in X:

$$\mathit{Divergent\ Terms} = \widetilde{X} \cap Y$$

where X is the set of LLM generalised terms and Y is the nearest neighbour terms with the ES corpus. The fourth column in Table 3 lists the divergent terms.

3.4.4 Jargony

Jaccard Distance (JD), also known as the Jaccard similarity coefficient, is a measure commonly used to calculate the similarity or dissimilarity between two sets of words such as those in columns 2 and 3 in Table 3. JD is particularly useful in scenarios where the presence or absence of elements is more important than their actual values. In this case, the Jaccard distance represents a measure of divergence. A higher distance suggests the divergent terms are more 'jargony' (i.e. more unlikely to be understood outside the enterprise). The calculated JD score is presented in the final column of Table 3). Note that the jargony score is applied to the query rather than to the divergent terms.

3.5 Learning to Rank Methodology

The Apache Solr 'weightExpression' parameter of the 'DocumentExpressionDictionaryFactory' dictionary implementation is used to score the suggestions.

Table 3: A comparison of the top synonyms for two examples of ‘fat head’ queries. The ‘Divergent Terms’ are those that commonly feature in the Enterprise Corpus but are not part of LLM’s common vocabulary.

’Fat Head’ Query	**LLM Nearest (X)**	**Corpus Nearest (Y)**	**Divergent Terms $(\widetilde{X \cap Y})$**	**Jaccard Distance**
scholarship	grant, bursary, fellowship, financial aid, award, stipend, tuitions assistance, academic fund, educational grant, study grant	foundation scholarship, scholarship examinations, entrance scholarships, scholarship exams, schols, visiting scholar	schols	0.65
id card	Identification Card, Identity Card, Personal Identification, Photo ID Card, Driver’s License, Passport, Employee Badge, Student Card, Membership Card, Official Documentation	id card, student card, student id, tcard	tcard	0.9

The ‘AnalyzingInfixLookupFactory’ Lookup Implementation allows for suggestions where the starting string does not necessarily match the query prefix. These numeric weights were calculated offline using the RankEval framework (Lucchese et al., 2020). Figure 3 depicts how each feature weighting contributes to ranked suggestions in our ES search box. The solrconfig.xml file is published on github[4]. The RankEval Python open-source tool (Lucchese et al., 2017; Lucchese et al., 2020), based on ensembles of decision trees, is then employed to determine the optimal relative feature weighting and calculate each feature’s contribution to the overall ranking efficiency.

3.5.1 Model

The ranking model is generated using the XGBoost implementation of the LambdaMART list-wise ranking algorithm[4]. Table 4 lists the hyper-parameters used.

3.6 MRR Metric Calculation

In our offline study, we break down the MRR scores for the three zones of our ES Search Demand Curve (Figure 2). We compute the MRR scores after three keystrokes. This breakdown helps us distinguish the QAC ranking performance for the most popular queries versus more obscure queries in the long tail zone.

Table 4: Hyper-parameter settings used to evaluate ranking performance for the Learning to Rank ranking method.

Parameter	**Value**
Algorithm	LambdaMark
Framework	XGBoost
max_depth	10
rank	MAP
num_round	10
eta	0.5

For the online study, MRR is computed for the users’ last keystroke (sometimes referred to as MRR@last (Chang and Demg, 2020)) since this is where the user selects a suggested candidate for submission to the search engine. A score of zero is used where no candidates are offered or in the case where users fail to select any of the offered candidates. We calculate MRR@k, where k is the number of offered completion candidates. We present results for k=10 (as is the norm) and k =7 because our live ES service presents a maximum of seven candidates.

The calculated MRR scores are initially computed for heuristic features, which serve as our baseline system (i.e., the ’A’ in our online A/B test). The online test will also give us an overall MRR score based on ‘real‘ data (i.e. the combined score across all of the Search Demand Curve zones).

[4] https://github.com/colindaly75/QAC_LTR_for_ES

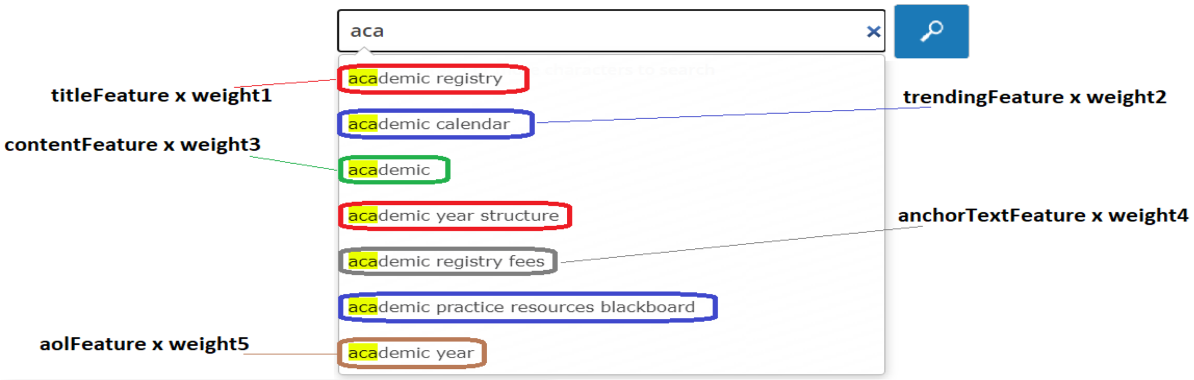


Figure 3: The typed ”aca” prefix presents a list of completion choices via a ranked list of prefix-candidate pairs. Candidates are generated from various sources/features, each of which is ‘weighted’ using LTR.

3.7 Ablation Study Methodology

Not all of the described ranking features are equally important. We perform an ablation study to better understand the contribution of each of the different features toward our QAC ranking model learning capability. We remove one feature from each iteration and perform again the LTR training and testing steps as before. If we observe a large decrease in QAC MRR scores, this indicates that the ablated feature is very important for the model. Our ablation study is carried out for MRR@{1,3,7 and 10}.

3.8 Online A/B Test Methodology

An online test will give us the overall MRR score (across all of the Search Demand Curve zones). Before commencing the A/B test, we perform an A/A test. Also known as a null test, the A/A test is used to establish trust in our experimental platform. This involves splitting the search requests into two pools, as in a regular A/B test, but where B is identical to A. If the scripts to record search requests and compute metrics from the logfiles are functioning consistently, we expect a t-test to prove that any difference in MRR results is not statistically significant (Kohavi et al., 2020).

Having established the integrity of our experimental platform, we subsequently undertake the A/B test to compare the ranking performance of QAC candidates using two pools: -

- A: This is the Control pool and encompasses all of the heuristic features.
- B: This Treatment pool includes A’s heuristic features *and* the additional QACES feature.

We use a load balancer with two servers in an active/active configuration, with a 50% traffic allocation to both the Control and Treatment groups. The load balancer has a session persistence (stickiness) parameter enabled so that the suggestion candidates are presented by the same back-end server that executes the submitted query. This ensures that each log file has a complete record.

Since the data collected from group A is independent of the data collected from group B, we use an *unpaired two-sample t-test* to validate statistical significance with α set to 0.05.

4 EVALUATION

In this section, we present the computed LTR weights for each feature, the results of the ablation study and the MRR scores for our offline study. Finally, we compare the MRR scores for our online evaluation (A/B test) of our ranking models, with and without the QACES feature.

4.1 LTR Weights

Table 5 shows the RankEval computed weighting associated with each of our ranking features.

Table 5: LTR weightings for features calculated using the RankEval framework.

Feature	**Weight**
MFQ	69
titleFeature	22
anchortextFeature	25
contentFeature	20
trendingFeature	9
aolFeature	6
QACES	4

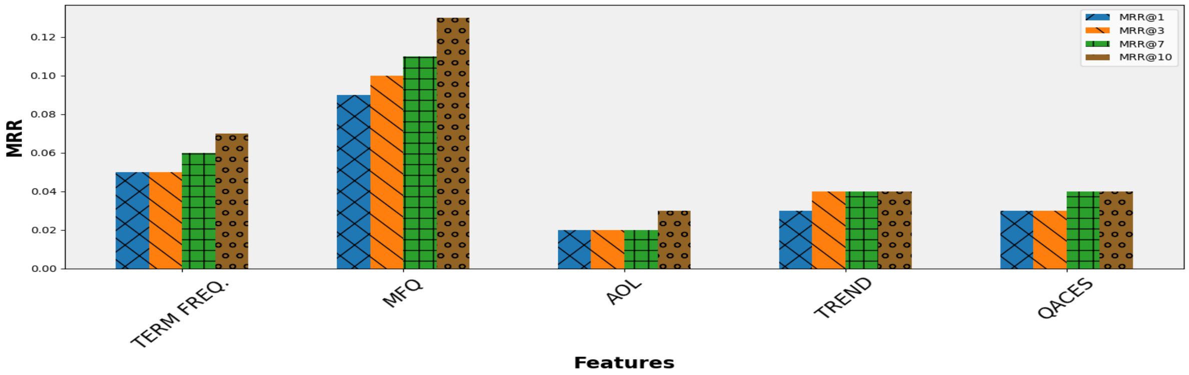


Figure 4: Ablation/leave-one-out analysis showing the contribution of individual features to the MRR performance across the QAC ranking model.

4.2 Offline Evaluation

The MRR scores for selected k-value cutoffs for the three Search Demand Curve zones are listed in Table 6. For comparison, we include the scores for the AOL WS dataset as well as our ES dataset. Across all zones, the AOL scores are consistently higher; we speculate this may be due to the use of a Personalization feature in the AOL WS dataset.

Singh et al. have achieved QAC MRR scores in the region of 0.485 for e-Commerce site search (Singh et al., 2023b), but any comparison is complicated as they omitted to break down scores into Search Demand Curve zones.

Table 6: Offline MRR scores for WS and ES data, with a breakdown for the sections of the Search Demand Curve.

Dataset / Zone	**MRR @1**	**MRR @7**	**MRR @10**
AOL WS Data			
Fat Head	0.72	0.75	**0.78**
Chunky Mid	0.29	0.36	0.36
Long Tail	0.23	0.25	0.29
All Zones	0.33	0.36	0.36
ES Data			
Fat Head	0.60	0.68	**0.68**
Chunky Mid	0.32	0.34	0.36
Long Tail	0.19	0.24	0.24
All Zones	0.29	0.33	0.34

4.3 Ablation Study Results

We performed an ablation analysis to better understand the contribution of each of the different features of our QAC ranking model (for MRR@{1,3,7 and 10}). Figure 4 shows their relative contribution totals.

We see that MFQ is the most important ranking feature, as its removal from the model results in a sharp decrease in MRR scores. This suggests that the collective query history is the best indicator of user intent. The AOL feature makes the smallest contribution to our model; this may be because many of the suggestions in the feature are deprecated or simply because the feature was not generated from our ES index. The contribution of our QACES jargon/terminology feature is comparable to that of the 'trend' feature.

4.4 Online Evaluation (A/B Test)

We evaluated two ranking models. The first (A) included heuristic features only. The second (B) also included our QACES feature. The ranking score for each ranking model is presented in Table 7. The A/B test was undertaken on the live Enterprise Search service of a large third-level education institution. The model was tested for 16 weeks on 140,000 queries, which resulted in a statistically significant increase in MRR score of +3.8% with a p-value < 0.05.

Table 7: A/B test results for ranking models with the percentage change in MRR score after implementation of the QACES feature.

Feature	**MRR@10**
Heuristic only	0.219 ± 0.01
With QACES	0.227 ± 0.02
Percentage change	+3.8%

The observed MRR@10 score in our online evaluation (0.227) is substantially lower than that calculated in our offline study (0.34). A possible cause is that search users type with speed and urgency and opt not to take the time to engage with the QAC interactive search offering, even where a candidate was an exact match to the query. We call this phenomenon

'QAC abandon' and speculate that it frequently occurs in the case of 'navigational searches' (i.e. returning users who already have a good idea of what document they are looking for).

5 CONCLUSIONS AND FUTURE WORK

In this paper, we discuss the unique requirements for QAC in Enterprise Search and demonstrate the implementation of a QAC ranking model using features whose weightings are computed using Learning to Rank.

We hypothesise and prove that adding our QACES LLM-based jargon/terminology ranking feature to our baseline heuristic LTR model results in a statistically significant improvement to QAC ranking performance (MRR score) on a live Enterprise Search service.

A limitation of the GPT-4 language processing model is that, unlike WordNet, it does not always produce repeatable results. For example, we may get a slightly different list of synonyms for the same query each time it is run. While this does not influence the general reproducibility of results, it may affect the consistency of the generated list of jargon/terminology terms. Our initial investigation of the GPT-4o LM also seems to produce more consistent results and this will be detailed in future work.

Another idea for future work is to explore how our QACES innovation could be adapted for use with query expansion. Finally, an investigation of the causes and factors that affect 'QAC abandon' would be an interesting new direction for the field of autocompletion.

ACKNOWLEDGEMENTS

This research was conducted with the financial support of Science Foundation Ireland under Grant Agreement No. 13/RC/2106_P2 at the ADAPT SFI Research Centre at Trinity College Dublin. ADAPT, the SFI Research Centre for AI-Driven Digital Content Technology, is funded by Science Foundation Ireland through the SFI Research Centres Programme. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

Balloccu, S., Schmidtová, P., Lango, M., and Dušek, O. (2024). Leak, Cheat, Repeat: Data Contamination and Evaluation Malpractices in Closed-Source LLMs. *EACL 2024 - 18th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference*, 1:67–93.

Bar-Yossef, Z. and Kraus, N. (2011). Context-sensitive query auto-completion. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pages 107–116.

Bentley, J. (2011). Mind the Enterprise Search Gap: Smartlogic Sponsor MindMetre Research Report.

Cai, F. and De Rijke, M. (2016). A survey of query auto completion in information retrieval. *Foundations and Trends in Information Retrieval*, 10(4):273–363.

Chang, Y. and Demg, H. (2020). *Query Understanding for Search Engines*, volume 46 of *The Information Retrieval Series*. Springer International Publishing, Jilin.

Cleverley, P. H. and Burnett, S. (2019). Enterprise search and discovery capability: The factors and generative mechanisms for user satisfaction:. *Journal of Information Science*, 45(1):29–52.

Craswell, N., Cambridge, M., and Soboroff, I. (2005). Overview of the TREC-2005 Enterprise Track. In *TREC 2005 conference notebook*, pages 199–205.

Croft, W. B. (2010). Search engines : information retrieval in practice / by Bruce Croft, Donald Metzler, Trevor Strohman.

Daly, C. (2023). Learning to Rank: Performance and Practical Barriers to Deployment in Enterprise Search. In *3rd Asia Conference on Information Engineering (ACIE)*, pages 21–26. IEEE.

Davis, M., Eager, A., Edwards, R., Ganesh, B., Holt, M., and Mukherjee, S. (2011). Enterprise Search and Retrieval 2011/2012. In James, M., editor, *Technology Evaluation and Comparison Report*, page 277. OVUM.

Duan, H. and Hsu, B. J. (2011). Online spelling correction for query completion. *Proceedings of the 20th International Conference on World Wide Web, WWW 2011*, pages 117–126.

Fiorini, N. and Lu, Z. (2018). Personalized neural language models for real-world query auto completion. *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 3:208–215.

Guo, J., Fan, Y., Ai, Q., and Croft, W. B. (2016). A deep relevance matching model for Ad-hoc retrieval. In *International Conference on Information and Knowledge Management, Proceedings*, volume 24-28-Octo, pages 55–64. Association for Computing Machinery.

Hayes, B. (2008). The Britney Spears Problem. *American Scientist*, 96(4):274.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Kim, G. (2019). Subword Language Model for Query Auto-Completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5022–5032, Hong Kong, China. Association for Computational Linguistics.

Kohavi, R., Tankg, D., and Xu, Y. (2020). *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing - Ron Kohavi, Diane Tang, Ya Xu - Google Books*. Cambridge University Press, Cambridge.

Kritzinger, W. T. and Weideman, M. (2013). Search Engine Optimization and Pay-per-Click Marketing Strategies. *Journal of Organizational Computing and Electronic Commerce*, 23(3):273–286.

Kruschwitz, U. and Hull, C. (2017). Searching the Enterprise. *Foundations and Trends® in Information Retrieval*, 11(1):1–142.

Kruschwitz, U., Lungley, D., Albakour, M. D., and Song, D. (2013). Deriving query suggestions for site search. *Journal of the American Society for Information Science and Technology*, 64(10):1975–1994.

Li, H. (2011). A Short Introduction to Learning to Rank. *IEICE Transactions*, 94-D:1854–1862.

Li, L., Deng, H., Dong, A., Chang, Y., Baeza-Yates, R., and Zha, H. (2017). Exploring query auto-completion and click logs for contextual-aware web search and query suggestion. *26th International World Wide Web Conference, WWW 2017*, pages 539–548.

Lucchese, C., Muntean, C., Nardini, F., Perego, R., and Trani, S. (2020). RankEval: Evaluation and investigation of ranking models. *SoftwareX*, 12:100614.

Lucchese, C., Muntean, C. I., Nardini, F. M., Perego, R., and Trani, S. (2017). RankEval: An evaluation and analysis framework for learning-To-rank solutions. *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1281–1284.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems*, 26.

OpenAI 2023 (2023). OpenAI GPT-4.

Pass, G., Chowdhury, A., and Torgeson, C. (2006). A picture of search. *ACM International Conference Proceeding Series*, 152.

Princeton University (2010). About WordNet.

Rahangdale, A. and Raut, S. (2019). Deep Neural Network Regularization for Feature Selection in Learning-to-Rank. *IEEE Access*, 7:53988–54006.

Scott, E. (2022). 9 UX Best Practice Design Patterns for Autocomplete Suggestions (Only 19% Get Everything Right) – Articles – Baymard Institute.

Singh, S., Farfade, S., and Comar, P. M. (2023a). Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete. *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pages 469–474.

Singh, S., Farfade, S., and Comar, P. M. (2023b). Multi-Objective Ranking to Boost Navigational Suggestions in eCommerce AutoComplete. *ACM Web Conference 2023 - Companion of the World Wide Web Conference, WWW 2023*, pages 469–474.

The Apache Software Foundation. (2004). Apache Solr.

Turnbull, D. and Berryman, J. (2016). *Relevant Search*. Manning Publications Co., New York.

White, M. (2018). *Enterprise search*. O'Reilly Media, Sebastopol, CA.

White, R. W. and Marchionini, G. (2007). Examining the effectiveness of real-time query expansion. *Information Processing & Management*, 43(3):685–704.

Xu, J., Wei, Z., Xia, L., Lan, Y., Yin, D., Cheng, X., and Wen, J.-R. (2020). Reinforcement Learning to Rank with Pairwise Policy Gradient. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, volume 10, page 10, New York, NY, USA. ACM.

Yadav, N., Sen, R., Hill, D. N., Mazumdar, A., and Dhillon, I. S. (2021). Session-Aware Query Auto-completion using Extreme Multi-Label Ranking. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3835–3844.

Machine Learning Unravels Sex-Specific Biomarkers for Atopic Dermatitis

Ana Duarte[a] and Orlando Belo[b]
Algoritmi R&D Centre / LASI, University of Minho, Campus of Gualtar, 4710-057 Braga, Portugal
id9618@alunos.uminho.pt, obelo@di.uminho.pt

Keywords: Atopic Dermatitis, Machine Learning, Gene Signature, Sex-Specific Biomarker, Precision Medicine.

Abstract: The prevalence of atopic dermatitis is significantly higher in women than in men. Understanding the differences in the manifestation of the disease between males and females can contribute to more tailored and effective treatments. Our goal in this paper was to discover sex-specific biomarkers that can be used to differentiate between lesional and non-lesional skin in atopic dermatitis patients. Using transcriptomic datasets, we first identified the genes with the highest expression difference. Subsequently, several feature selection methods and machine learning models were employed to select the most relevant genes and identify potential candidates for sex-specific biomarkers. Based on backward feature elimination, we obtained a male-specific signature with 11 genes and a female-specific signature with 10 genes. Both candidate signatures were properly evaluated by an ensemble classifier using an independent test. The obtained AUC and accuracy values for the male signature were 0.839 and 0.7222, respectively, and 0.65 and 0.6667 for the female signature. Finally, we tested the male signature on female data and the female signature on male data. As expected, the analysed metrics decreased considerably in these scenarios. These results suggest that we have identified two promising sex-specific gene signatures, and support that sex affects the ability to distinguish lesions in patients with eczema.

1 INTRODUCTION

Atopic dermatitis (AD) is a chronic and highly complex inflammatory skin condition that significantly reduces the quality of life of patients. In Europe and the USA, one in five children and 10% of adults are diagnosed with AD (Bylund et al., 2020; Laughter et al., 2021). Probably due to socioeconomic and environmental changes, including the growing levels of urbanisation and industrialisation, the prevalence of AD is increasing worldwide, having particularly alarming rates in low-income countries (Nutten, 2015; Schuler et al., 2023; Skevaki et al., 2021; Tsai et al., 2019). The incidence of the disease varies significantly between different geographical regions and cultures, age, sex, and ethnicity (Mesjasz et al., 2023; Nutten, 2015; Schuler et al., 2023). The reasons for this heterogeneity are not clearly understood. They require further investigation and a deeper understanding of how the combination of the multiple factors involved affects the susceptibility, development, progression, and treatment of the disease.

Numerous diseases, including AD, present a broad spectrum of clinical manifestations, unpredictable courses, and variable responses to therapy. In this regard, the effective management of these complex diseases associated with multiple phenotypes and endotypes requires a precision medicine-based strategy. Concretely, AD is among the disorders that can benefit most from more personalised and targeted interventions (Muraro et al., 2016). However, in contrast to other diseases, precision medicine in AD is still in its early stages. Despite the progress made in recent decades, the clinical reality is still not based on a multifactorial approach tailored to the needs of each patient (Mesjasz et al., 2023; Muraro et al., 2016). Currently, the main challenges for the development of precision medicine in AD are the discovery of new effective therapies with few side effects, taking into account the profile of each patient. Prescribing the most appropriate therapies for each profile presupposes the identification of the pathophysiological mechanisms associated with the disease (Arkwright and Koplin, 2023; Leung, 2024). Particularly, the identification of biomarkers capable of distinguishing lesional from non-lesional skin in AD patients may facilitate the

[a] https://orcid.org/0000-0001-6505-9888
[b] https://orcid.org/0000-0003-2157-8891

Duarte, A. and Belo, O.
Machine Learning Unravels Sex-Specific Biomarkers for Atopic Dermatitis.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 27-35
ISBN: 978-989-758-716-0; ISSN: 2184-3228

understanding of the mechanisms involved in the development of lesions. A thorough comprehension of these mechanisms is essential to intervene with targeted therapies in the critical pathways. Gene expression profiling based on transcriptomic data can be used to compare lesional and non-lesional skin tissues and identify the key biomarkers involved. Given the multiplicity of factors that influence the disease, the inclusion of supplementary patient characteristics such as sex, age, or existing comorbidities can lead to more accurate biomarker detection. Hence, a holistic approach with the creation of more complex profiles can be tremendously valuable for the management and control of the disease, as well as for enhancing its treatment (Muraro et al., 2016).

With this paper, we expect to contribute to the advancement of precision medicine in AD. Specifically, we aim to discover candidate biomarkers for AD by applying machine learning (ML) algorithms to gene expression data of two distinct patient profiles. ML and classical statistical methods have demonstrated great potential in biomedicine, namely in the processing of high-dimensional datasets such as those used in the identification of gene signatures (Karthik and Sudha, 2018; Liu et al., 2022). We also hypothesise that sex plays a role in identifying a reliable gene signature to differentiate lesional from non-lesional skin in AD patients. For this reason, we speculate that males and females have different molecular mechanisms involved in the manifestation of AD and, consequently, a separate analysis is required.

The remaining part of this paper is organised as follows. Section 2 reports some of the literature that addresses the application of ML techniques to the discovery of gene signatures, and the influence of sex on the manifestation of AD and other diseases. Section 3 summarises the methodology followed to identify the gene signatures and section 4 presents and discusses the results obtained. Finally, Section 5 concludes with an overview of the main findings and possible future research directions.

2 GENE SIGNATURES FOR AD: OPPORTUNITIES AND CHALLENGES

Despite the urgent need to find candidate gene signatures that could revolutionise current dermatological care, very little research has been conducted at the AD level using ML algorithms. In fact, most research applying ML to advance precision medicine focuses on the most fatal diseases, such as cancer. Nevertheless, a few ML-based studies have explored the identification of biomarker candidate genes for AD. One such example is the work developed by Zhong et al. (Zhong et al., 2021). Based on a bioinformatics approach and using LASSO, the authors used transcriptomic datasets and identified GZMB, CXCL1 and CD274 as potential biomarkers to distinguish AD lesions from non-lesions. On the other side, Möbus and colleagues (Möbus et al., 2022), also based on transcriptomic datasets, observed two distinct endotypes for AD associated with notable clinical differences, allowing patients to be stratified into eosinophil-high and eosinophil-low groups. Implementing the Boruta algorithm and a random forest model, the authors identified the most relevant genes to predict the clusters to which the patients belong. Both investigations demonstrate that some key genes are promising candidate biomarkers that have a major impact on the diagnosis and management of AD. However, these studies do not explore the different phenotypes of the patients, such as age, sex, or ethnicity, which are known to play a preponderant role in the manifestation of the disease.

In contrast, other researchers have analysed the discovery of differentiated biomarkers depending on the phenotypic characteristics of the patients, namely sex. For example, Moon et al. (Moon et al., 2013) proposed a procedure to find sex-specific biomarkers based on three datasets from patients with acute myeloid leukaemia, chronic lymphocytic leukaemia, and cutaneous melanoma. The considered methodology consisted of an algorithm based on the importance of each feature in order to extract the top-ranked genes for male and female patients. The selected genes were properly tested and the results obtained revealed high accuracy values, confirming the validity of the strategy followed and underlining the relevance of sex-specific biomarkers for enhancing prognosis prediction. Further experiments have also been conducted to unveil sex-specific biomarkers for different diseases. For instance, some papers exploit the use of ML methods to find sex-specific biomarkers for Alzheimer's disease, emphasising the importance of considering clinical features in addition to genes for a more thorough and sensitive analysis (Bourquard et al., 2023; Ji et al., 2022).

As far as we know, no previous research has addressed the identification of sex-specific genes for AD. However, many studies have suggested that sex has a significant contribution to the prevalence and severity of the disease. For example, Johansson et al. examined the distribution and characteristics of AD in a Swedish population and concluded that the disease is more common in females among young adults (Johansson et al., 2022). Similarly, Kiiski et al. noted that in a Finnish cohort, the prevalence of AD was higher among women aged 30 to 49 years than among men of the same age (Kiiski et al., 2022). These results are consistent with other investigations conducted in other countries, such as Italy (Pesce et al., 2015) or the USA (Silverberg and Hanifin, 2013), where the female sex also appears to be a risk factor for AD. Therefore, we argue that the process of searching for suitable candidate gene signatures for AD requires a sex-separated analysis. Accordingly, this paper presents a novel contribution to the identification of sex-specific biomarkers for AD based on a ML approach.

3 MATERIALS AND METHODS

From a broader perspective, the methodology considered in our proposal can be divided into two major stages. Each of these stages must be performed independently for the male and female data. First, two datasets sharing the same platform were used to identify the differentially expressed genes (DEGs). Subsequently, based on the selected DEGs, an extra dataset was also taken into account to determine gene signatures for AD. A more detailed description of the sequential processes considered at each stage is provided in Figure 1.

3.1 Data Gathering and Exploration

The datasets used to conduct this project were obtained from the Gene Expression Omnibus[1] (GEO), a public repository containing experimental gene expression data. A preliminary search was performed in order to obtain transcriptomic datasets for AD with information about the sex of the patients. With this objective in mind, we selected the datasets GSE130588, GSE58558, and GSE150797. Since some academics state that datasets from different manufacturers should not be combined to avoid potential bias in the data, all these data sources were generated by the Affymetrix manufacturer (Liu et al., 2021; Serio, 2023). These datasets contain normalised microarray data from skin samples collected from AD patients. As each of these datasets was designed with the purpose of assessing the patients' response to treatments, only the samples that referred to the start of the therapies were considered. Furthermore, we took into account only patients with lesional (AD-L) or non-lesional (AD-NL) AD – Table 1 indicates the main characteristics of the selected data.

Combining the three datasets, the number of samples by sex is balanced. In total, 87 samples correspond to male patients and 88 samples to female patients. Particularly, 51 samples from males refer to AD-L, while 36 refer to AD-NL. Conversely, 49 samples from females correspond to AD-L and 39 to AD-NL.

3.2 Differential Expressed Genes

Since the identified DEGs may have a significant impact on the overall results, we need to plan the differential gene expression analysis in order to avoid the exclusion of important genes. Even if they are from the same manufacturer, datasets from distinct platforms are more likely to introduce bias into the data (Campain and Yang, 2010). For this reason, we decided to perform the differential gene expression analysis using exclusively the datasets produced on the same platform, i.e., datasets GSE130588 and GSE58558.

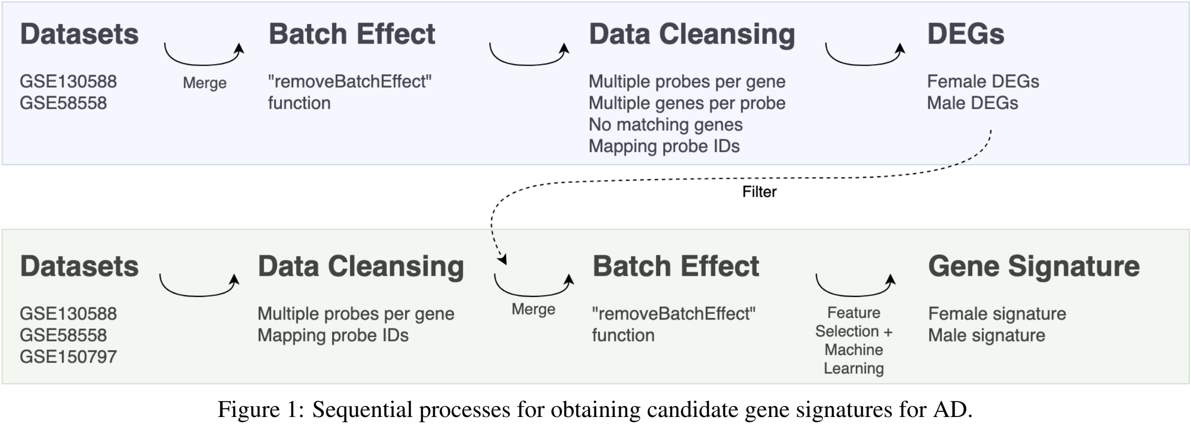


Figure 1: Sequential processes for obtaining candidate gene signatures for AD.

[1] https://www.ncbi.nlm.nih.gov/geo/

Table 1: Summary of the properties of the datasets used.

					Samples	
Dataset	Manufacturer	Platform	Requirements	Sex	AD-L	AD-NL
GSE130588	Affymetrix	GPL570	Time: week 0	Female	22	21
			Tissue: LS or NL	Male	29	21
GSE58558	Affymetrix	GPL570	Time: day 1	Female	6	7
				Male	12	10
GSE150797	Affymetrix	GPL23159	Treatment:	Female	21	11
			untreated	Male	10	5

The first step in determining the DEGs was to divide each of the datasets into two groups, each corresponding to a profile (male or female). Each profile was analysed separately in order to find specific DEGs for the group of male patients and specific DEGs for the group of female patients. All the necessary processes were performed in R (version 4.3.1) using the limma package. One of the first data treatments was to merge the datasets associated with the same profile, i.e., the male samples in the GSE130588 dataset were combined with the male samples in the GSE58558 dataset, and the same was done for the female samples. As a result, we obtained a specific dataset for males with 41 AD-L and 31 AD-NL samples and a dataset for females with 28 AD-L and 28 AD-NL records. Since the merged data came from different experiments, we corrected the batch effect using the "removeBatchEffect" function from the limma package. Moreover, because we were working with microarray data, some additional cleansing tasks were also required for the following encountered situations:

- **Multiple probes corresponding to the same gene –** only the probe with the highest average expression was kept, and the other probes were discarded. Ties in average counts were resolved by choosing one of the probes and eliminating the rest (Miller et al., 2011).
- **Probes associated with various genes –** these records were removed (Hu et al., 2023).
- **Probes that did not match a specific gene –** these records were removed (Wang and Yu, 2023).

To finalise the data treatment, the probe IDs were converted into their corresponding gene symbols. Finally, genes with an absolute $\log_2(\text{fold change}) \geq 1$ and $p_{\text{adj}} < 0.05$ were identified as DEGs and saved in text files.

3.3 Gene Selection Strategy

After identifying the DEGs for each profile, we prepared the datasets GSE130588, GSE58558, and GSE150797 for feature selection and ML modelling. As with the differential expression analysis, this process was also performed using R, analysing each patient profile separately. Each of the three datasets was therefore split into male and female groups, resulting in a total of six subsets. For each subset, we checked the existence of multiple probes corresponding to the same gene and addressed the issue with a similar approach to the one we used to identify the DEGs. Probe IDs were also transformed into the corresponding gene symbols. By importing the text file containing the determined DEGs, we filtered the data in order to have only DEGs. For each profile, the corresponding subsets were merged, and the batch effect was removed. The resulting male and female datasets were then ready to support the next steps of the work.

The following tasks, including additional data processing, feature selection, and construction of ML models, were performed in Python 3.6 using the scikit-learn library. All these tasks were applied in parallel to the male and female datasets. For each dataset, we divided the data into train (80%) and test (20%), using a stratified strategy to maintain the proportion of AD-L and AD-NL samples in both sets. Only the training data were used for feature selection and for the construction of ML classifiers to identify potential gene signatures. For ML modelling, we considered a shuffled and stratified 5-fold cross-validation. The optimal hyperparameters for each algorithm were found using BayesSearchCV with 30 iterations.

Although the identification of DEGs helps to reduce the dimensionality of the data, the resulting high number of genes is still a drawback for efficient processing by ML methods. Feature selection is a common strategy used to minimise these gaps. Therefore, before building ML algorithms, we conducted a feature selection approach using Boruta, Support Vector Machine Recursive Feature Elimination (SVM-RFE), and Least Absolute Shrinkage and Selection Operator (LASSO). These three methods reduced the number of genes differently and thus originated three distinct gene sets. The Random Forest (RF), XGBoost, Ad-

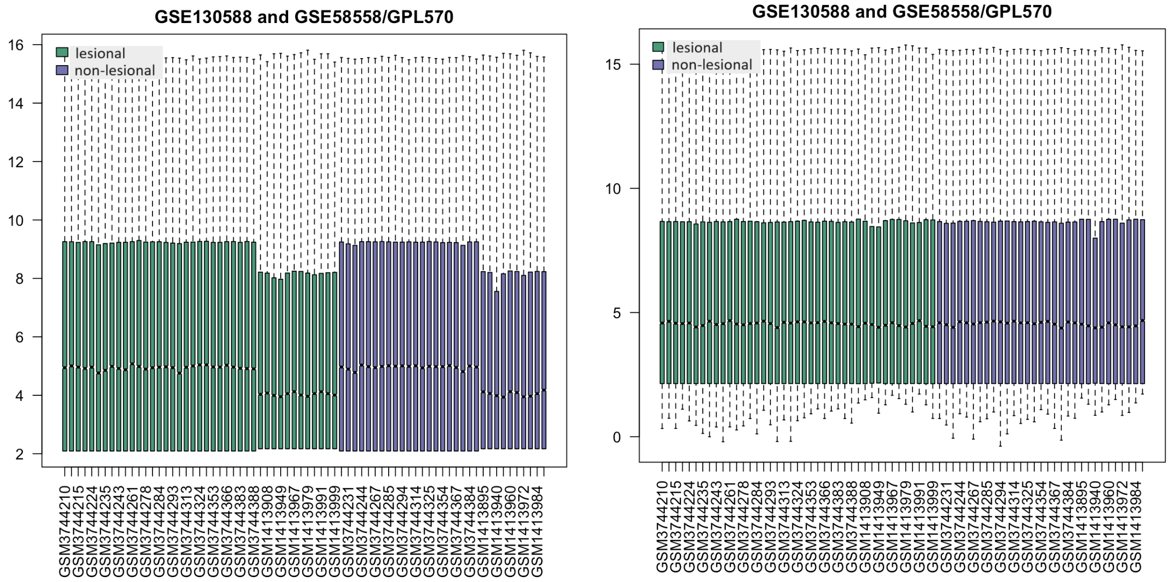


Figure 2: Before (left) and after (right) batch effect correction in the male profile.

aBoost, linear Support Vector Machine (SVM), and Logistic Regression (LR) methods were applied to each of these sets. For each ML method, we extracted the importance of each gene and used the Min-Max method to normalise the obtained value between 0 and 1. Thus, for each of the three gene sets, we obtained the normalised importance values of each gene for each algorithm, which we then summed to generate a gene score. The top genes with significantly higher scores were selected as candidate genes for the creation of a gene signature.

From the selected set of candidate genes, we conceived new ML models using the same five methods and constructed a soft-voting classifier that combines the predictions of the models. This ensemble method was used to evaluate the predictive power of the models, considering both AUC and accuracy values. Adopting a backward feature elimination strategy, we discarded the least important gene and created new classifiers to compare the AUC and accuracy values with the prior solution. These steps were performed iteratively until the performance metrics were worse than the values obtained in the previous step. Thus, the genes that were not eliminated by this iterative process became part of our proposed gene signature. In the end, we obtained one candidate gene signature for males and another for females.

To test our initial hypothesis that sex must be taken into account when establishing gene signatures for AD, we used the independent test to compare the AUC and accuracy values obtained when the male signature is applied to the male and female datasets and vice versa. The additional step of testing the male signature against the female dataset and vice versa was necessary to determine whether the identified genes can be considered sex-specific candidates.

4 RESULTS

4.1 DEGs Identification and Initial Gene Selection

At the beginning of the differential gene expression analysis, two distinct groups were identified after merging the datasets GSE130588 and GSE58558. In both profiles, the samples belonging to the same datasets had similar expression values, but these values were significantly different from the expression values of the samples in the other dataset. Batch effect correction removed these technical differences. Figure 2 shows the box plots obtained before and after removing the batch effect in the male profile. For females, the batch effect correction led to similar results. After processing the data, we identified 188 and 764 DEGs for the male and female datasets, respectively.

Once the DEGs were determined, we proceeded to the individual treatment of the three datasets, which involved filtering the genes according to the DEGs found. As the GSE150797 dataset was generated using a different platform, some DEGs did not match the probe IDs. Consequently, after merging the datasets, the number of DEGs was reduced to 172 in the male profile and 700 in the female scenario. Moreover, we also observed that there were larger differences between the GSE150797 dataset (GPL23159 platform) and the datasets from the GPL570 platform before batch effect correction. After removing the batch effect, the existing differences were successfully minimized. Boruta, SVM-RFE and LASSO yielded the three gene subsets indicated in Table 2.

Table 2: Number of selected genes by each feature selection approach.

Profile	Boruta	SVM-RFE	LASSO
Male	12	90	62
Female	23	75	92

4.2 Determination of Sex-Specific Gene Signatures

Since the number of DEGs in the male dataset is relatively small, in this case, we decided to include an additional scenario corresponding to the training and validation of the ML algorithms without using any feature selection method. Table 3 lists the top genes obtained after implementing the ML process and determining the scores. In general, the different feature selection strategies identified the top genes consistently for the same profiles.

Based on the results, we selected an initial set of 11 candidate genes for males, and for females, we considered the 16 most important genes (Table 3, highlighted in bold). Our backward feature elimination strategy led to the discovery of an optimal 11-gene signature specific to males (KIF2C, AKR1B10, PHYHIP, FOSL1, FPR1, HS3ST3A1, MX1, KANK4, PPARG, BCL2A1, and KLHDC7B) and a 10-gene signature specific to females (CEP126, FCHSD1, C17orf96, IL18RAP, P2RY10, PTAFR, ANKFN1, TBX18, P2RY2, and AEN). Interestingly, none of the genes are common to both signatures. This may indicate that the molecular pathways involved in the development of AD lesions may differ between men and women, suggesting that the sex of the patients should be considered for better disease management and treatment. The genes KANK4, PHYHIP and PPARG from the male profile were downregulated in the lesions, while the rest were upregulated. In the female profile, only ANKFN1, CEP126 and TBX18 were downregulated, while all others were upregulated. There is scientific evidence that some of these genes may have a major impact on AD. For example, the downregulation of the PPARG gene, identified in the male signature, may be associated with inflammation, keratinisation, and sebaceous gland function (Konger et al., 2021). On the other hand, some studies suggest that P2Y receptors, such as the P2RY10 and P2RY2 genes found in the female signature, can be involved in skin inflammation (Pastore et al., 2007).

Table 4 presents the AUC and accuracy values obtained for each candidate signature using the voting classifier. A closer analysis of the results shows that the AUC and accuracy values of the male signature in the independent test are considerably high when applied to the male data (0.839 and 0.7222, respectively). However, when this signature is tested with the female data, the AUC (0.575) and accuracy (0.6111) values deteriorate substantially. Although the difference is not as marked, the same is true for the female signature. The AUC and accuracy values of the female signature when applied to the female data are 0.650 and 0.6667, respectively. These values decrease when the female signature is tested on the male dataset (AUC = 0.552 and accuracy = 0.6111). These findings thus demonstrate that considering sex-specific biomarkers leads to improved gene signatures for distinguishing lesions from non-lesions, reinforcing the benefits of a sex-separate analysis to establish candidate gene signatures for AD.

Table 3: Top genes identified by the feature selection methods for each profile.

Male profile

Boruta	SVM-RFE	LASSO	All DEGs
KIF2C	KIF2C	FOSL1	KIF2C
AKR1B10	FOSL1	KIF2C	FOSL1
PHYHIP	FPR1	MX1	PHYHIP
FOSL1	**MX1**	PHYHIP	AKR1B10
FPR1	**KANK4**	HS3ST3A1	**KLHDC7B**
HS3ST3A1	**PPARG**	KANK4	FPR1
		BCL2A1	HS3ST3A1

Female profile

Boruta	SVM-RFE	LASSO
CEP126	FCHSD1	C17orf96
FCHSD1	C17orf96	IL18RAP
C17orf96	IL18RAP	MS4A14
GNA15	**PTAFR**	STRIP2
IL18RAP	**WIF1**	**TBX18**
P2RY10	**STRIP2**	**P2RY2**
	PLAG1	**AEN**
	MS4A14	GNA15
	ANKFN1	**HSD11B1**

Table 4: AUC and accuracy values of the optimal gene signatures when tested against the male and female datasets.

		Male data		Female data	
		AUC	accuracy	AUC	accuracy
Male signature	train	0.9737	0.8099	0.8543	0.8429
	test	0.839	0.7222	0.575	0.6111
Female signature	train	0.76	0.6659	0.975	0.9286
	test	0.552	0.6111	0.650	0.6667

5 CONCLUSION AND FUTURE WORK

AD presents an unequal distribution between men and women, and its incidence is increasing worldwide. Nevertheless, there are very few studies on this disease to identify biomarkers through ML techniques. Specifically, we have not found any scientific study aimed at finding sex-specific biomarkers for the disease. This could be of particular relevance to gain deeper insights into how AD manifests in men and women. To fill this gap in the literature, we developed a ML approach using transcriptomic datasets and intended to identify male and female biomarkers that distinguish normal from lesional skin in patients with atopic eczema.

Our research led to the definition of a male-specific gene signature consisting of the KIF2C, AKR1B10, PHYHIP, FOSL1, FPR1, HS3ST3A1, MX1, KANK4, PPARG, BCL2A1, and KLHDC7B genes, and a female-specific gene signature comprising the CEP126, FCHSD1, C17orf96, IL18RAP, P2RY10, PTAFR, ANKFN1, TBX18, P2RY2, and AEN genes. For some of the identified genes, there is evidence in the literature to support their possible influence on the skin. The difference between the genes of the two signatures could indicate that different mechanisms are involved in the manifestation of AD in men and women. A better understanding of these mechanisms could promote the emergence of targeted treatments and contribute to the development of precision medicine in AD.

Although the results obtained emphasise the need to investigate sex-specific biomarkers for AD, our study has certain limitations. The main shortcomings are the limited number of samples and the lack of public databases providing gene expression data in combination with clinical phenotypes. Therefore, new studies on this topic and the availability of new datasets integrating transcriptomic and phenotypic data are currently a priority. It would also be valuable for future research to replicate the proposed methodology to other diseases, particularly those where it is suspected that different molecular mechanisms may be involved depending on the sex. In addition, a thorough investigation of the discovered biomarkers as well as the associated molecular mechanisms is required to gain a comprehensive understanding of how men and women differ in the development of AD lesions. Finally, any research of this nature requires clinical validation.

ACKNOWLEDGEMENTS

This work has been supported by FCT – Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020, and the PhD grant: 2022.12728.BD.

REFERENCES

Arkwright, P. D. and Koplin, J. J. (2023). Challenging best practice of atopic dermatitis. *Journal of Allergy and Clinical Immunology: In Practice*, 11:1391–1393.

Bourquard, T., Lee, K., Al-Ramahi, I., Pham, M., Shapiro, D., Lagisetty, Y., Soleimani, S., Mota, S., Wilhelm, K., Samieinasab, M., Kim, Y. W., Huh, E., Asmussen, J., Katsonis, P., Botas, J., and Lichtarge, O. (2023). Functional variants identify sex-specific genes and pathways in alzheimer's disease. *Nature Communications*, 14.

Bylund, S., Kobyletzki, L. B. V., Svalstedt, M., and Åke Svensson (2020). Prevalence and incidence of atopic dermatitis: A systematic review. *Acta Dermato-Venereologica*, 100:320–329.

Campain, A. and Yang, Y. H. (2010). Comparison study of microarray meta-analysis methods. *BMC Bioinformatics*, 11.

Hu, Y., Chen, X., Mei, X., Luo, Z., Wu, H., Zhang, H., Zeng, Q., Ren, H., and Xu, D. (2023). Identification of diagnostic immune-related gene biomarkers for predicting heart failure after acute myocardial infarction. *Open Medicine*, 18.

Ji, W., An, K., Wang, C., and Wang, S. (2022). Bioinformatics analysis of diagnostic biomarkers for alzheimer's disease in peripheral blood based on sex differences and support vector machine algorithm. *Hereditas*, 159.

Johansson, E. K., Bergström, A., Kull, I., Melén, E., Jonsson, M., Lundin, S., Wahlgren, C. F., and Ballardini, N. (2022). Prevalence and characteristics of atopic dermatitis among young adult females and males - report from the swedish population-based study bamse. *Journal of the European Academy of Dermatology and Venereology*, 36:698–704.

Karthik, S. and Sudha, M. (2018). A survey on machine learning approaches in gene expression classification in modelling computational diagnostic system for complex diseases. *International Journal of Engineering and Advanced Technology (IJEAT)*, 8:182–191.

Kiiski, V., Salava, A., Susitaival, P., Barnhill, S., Remitz, A., and Heliovaara, M. (2022). Atopic dermatitis in adults: a population-based study in finland. *International Journal of Dermatology*, 61:324–330.

Konger, R. L., Derr-Yellin, E., Zimmers, T. A., Katona, T., Xuei, X., Liu, Y., Zhou, H.-M., Simpson, E. R., and Turner, M. J. (2021). Epidermal pparγ is a key homeostatic regulator of cutaneous inflammation and barrier function in mouse skin. *International Journal of Molecular Sciences*, 22.

Laughter, M. R., Maymone, M. B., Mashayekhi, S., Arents, B. W., Karimkhani, C., Langan, S. M., Dellavalle, R. P., and Flohr, C. (2021). The global burden of atopic dermatitis: lessons from the global burden of disease study 1990–2017. *British Journal of Dermatology*, 184:304–309.

Leung, D. Y. (2024). Evolving atopic dermatitis toward precision medicine. *Annals of allergy, asthma & immunology*, 132:107–108.

Liu, J., Liu, L., Antwi, P. A., Luo, Y., and Liang, F. (2022). Identification and validation of the diagnostic characteristic genes of ovarian cancer by bioinformatics and machine learning. *Frontiers in Genetics*, 13.

Liu, L., Wang, T., Huang, D., and Song, D. (2021). Comprehensive analysis of differentially expressed genes in clinically diagnosed irreversible pulpitis by multiplatform data integration using a robust rank aggregation approach. *Journal of Endodontics*, 47:1365–1375.

Mesjasz, A., Kołkowski, K., Wollenberg, A., and Trzeciak, M. (2023). How to understand personalized medicine in atopic dermatitis nowadays? *International Journal of Molecular Sciences*, 24.

Miller, J. A., Cai, C., Langfelder, P., Geschwind, D. H., Kurian, S. M., Salomon, D. R., and Horvath, S. (2011). Strategies for aggregating gene expression data: The collapserows r function. *BMC Bioinformatics*, 12.

Moon, H., Lopez, K. L., Lin, G. I., and Chen, J. J. (2013). Sex-specific genomic biomarkers for individualized treatment of life-threatening diseases. *Disease Markers*, 35:661–667.

Muraro, A., Lemanske, R. F., Hellings, P. W., Akdis, C. A., Bieber, T., Casale, T. B., Jutel, M., Ong, P. Y., Poulsen, L. K., Schmid-Grendelmeier, P., Simon, H. U., Seys, S. F., and Agache, I. (2016). Precision medicine in patients with allergic diseases: Airway diseases and atopic dermatitis - practall document of the european academy of allergy and clinical immunology and the american academy of allergy, asthma & immunology. *Journal of Allergy and Clinical Immunology*, 137:1347–1358.

Möbus, L., Rodriguez, E., Harder, I., Boraczynski, N., Szymczak, S., Hübenthal, M., Stölzl, D., Gerdes, S., Kleinheinz, A., Abraham, S., Heratizadeh, A., Handrick, C., Haufe, E., Werfel, T., Schmitt, J., and Weidinger, S. (2022). Blood transcriptome profiling identifies 2 candidate endotypes of atopic dermatitis. *Journal of Allergy and Clinical Immunology*, 150:385–395.

Nutten, S. (2015). Atopic dermatitis: Global epidemiology and risk factors. *Annals of Nutrition and Metabolism*, 66:8–16.

Pastore, S., Mascia, F., Gulinelli, S., Forchap, S., Dattilo, C., Adinolfi, E., Girolomoni, G., Virgilio, F. D., and Ferrari, D. (2007). Stimulation of purinergic receptors modulates chemokine expression in human keratinocytes. *Journal of Investigative Dermatology*, 127:660–667.

Pesce, G., Marcon, A., Carosso, A., Antonicelli, L., Cazzoletti, L., Ferrari, M., Fois, A. G., Marchetti, P., Olivieri, M., Pirina, P., Pocetta, G., Tassinari, R., Verlato, G., Villani, S., and Marco, R. D. (2015). Adult eczema in italy: prevalence and associations with environmental factors. *Journal of the European Academy of Dermatology and Venereology*, 29:1180–1187.

Schuler, C. F., Billi, A. C., Maverakis, E., Tsoi, L. C., and Gudjonsson, J. E. (2023). Novel insights into atopic dermatitis. *Journal of Allergy and Clinical Immunology*, 151:1145–1154.

Serio, P. (2023). Gene expression microarray merging. https://rpubs.com/Karksus/1013177.

Silverberg, J. I. and Hanifin, J. M. (2013). Adult eczema prevalence and associations with asthma and other health and demographic factors: A us population-based study. *Journal of Allergy and Clinical Immunology*, 132:1132–1138.

Skevaki, C., Ngocho, J. S., Amour, C., Schmid-Grendelmeier, P., Mmbaga, B. T., and Renz, H. (2021). Epidemiology and management of asthma and atopic dermatitis in sub-saharan africa. *Journal of Allergy and Clinical Immunology*, 148:1378–1386.

Tsai, T.-F., Rajagopalan, M., Chu, C.-Y., Encarnacion, L., Gerber, R. A., Santos-Estrella, P., Llamado, L. J. Q., and Tallman, A. M. (2019). Burden of atopic dermatitis in asia. *Journal of Dermatology*, 46:825–834.

Wang, X. and Yu, G. (2023). Drug discovery in canine pyometra disease identified by text mining and microarray data analysis. *BioMed Research International*, 2023.

Zhong, Y., Qin, K., Li, L., Liu, H., Xie, Z., and Zeng, K. (2021). Identification of immunological biomarkers of atopic dermatitis by integrated analysis to determine molecular targets for diagnosis and therapy. *International Journal of General Medicine*, 14:8193–8209.

`MLN-Subdue`: Substructure Discovery In Homogeneous Multilayer Networks

Anish Rai, Anamitra Roy, Abhishek Santra and Sharma Chakravarthy
Computer Science and Engineering Department and Information Technology Laboratory (IT Lab),
The University of Texas at Arlington, Arlington, Texas 76019, U.S.A.
{*anish.rai, axr9563, abhishek.santra*}*@mavs.uta.edu, sharmac@cse.uta.edu*

Keywords: Multilayer Networks, Substructure Discovery, Decoupling Approach, Map/Reduce Architecture.

Abstract: Substructure discovery is a well-researched problem for graphs (both simple and attributed) for knowledge discovery. Recently, multilayer networks (or MLNs) have been shown to be better suited for modeling complex datasets that have multiple entity and relationship types. However, the MLN representation brings new challenges in finding substructures due to the presence of layers, and substructure discovery methods for MLNs are currently not available.

This paper proposes a substructure discovery algorithm for homogeneous MLNs using the decoupling approach. In HoMLNs, each layer has same or a common subset of nodes but different intralayer connectivity. This algorithm has been implemented using the Map/Reduce framework to handle arbitrarily large layers and to improve the response time through distributed and parallel processing. In the decoupled approach, each layer is processed independently (without using any information from other layers) and in parallel and the substructures generated from each layer are *combined after each iteration* to generate substructures that *span layers*. The focus is on the correctness of the algorithm and resource utilization based on the number of layers. The proposed algorithm is validated through extensive experimental analysis on large real-world and synthetic graphs with diverse graph characteristics.

1 MOTIVATION

Mining has been traditionally performed on transactional data whether it is clustering, classification, or identifying frequent itemsets. For applications where there is an inherent relationship, graphs offer better representation for the modeling of data. As a result, mining techniques were extended to graph models. Graphs can also be used to model relationships among multiple object types and relationships in a variety of applications such as chemical compounds, virus propagation, electrical and road transportation networks, web analysis, etc. In particular, with graph models where each vertex corresponds to an entity and each edge corresponds to a relationship between two entities, the problem of finding frequent patterns becomes one of discovering subgraphs that occur frequently or compress the graph or forest better.

Substructure discovery (Cook and Holder, 1993) was developed as a main memory algorithm for graph models when data sizes were not very large. However, with the advent of social networks and the Internet, graph sizes have grown significantly, necessitating alternative approaches to substructure discovery.

Why Multilayer Networks (MLNs)? As graphs become larger (in terms of the number of nodes and edges) and complex (in terms of the number of entity types and relationships), modeling the data using a representation that preserves the structure and semantics of data becomes important. A model that is also amenable to efficient analysis is another issue to reckon with. From these perspectives, MLNs offer distinct advantages.

Simple graphs are unable to capture the complexity of data and its semantics although a large number of analysis algorithms exist for them. Attributed graphs can handle additional complexity with multiple edges, but lack analysis algorithms. MLNs, as a network of networks, offer separation of semantics (as individual layers) and flexibility of analysis (when decoupling approach (Santra et al., 2017b; Santra et al., 2017a) is used) using desired subsets of layers. In addition, extant simple graph analysis algorithms can be leveraged in the decoupling approach.

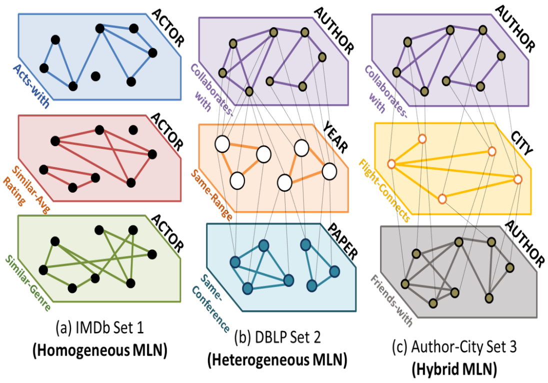
If MLNs are used for modeling, each layer in the MLN represents a different relationship, either be-

Rai, A., Roy, A., Santra, A. and Chakravarthy, S.
MLN-Subdue: Substructure Discovery In Homogeneous Multilayer Networks.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 36-47
ISBN: 978-989-758-716-0; ISSN: 2184-3228

tween the *same* type of entities within a layer (intralayer edges), or across layers between *different* types of entities (interlayer edges). The advantages of modeling data using MLNs are discussed in (Boccaletti et al., 2014; Santra et al., 2017b; Kivelä et al., 2013). However, with the use of MLNs, the challenge is to extend graph analysis algorithms, be it community, centrality, or substructure detection, to the new representation.

Depending on the types of entities, multiple layers can be defined for the same (or a subset of) entity type or different entity types. MLNs can be of three different types: Homogeneous, Heterogeneous, and Hybrid. Homogeneous Multilayer Networks (HoMLNs) are used to model multiple distinct relationships existing between the *same type of entities*. Each set of *intralayer* edges represent one particular type of relationship, and the *interlayer* edge sets are implicit, as the same set of nodes are present in every layer. For example, the same set of actors can be connected based on co-acting, similar average rating, and similar movie genres they work in forming three different layers as shown in Figure 1 (a). Relationships among different types of entities are modeled through Heterogeneous Multilayer Networks (or HeMLNs), as shown in Figure 1 (b). The *interlayer* edges here are explicitly represented to demonstrate the relationship across layers. For example, there can be edges between the author layer and paper layer, if an author has written that paper. Finally, for data that includes multiple relationships within and across different types of entity sets, a combination of homogeneous and heterogeneous multilayer networks can be used, called Hybrid Multilayer Networks (or HyMLNs), as shown in Figure 1 (c).



(a) IMDb Set 1 (Homogeneous MLN)

(b) DBLP Set 2 (Heterogeneous MLN)

(c) Author-City Set 3 (Hybrid MLN)

Figure 1: Examples of MLN Types.

Why Map/Reduce? We have used the Map/Reduce paradigm as an example of the distributed and parallel processing approach. Without loss of generality, any other paradigm (e.g., Spark) can be used in its stead without modifying the overall approach.

Problem Statement: The problem addressed in this paper is finding interesting and frequent substructures in a given Homogeneous Multilayer Network (HoMLN) using the decoupling approach proposed in (Santra et al., 2017a; Santra et al., 2017b; Santra et al., 2022) to leverage its advantages. This amounts to not converting the MLN into a single graph, but still getting the same result as if the MLN was processed as a single graph.

The main challenge here is to use the substructures generated for each layer **independently** and in parallel to compose them to compute the *missing* substructures that span multiple layers correctly and efficiently. To the best of our knowledge, there are no MLN substructure discovery algorithms. However, the rationale for using the decoupling approach is that existing algorithms from the literature(Cook and Holder, 1993; Das and Chakravarthy, 2015) can be used effectively for **each layer**. When these algorithms are used to find substructures in each layer independently, many substructures that span layers will be missing as shown in Figure 2.

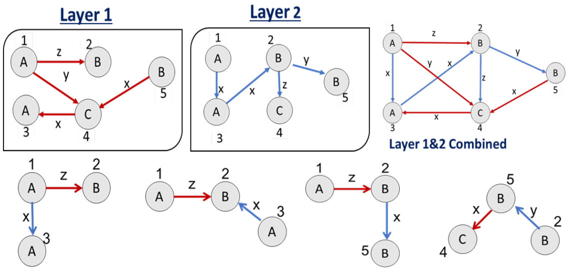


Figure 2: Substructures that span layers - only combined graph can generate them (color coded).

Approaches for MLN Substructure Discovery: MLNs can be processed for various types of analyses (community detection, substructure discovery etc.) Alternative approaches are listed below with a brief explanation.

1. Traditional Aggregation Approach: In this approach, layers of a MLN are conflated into a single graph. Boolean AND or OR aggregation can be used to reduce a HoMLN to a simple graph and find substructures correctly for a given multilayer network. The aggregation process can be costly (depending on the number of layers) and the resulting graph can be large (for OR composition). As a result, substructure discovery time can be significantly greater than that of a single layer. This approach also restricts parallelization as each layer cannot be processed in parallel. The loss in MLN structure, due to aggregation, also makes it difficult to drill-down results without maintaining extensive mapping. Finally, for processing a subset of layers, separate aggregation is required – making the approach less flexible and inefficient.

2. Decoupling-Based Approach: It is a "divide and conquer" approach for analyzing multilayer networks. The goal is to find substructures in each layer independently and then use a *separate* **composition function** to combine the results to generate what is missing for that analysis (e.g., community, centrality, substructure, motif, etc.) The decoupling-based approach used in this paper to address the substructure discovery problem is described along with a figure (Figure 5) in Section 4.

3. Holistic Approach: In this approach, neither the MLN is aggregated nor the decoupling approach is used. An algorithm is developed from scratch (as available graph analysis algorithms cannot be used as in the previous two approaches), keeping the structure and semantics intact. However, a new algorithm need to be developed for each analysis making this approach complex as the algorithm needs to traverse back and forth across layers. This approach has been used for coherent clusering in (Boden et al., 2012)

We use the decoupling-based approach in this paper as it is efficient and extant analysis algorithms (there are several of them depending on the analysis) can be used. The main challenge to be addressed is the development of the composition function and establishing its correctness (soundness and completeness.) Also, evaluating the efficiency of this approach as compared to the aggregation approach (used as GT or ground truth for validation.)

The contributions of this paper are:

- Adapting the **decoupling approach** for substructure discovery
- **Developing Map/Reduce approach** for substructure discovery
- **Composition algorithm** for HoMLN substructure discovery
- **Establishing the empirical correctness** of the composition algorithm
- **Extensive experimental analysis with diverse synthetic and real-world datasets**

This paper is organized as follows. Section 2 discusses related work. Section 3 discusses the preliminaries for substructure discovery using Map/Reduce. Section 4 details the adaptation of the decoupling approach for iteration-based substructure discovery. Section 5 discusses the composition algorithm and its correctness. Section 6 discusses the Map/Reduce approach for distributed and parallel computation on MLN. Section 7 provides experimental analysis. Conclusions are in Section 8.

2 RELATED WORK

SUBDUE (Cook and Holder, 1993; Ketkar et al., 2005) was developed as a main-memory substructure discovery algorithm. It performs a computationally constrained *beam* search where substructures of increasing size are generated iteratively and evaluated using the Minimum Description Length (or MDL) (Rao and Lu, 1992) metric. The algorithm begins with all substructures of size one (i.e., an edge), and in each iteration expands the instances by one neighboring edge in all possible ways. After each iteration, the top *beam* (parameter specified) substructures are carried over to the next iteration. Best substructures are output based on the size and other parameters specified.

Due to the limitations of the main-memory approach, the disk-based approach (Wang et al., 2005) stores the data on disks and stages chunks of data to memory as needed. The graph is indexed to speed up retrieval. However, this approach needs to marshal data between the disk and main memory buffer, and its performance can be very sensitive to buffer size, replacement policies, hit ratios, etc. To overcome the pitfalls of the disk-based approach, a database management system (or DBMS) was used to store the graph and SQL for substructure discovery(Padmanabhan and Chakravarthy, 2009). This takes advantage of the buffer management and optimization provided by the DBMS. Although scalability was achieved to graphs of over a million nodes and edges, use of self-joins on large relations for substructure expansion seems to have made it difficult to go beyond due to computation resulting in unacceptable response time. Also, it appears that the removal of duplicate substructures required sorting columns in row-based Relational DBMS, making it expensive in terms of the number of joins needed.

As the graph sizes grew further with the advent of social networks and the Internet, graphs had to be partitioned to deal with the increasing sizes. As a result scalable parallel computing algorithms had to be developed. Map/Reduce (Das and Chakravarthy, 2015; Das and Chakravarthy, 2018) and other architectures were used to address the problem of substructure discovery on a large graph by dividing the graph into smaller partitions and then combining the results across partitions. In addition to substructure discovery, partitioning of graphs has been explored in other research as well (Yang et al., 2012).

With multilayer networks being used for modeling large complex datasets, there is a need for substructure discovery algorithms on MLNs. A clustering algorithm (Boden et al., 2012) for finding clusters in a multilayer graph has been proposed using the holistic approach described earlier. Similarly, another algorithm (Liu and Wong, 2008) has been proposed to find quasi-cliques to find all one-dimensional clusters in a single layer, which are then used to find multi-dimensional clusters. The focus of this work is to find clusters of vertices that are densely connected by edges with similar edge labels in a subset of graph layers.

This paper differs from the above in that we are proposing a decoupling-based algorithm for substructure discovery that gives the same results as the ground truth. It is also efficient as compared with the algorithm used for GT. Further, it uses Map/Reduce to process each layer and for composition providing better scalability than extant approaches.

3 TERMINOLOGY USED

Graphs are input as text files and this section indicates input formats for `MLN-Subdue` as well as some terminology needed for understanding the paper.

Input Graph Representation: For substructure discovery, labeled graphs are used where vertex and edge labels are not assumed to be unique, but all vertex IDs are unique. The input graph is represented as an unordered list of 1-edge substructures where each edge is represented as a 5-element tuple <*edge label, source vertex id, source vertex label, destination vertex id, destination vertex label*>. The input graph is stored in an ASCII file with a 1-edge substructure in each line. If needed, graphs in other formats are converted to this format.

Adjacency List: Adjacency list of a vertex ID is the set of edges that are incident (both outgoing and incoming) on that vertex ID. The adjacency list is used to expand a substructure from each vertex ID in that substructure using an edge from the adjacency list. Each 1-edge expansion becomes a separate substructure. This allows an n-edge substructure to be expanded into a number of (n+1)-edge substructures in an iteration.

Substructure Expansion: Starting from 1-edge, substructures of increasing size are generated systematically in each iteration using the adjacency list. As the goal is to discover *interesting* substructures of any size, systematic graph expansion is critical to the process. Expansion is done on each substructure *independently* as indicated above. Expansion is unconstrained, i.e., each substructure independently grows into a number of larger substructures in each iteration. This *independent expansion* leads to the generation of duplicate substructures which must be removed to ensure correctness. A substructure is represented as a (lexicographically) ordered list of edges.

Canonical Instances for Duplicate Elimination: Lexicographic ordering of edges in a substructure is used to identify duplicates. Each edge in a substructure is ordered based on edge label, then source vertex label, then destination vertex label, and finally source and destination vertex IDs. If any of the values match, the comparison moves forward to the next component, else the ordering is performed. A substructure can be uniquely represented using the lexicographic order of 1-edge components. This is called a canonical k-edge instance. Intuitively, two duplicate k-edge substructures must have the same ordering of labels and vertex IDs when converted to canonical k-edge instance[1]. Figure 3 shows an example of duplicate substructures generated by two **different** substructure instances during independent expansion and how they are detected as duplicates using their canonical instances.

Canonical Substructures for Graph Isomorphism: Isomorphs in a graph have the same graph structure in terms of vertex and edge labels as well as connectivity, but differ in vertex IDs. In contrast, duplicates have the same vertex IDs. After duplicate elimination, we need to identify isomorphs to count their occurrences. We need to convert canonical instances of substructures to *canonical substructures* using relative ordering of vertex IDs. Intuitively, in the canonical form, two isomorphic substructures have the **same relative ordering** of vertex numbers. To identify isomorphs, the canonical instance is converted into a canonical substructure. This is done by replacing each vertex ID with their relative positions in the instance starting from 1. The inclusion of these relative positions is critical for differentiating the connectivity of the instances. Figure 4 shows an example of how canonical substructure is created from the canonical instance. It can be seen that the isomorphs have different canonical instances. Using the above technique, the relative positioning of vertex Ids (2, 5, 4) for the canonical instance 1 and (7, 10, 9) for the canonical instance 2 are converted to (1, 2, 3). Hence we can

[1] Substructures and substructures instances are used interchangeably when the meaning is clear from the context. However, substructure instances are converted into relative ordering of vertex IDs, termed canonical substructures, which are used for detecting substructure isomorphs.

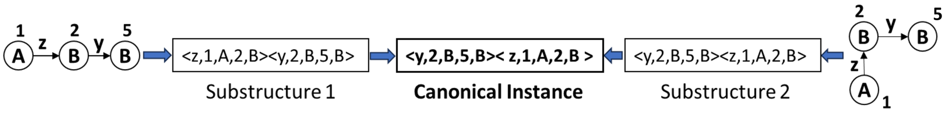


Figure 3: Duplicate substructure identification using canonical instances.

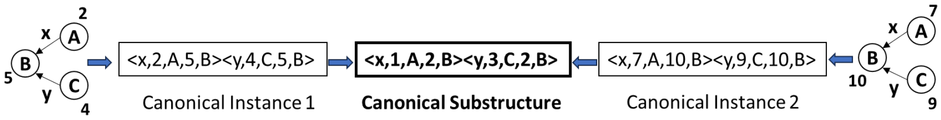


Figure 4: Graph isomorphism and canonical substructures.

identify isomorphs using canonical substructures.

Metrics for Ranking: Many metrics are used to rank substructures based on isomorphs. Frequency of isomorphic substructures is one such metric (higher the frequency, the better the substructure.) Another widely used information-theoretic metric is the Minimum Description Length (or MDL (Rao and Lu, 1992)). MDL calculates the importance of a substructure on how well it compresses the entire graph/forest. A substructure that compresses the graph better is considered an *interesting and repetitive* substructure. When MDL is used, an MRN (Most Restrictive Node)(Bringmann and Nijssen, 2008; Elseidy et al., 2014) metric is used to count non-overlapping instances. But overlapping instances have also been used to compute frequency and MDL for each substructure.

4 DECOUPLING APPROACH

Multilayer networks consist of multiple layers of simple graphs where each layer represents a relationship between entities in that graph. However, most algorithms convert a MLN (or a subset of it) into a simple graph using aggregation (Domenico et al., 2014) and/or projection techniques (Berenstein et al., 2016) to use extant algorithms. However, this leads to loss of structure, semantics, and information from the final analysis results (Kivelä et al., 2013; De Domenico et al., 2014). For the other end, existing single graph algorithms cannot be directly used for the holistic approach described earlier. In this paper, the decoupling-based approach has been used which preserves the structure and semantics of MLNs (Santra et al., 2017a; Pavel et al., 2023) while performing analysis on complex datasets without losing any information. It is also shown to be efficient (Santra et al., 2017a).

The network decoupling approach has been illustrated with respect to substructure discovery in HoMLNs (focus of the paper) in Figure 5, where each layer has the same set of nodes but different edge connectivity. It consists of two functions: analysis (Ψ) and composition (Θ). Using the analysis function, each layer in the network is analyzed independently (and in parallel) to obtain the layer substructures. Then, the partial results from any two (or more) layers are combined and processed by the composition function to produce substructures that span participating layers. This is done for each iteration as substructure discovery is an iterative algorithm. This composition can be binary or n-ary. If binary, it can be repetitively applied to n layers using previously generated results. This approach allows parallel analysis of each layer to improve efficiency (Santra et al., 2017a). Further, due to the layer-wise analysis, each graph is likely to be small, which requires less memory for computing layer-wise results. Each layer is also analyzed **only once**, and the existing single graph algorithms can be used for that. The results obtained are then used by the composition function. In addition, this approach is application-independent.

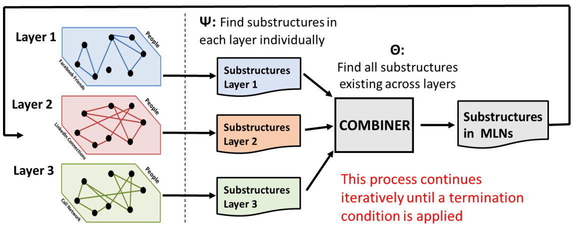


Figure 5: Substructure Discovery: Decoupling Approach.

In this approach, the major challenge is to develop the composition algorithm (Ψ) which is sound and complete in generating missing substructures that span layers. It is known from earlier work that substructure generation in each layer is sound and complete.

5 ALGORITHM DESIGN

Substructure discovery in single (simple and attributed) graphs is an iterative process, where inter-

Algorithm 1 : **MLN-Subdue**: Substructure Discovery Algorithm for Homogeneous Multilayer Networks.

```
Require:
    Input: Substructures of size $k$ for $k^{th}$ iteration
    Output: Top beam substructures (intralayer & interlayer) of size $k+1$
 1: Load Adjacency list for each layer
 2: for each substructure of size $k$ in MLN:
 3:     Expand each k-edge substructure by one edge in all directions in each layer
 4:     Eliminate Duplicates using canonical representation in each layer
 5: end for
 6: for each expanded $k+1$ edge substructure:
 7:     Group all the expanded substructures from each layer based on vertex ID
 8:     Apply Combine-MLN recursive function to form $k+1$ edge interlayer substructures from k+1 edge intralayer substructures
 9:     Eliminate duplicates generated during combination
10: end for
11: for all the canonical instances in the MLN:
12:     Count the frequency of all substructures using isomorphism
13: end for
14: Apply metric Frequency or MDL
15: Apply beam heuristic to determine top-beam substructures and send their instances to be used the next iteration // Specify beam as required
16: Increment k by 1 for next iteration
17: Goto Step 1 for the next iteration
```

esting (k+1)-edge substructures are generated in the k^{th} iteration after a multi-step process – independent substructure instance expansion, conversion to canonical form, duplicate elimination, substructure counting & metric evaluation, ranking based on graph isomorphism, and finally, retaining top *beam* substructures to be used for the next iteration. For using the divide-and-conquer-based decoupling approach, the main task during the k^{th} iteration is to figure out how to systematically use the *beam k-edge* substructures and composed substructures (from the previous iteration) to generate next *beam* (k+1)-edge substructures correctly and completely. Thus, in case of MLNs, the challenge is to perform the substructure discovery (from expansion to substructure ranking) synergistically using what is generated in the layer-wise analysis phase and what is composed in the previous iteration.

Algorithm 1 presents the composition algorithm to discover interesting substructures in HoMLNs. The major steps are discussed below:

Expansion (Layer-wise): In the k^{th} iteration, all instances of the *beam k*-edge substructures, generated in the previous iteration, are used (for k = 1, all edges in the layer are used.) In each layer, using the adjacency list for that layer, each substructure instance is expanded *independently* by adding one incident edge (both in and out) to generate as many $(k+1)$-edge substructure instances using the adjacency list (**Lines 1 & 2** of Algorithm 1). However, this unconstrained expansion generates *local (layer-wise)* duplicates, which are identified using canonical ordering (as outlined in Section 3) and are eliminated (**Line 4**).

Composing Layer-wise Substructures: This step generates substructure instances similar to the expansion in layers, but uses a substructure from one layer and edges from a *different* layer which are termed *composed interlayer* substructures. To achieve this, first the expanded substructure instances generated from each layer are *grouped* based on a shared vertex ID (**Line 7**) as vertex IDs are the same in all layers of HoMLNs. This brings together the substructure instances from all layers that have a shared vertex ID. Then, a recursive call (***Combine-MLN***) is made (with two parameters: **set of layers**, and **size of the substructure**) to combine the intralayer substructures from different layers sharing a common vertex ID, to generate $(k+1)$-edge substructures that span multiple layers (**Line 8**). Here, the recursive call performs a systematic exploration of all combination possibilities to generate a $(k+1)$-edge substructure from m layers using the layer-wise substructures of size (k+1). This is applied on all vertices.

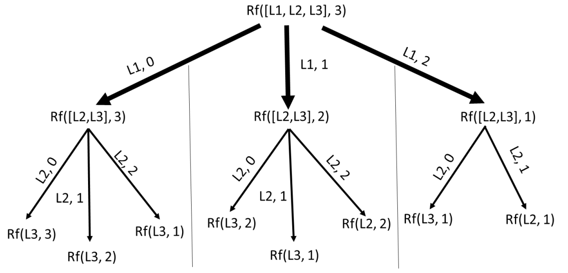


Figure 6: Combination possibilities of generating a 3-edge interlayer substructure.

Figure 6 shows an example of the combinatorial possibilities applied by the ***Combine-MLN*** recursive function, where the set of layers is [L1, L2, L3] and the substructure size is 3. For example, one possibility is to choose 2 edges from L1 and 1 edge from either L2 or L3 (the rightmost subtree) or 1 edge from L1, one edge from L2, and 1 edge from L3 (middle subtree.) Also, this composition stage ensures that only connected composed substructures are generated.

Figure 7 illustrates the working of ***Combine-MLN*** on a 2-layer MLN. For each layer we show all the 2-edge substructures. When we apply the ***Combine-MLN*** function on each Vertex id of the graph as shown above, we find all the substructures of size

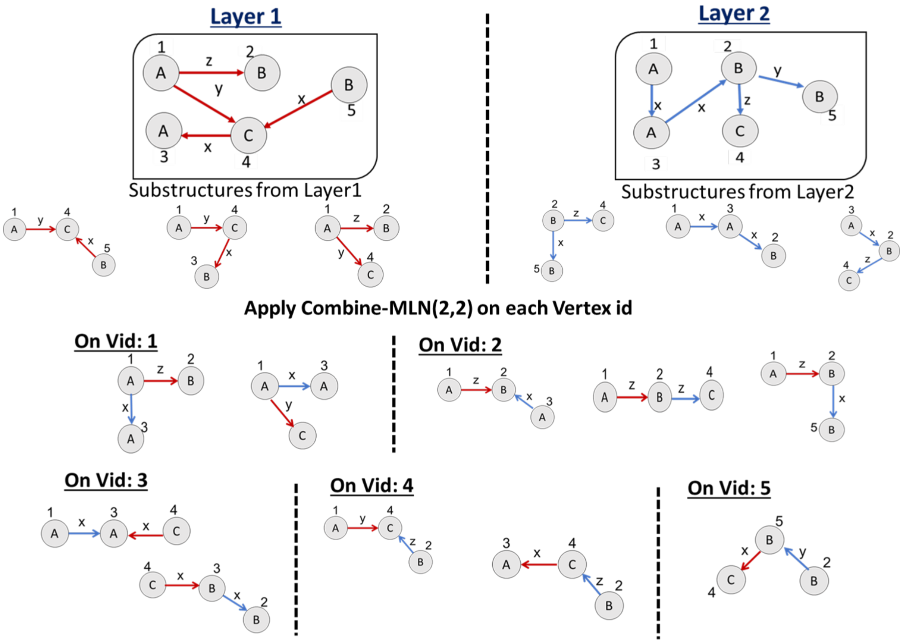

Layer 1
Layer 2
Substructures from Layer1
Substructures from Layer2
Apply Combine-MLN(2,2) on each Vertex id
On Vid: 1
On Vid: 2
On Vid: 3
On Vid: 4
On Vid: 5

Figure 7: Generating composed interlayer substructures by combining intralayer substructures on shared vertex id.

2, which exist across layers. The parameters of Combine-MLN function here is (2,2), which means there are 2 layers and the required size of the substructure is 2.

Duplicate Elimination: The previous step leads to the generation of duplicate instances (e.g., from figure 6, the center branch would be generated while expanding on each of the three layers, L1, L2, and L3.) As the composed substructures are cast into the canonical form, duplicates are identified and removed (**Line 9**).

Frequency Counting: Exact isomorphs are used to detect identical substructures, as two isomorphic substructures have the same relative ordering of the vertex IDs and have same vertex and edge labels. Canonical instances follow the lexicographic ordering, hence it is easy to generate *k*-edge canonical substructures using the relative ordering of unique vertex ID in the order of their appearance in the canonical instance. All the instances are grouped based on their canonical form and their frequency is counted (**Line 12**).

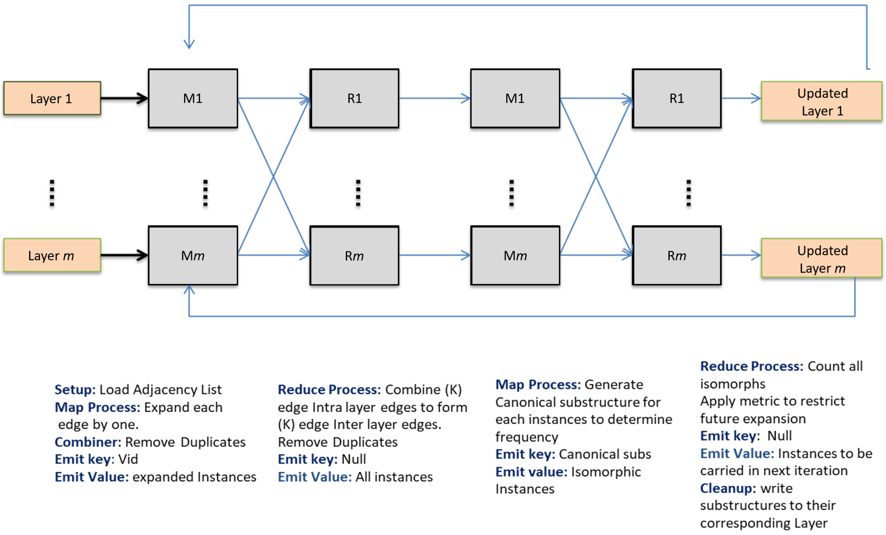
Application of Metric: To restrict the future expansion to high quality substructures, a metric – either MDL or frequency – is used to determine the importance of a particular substructure (**Line 14**). The *beam* parameter (user specified with a default) is used as a heuristic to ensure only the top *beam* substructures based on the metric score will be passed on to the next iteration, thereby restricting the expansion of less important substructures (**Line 15**).

Using the proposed algorithm, top substructures of size *k+1* that exist within and across layers in the k^{th} iteration are generated. Algorithm 1 is applied iteratively to find substructures of the desired size.

6 MAP/REDUCE IMPLEMENTATION

The different components of the proposed Algorithm 1 (layer-wise substructure expansion and duplicate elimination followed by composition, and duplicate elimination again in each iteration) have been implemented using an iterative two-chained Map/Reduce architecture. In the first Map/Reduce job, the mapper performs the expansion of substructures and duplicate elimination in each layer, and the reducer performs the composition to generate substructures that span layers and removes the duplicates from the composed substructures. In the second job, the mapper converts *all* substructure instances into canonical isomorphic instances to count frequency and emits isomorphs as key. The reducer applies the metric and outputs the top-*beam* substructures (intralayer and spanning layers) to be used as candidates for expansion in the next iteration. Figure 8 shows the overall workflow of substructure discovery in a HoMLN using the Map/Reduce framework.

Expansion by Mapper 1: Substructure instances



Setup: Load Adjacency List
Map Process: Expand each edge by one.
Combiner: Remove Duplicates
Emit key: Vid
Emit Value: expanded Instances

Reduce Process: Combine (K) edge Intra layer edges to form (K) edge Inter layer edges. Remove Duplicates
Emit key: Null
Emit Value: All instances

Map Process: Generate Canonical substructure for each instances to determine frequency
Emit key: Canonical subs
Emit value: Isomorphic Instances

Reduce Process: Count all isomorphs
Apply metric to restrict future expansion
Emit key: Null
Emit Value: Instances to be carried in next iteration
Cleanup: write substructures to their corresponding Layer

Figure 8: Map/Reduce workflow of **MLN-Subdue** (including composition).

Table 1: Dataset description.

Purpose	Dataset	#Nodes	#Edges
Correctness	Synthetic (SUBGEN)	10K	20K
Correctness	Synthetic (SUBGEN)	100K	800K
Scalability	LiveJournal	3.9M	34.8M
Scalability	Orkut	3.87M	114.8M
Varying Density	Synthetic (SUBGEN)	2K	1M, 1.9M, 2.9M, 3.9M

with their respective layer number are read as mapper input, one at a time. The adjacency list for the layer is loaded using the setup function. For the k^{th} iteration, the input is a k-edge canonical instance. Each instance is expanded by one edge at a time using the adjacency list. The mapper emits the vertex ID as key, and expanded instances as values. As the expansion process is unconstrained, duplicates are generated, which are removed using a combiner. In this version, since each layer is processed by a mapper (even as multiple map tasks based on the block size), all duplicates can be eliminated by the combiner.

Composition in Reducer 1: Each reducer receives a list of expanded substructure instances as values, grouped on the vertex ID as the key. The recursive function (Combine-MLN) is used to combine all $(k+1)$-edge substructures from the mapper outputs to generate $(k+1)$-edge spanning substructures using edges from multiple layers. All substructures, both intralayer and spanning layers, are then emitted to the next Map/Reduce job as input to generate canonical substructures to determine graph isomorphism. The key is null, and the value is the substructure instance.

Identifying Isomorphs in Mapper 2: Creating canonical substructures from instances requires a hash table to identify the relative positioning of each vertex. The mapper receives all the substructure instances as input, and canonical substructure is generated for every instance. The mapper emits the canonical substructure as the key and the corresponding substructure instance as the value.

Ranking and Emitting top-*beam* Instances in Reducer 2: The reducer receives substructure instances across mappers grouped on canonical substructure. The *beam* value is used to rank the best *beam* substructures in a hash map. This is done by calculating the MDL value for each canonical substructure, and storing the top *beam* substructures with the highest MDL values, emitting only their respective instances in order to restrict future expansion to high-ranking substructures in the next iteration. The reducer emits the $(k+1)$-edge substructure instances and layer IDs as the values in the k^{th} iteration, which are then fed into Mapper 1 of the next iteration as input.

7 EXPERIMENTAL ANALYSIS

Experimental Setup: All experiments have been performed using Java with Hadoop on Comet cluster at SDSC (San Diego Supercomputer Center). The

Comet cluster has 1944 nodes and each node has 24 cores (built on two 12-core Intel Xeon E5 2.5 GHz processors) with 128 GB memory, and 320GB SSD for local scratch space.

Dataset Description: Experiments were done on several real-world and synthetic datasets (as shown in Table 1) of varying sizes to establish the correctness, speedup, and scalability of the approach. Subgen[2], an artificial graph generator was used to generate synthetic graphs as it allows the embedding of substructures with user-defined frequency in a larger single graph. To generate HoMLN datasets from each base single graph, edges were randomly distributed across multiple layers among the same set of nodes.

Empirical Correctness: For a given HoMLN with multiple layers, the ground truth is generated by first aggregating the MLN into a single graph by taking the *union* of edges (Boolean OR) and then executing SUBDUE(Ketkar et al., 2005). Both the proposed algorithm and SUBDUE for different HoMLNs generated the same set of substructures with correct frequency, thus *establishing the correctness of the approach empirically*.

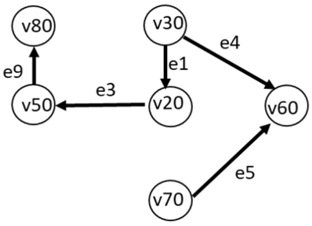


Figure 9: Example 5-edge Embedded Substructure.

However, SUBDUE being a main memory approach failed to execute on large graphs with more than 100K vertices & 800K edges. So, to verify the correctness on large graphs, synthetic graphs were generated having substructures embedded with a user-defined frequency, the goal being to *find the same substructures with the same frequency*. A 5-edge embedded substructure, shown in figure 9, was embedded with a frequency of 1000 in a graph of size 100K vertices & 800K edges. The exact substructure was found with the same frequency (of 1000) using the proposed algorithm, which empirically validates the correctness of the proposed algorithm for this dataset.

Several such experiments were conducted with different embedded graph sizes and frequencies.

Effect of Layer Generation Schemes: This set of experiments is performed using the synthetic dataset of size 400K vertices & 1.2 million edges with embedded substructures. Two partitioning schemes were used: random and edge-based, to verify the correctness of the algorithm and its effect on response time. Random scheme partitions a graph into l layers by distributing edges randomly. Nodes are same in all l layers. Edge-based partitioning, on the other hand, creates layers containing all edges having the same edge label. Multiple edge labels can be in a layer in edge-based partitioning.

Table 2: Edge distribution for different layer generation schemes.

Layers	Random	Edge-Based
Layer 1	399903	351360
Layer 2	399730	478441
Layer 3	400637	370469

Table 2 shows the edge distributions for both partitioning schemes. Notice that the distribution remains even for random partitioning, but becomes skewed for edge-based as there can be edge labels with higher frequency going into a single layer, making it uneven. Figure 10 shows the total time taken by both of the partitioning schemes. There is no substantial difference in the total response time as it more or less remains the same. So, the numbers are drilled-down into and the Map and Reduce times are inspected to understand them clearly.

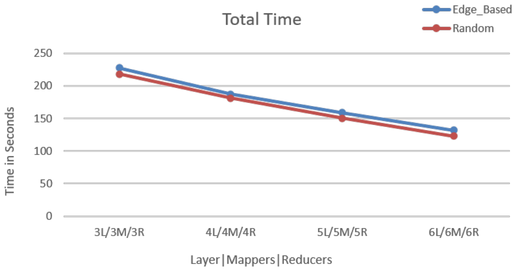


Figure 10: Total response time for partitioning schemes.

As seen in Figure 11, the total map time for edge-based partitioning is significantly higher as compared to the total map time for random partitioning. The reason being, as the edge distribution is skewed, some mappers end up processing more data for edge-based partitioning, contributing to more computation time. On the other hand, total time taken by the reducers does not change as the data processed by each reducer remains the same, because all the substructures in the reducer are grouped based on their vertex IDs and edge labels have no effect on them.

The same embedded substructures were found for both partitioning schemes, indicating that the algorithm is not affected by the connectivity of the graph.

Scalability of Approach: Without altering the graph size, an increase in the number of processors is typi-

[2] https://ailab.wsu.edu/subdue/

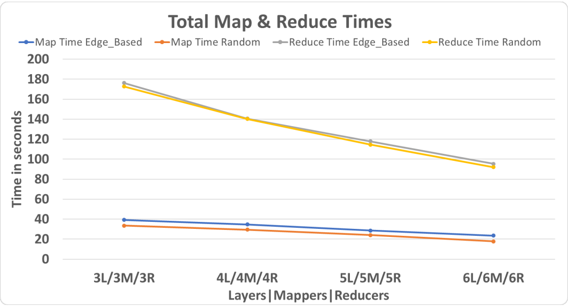


Figure 11: Map & Reduce times using multiple partitioning schemes.

cally beneficial for mining. So, this set of experiments was performed on LiveJournal (Leskovec and Krevl, 2014a) and Orkut (Leskovec and Krevl, 2014b) data to determine the speedup and effectiveness of the algorithm as the number of processors was increased. This has been showcased using 3 scenarios:

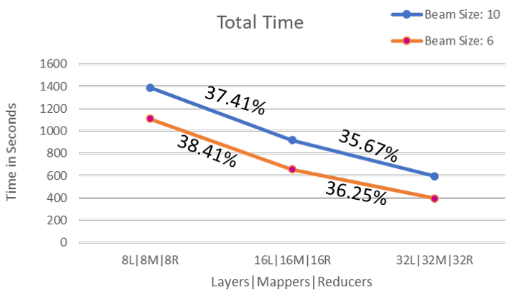


Figure 12: Speedup achieved on LiveJournal data.

Scenario 1: **Same number of mapper and reducer nodes as layers:** For this scenario, all the layers are processed in parallel with the same number of mapper and reducer nodes. Results in Figure 12 show a speedup of over 35% for the LiveJournal dataset when the number of layers, mappers, & reducers are increased from 8 to 16, and again from 16 to 32. As the same dataset is partitioned into more layers and processed by greater number of processors, it leads to smaller partitions and less computation in each processor. This reduction in computation cost contributes to the speedup achieved. Moreover, as the number of layers and reducers are doubled, the data received by each reducer node gets halved, but the mining cost in the 1st reduce job also increases, as the height of the tree grows with the increase in number of layers (as there are composed substructures that span more layers and the **number of combinations** for the recursive procedure ***Compose-MLN*** increases), leading to a higher number of possible interlayer combinations.

Figure 12 also shows that the total time taken when *beam* size is set to 10 is more as compared to *beam* size 6. This is because *as the beam size increases, the number of substructures carried forward in each iteration increases (illustrated in Table 3)*, resulting in more data being processed by each processor.

Table 3: Number of substructures generated in each iteration with *beam* 10 and 6 for the LiveJournal data.

Iterations	Beam 10	Beam 6
1	2836928	2836928
2	3929240	3172184
3	4593864	3628496
4	5249968	3849008
5	5346152	4004984
6	5716984	4128496

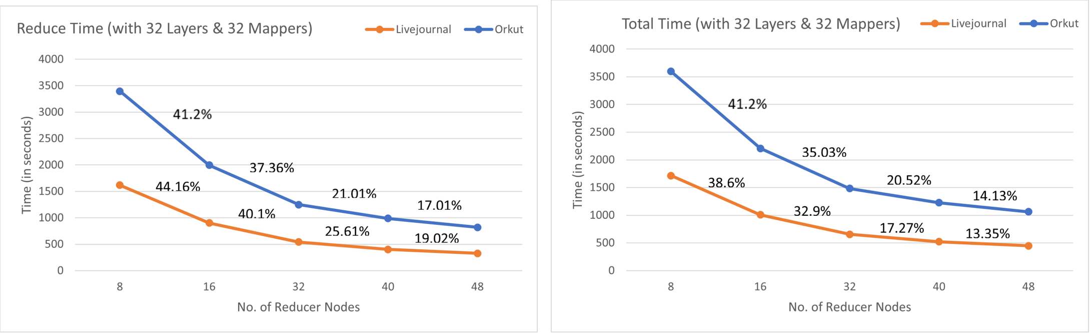
Orkut and LiveJournal datasets were used for the next two scenarios with 32 layers (Figures 13a, 13b.) The purpose is to understand the importance of mappers vs. reducers with respect to the work/computation done during the algorithm.

Scenario 2: **Changing *only* the number of reducer nodes:** In this scenario, the number of mapper nodes is fixed at 32, and the number of reducer nodes is varied to see the effect of reducers on response time.

Figure 13a initially shows a high percentage of speedup for both LiveJournal and Orkut in the reduce phase when the number of reducer nodes is increased from 8 to 16 to 32. But after that, the speedup starts to plateau, with the percentage improvement reducing significantly when going from 32 to 40 and then to 48 nodes. So, *merely adding more reducer nodes beyond a certain number does not keep improving speedup*, as the overhead (e.g., reducer setup & cleanup) increases. Orkut takes more absolute time than LiveJournal, as it has more than 3 times the number of edges for a similar number of vertices, leading to more expansion and combination possibilities. In contrast to the reduce time, total time is shown in Figure 13b. This speedup is similar to the reduce time (except in absolute values) indicating that all the speedup comes from the reducers. This matches the work done in the reducers as explained in Section 6 and validates the need for more reducers. Remember, composition is happening in the reducer of job 1 and counting all isomorphs and metric computation is happening in the reducer of job 2.

Scenario 2 clearly indicates that the mappers take less time overall as compared to the reducers. Hence, **distributing reduce computation among more reducers has a significant effect on the speedup** which is exemplified in scenario 3 below.

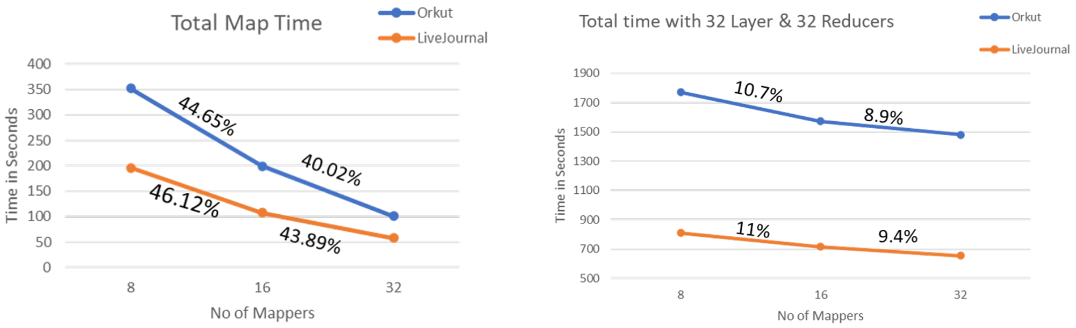
Scenario 3: **Changing *only* the number of mapper nodes:** Here, the number of reducer nodes is fixed at 32, and the number of mapper nodes is varied to inspect the effect of mappers on response time.



(a) Effect on Reduce Phase Time.

(b) Effect on Overall Map/Reduce Time.

Figure 13: Speedup comparison by varying only the number of reducer nodes.



(a) Effect on Map Phase Time.

(b) Effect on Overall Map/Reduce Time.

Figure 14: Speedup comparison on varying only the number of mapper nodes.

In Figure 14a, significant speedup is seen in the **map time** as the number of mapper nodes is increased from 8 to 16 to 32. But the total time shown in Figure 14b is distinctly different from the one we see in Figure 13b. **Total time speedup** has reduced significantly due to much smaller portion of the computation being distributed. **This, again, clearly indicates that there is not much computation in the mappers to effectively distribute them to improve the overall Map/Reduce performance.** In both jobs in the chain, the reducers are doing heavy lifting in this architecture. *From Scenarios 2 and 3, it can be inferred that for this algorithm, prioritizing an increase in the number of reducers over mappers is more beneficial, as the reduce phase has a greater effect on total execution time.*

Effect of Graph Density: Connectivity of graphs also influences the performance of substructure discovery due to large number of substructures generated during expansion. We categorize graphs as dense to sparse, where the number of vertices is fixed but the number of edges vary along the spectrum ranging from a completely connected graph to a very sparsely connected graph. This experiment was performed on a graph with 2K vertices, but varying densities from 25% (1M edges) to 100% (3.9M edges) on 4 layers using 4 mapper nodes and 4 reducer nodes to see the effect of connectivity of graphs on response time.

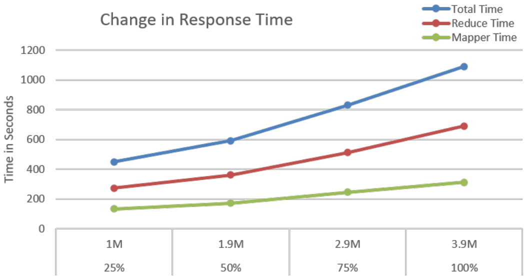


Figure 15: Effect of graph density on response time.

The results in Figure 15 show that with *dense graphs*, where each vertex is connected to more vertices on average, *the computation cost increases as there are more expansions which leads to more map time and more combinations of intralayer edges in the reducer contributing to more reduce time. This further confirms that more time is spent in the reducer in this algorithm as compared to the mapper.*

8 CONCLUSIONS

This paper proposes a scalable substructure discovery algorithm for HoMLNs using the decoupling-based strategy. A generic Map/Reduce algorithm was introduced for the parallel processing of layers and then composing the results, again, using a Map/Reduce framework. The basic components of substructure discovery - substructure expansion, *combining substructures from each layer to generate substructures spanning layers (composition function)*, duplicate elimination, and counting isomorphic substructures - were incorporated into the Map/Reduce framework. Empirical correctness was established. Extensive experimental analysis was performed on diverse synthetic and real-world graph datasets.

ACKNOWLEDGMENTS

This work was supported by NSF awards #1955798 and #1916084.

REFERENCES

Berenstein, A., Magarinos, M. P., Chernomoretz, A., and Aguero, F. (2016). A multilayer network approach for guiding drug repositioning in neglected diseases. *PLOS*.

Boccaletti, S., Bianconi, G., Criado, R., del Genio, C., Gómez-Gardeñes, J., Romance, M., Sendiña-Nadal, I., Wang, Z., and Zanin, M. (2014). The structure and dynamics of multilayer networks. *Physics Reports*.

Boden, B., Günnemann, S., Hoffmann, H., and Seidl, T. (2012). Mining coherent subgraphs in multi-layer graphs with edge labels. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1258–1266. ACM.

Bringmann, B. and Nijssen, S. (2008). What is frequent in a single graph? In *Pacific-Asia Conf. on Knowl. Disc. and Data Mining (PAKDD)*, pages 858–863. Springer.

Cook, D. J. and Holder, L. B. (1993). Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1:231–255.

Das, S. and Chakravarthy, S. (2015). Partition and conquer: Map/reduce way of substructure discovery. In *International Conference on Big Data Analytics and Knowledge Discovery*, pages 365–378. Springer.

Das, S. and Chakravarthy, S. (2018). Duplicate reduction in graph mining: Approaches, analysis, and evaluation. *IEEE Trans. Knowl. Data Eng.*, 30(8):1454–1466.

De Domenico, M., Solé-Ribalta, A., Gómez, S., and Arenas, A. (2014). Navigability of interconnected networks under random failures. *Proc. of Ntl. Acad. of Sciences*.

Domenico, M. D., Nicosia, V., Arenas, A., and Latora, V. (2014). Layer aggregation and reducibility of multilayer interconnected networks. *CoRR*, abs/1405.0425.

Elseidy, M., Abdelhamid, E., Skiadopoulos, S., and Kalnis, P. (2014). Grami: Frequent subgraph and pattern mining in a single large graph. *Proceedings of the VLDB Endowment*, 7(7):517–528.

Ketkar, N. S., Holder, L. B., and Cook, D. J. (2005). Subdue: Compression-based frequent pattern discovery in graph data. In *Proceedings of the 1st Int. Workshop on open source data mining: frequent pattern mining implementations*, pages 71–76.

Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., and Porter, M. A. (2013). Multilayer networks. *CoRR*, abs/1309.7233.

Leskovec, J. and Krevl, A. (2014a). LiveJournal - SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data/com-LiveJournal.html.

Leskovec, J. and Krevl, A. (2014b). Orkut - SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data/com-Orkut.html.

Liu, G. and Wong, L. (2008). Effective pruning techniques for mining quasi-cliques. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 33–49. Springer.

Padmanabhan, S. and Chakravarthy, S. (2009). Hdb-subdue: A scalable approach to graph mining. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 325–338. Springer.

Pavel, H. R., Roy, A., Santra, A., and Chakravarthy, S. (2023). Closeness centrality detection in homogeneous multilayer networks. In *Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2023, KDIR*.

Rao, R. B. and Lu, S. C. (1992). Learning engineering models with the minimum description length principle. In *AAAI*, pages 717–722.

Santra, A., Bhowmick, S., and Chakravarthy, S. (2017a). Efficient community re-creation in multilayer networks using boolean operations. In *International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland*, pages 58–67.

Santra, A., Bhowmick, S., and Chakravarthy, S. (2017b). Hubify: Efficient estimation of central entities across multiplex layer compositions. In *IEEE International Conference on Data Mining Workshops*.

Santra, A., Komar, K., Bhowmick, S., and Chakravarthy, S. (2022). From base data to knowledge discovery – a life cycle approach – using multilayer networks. *Data & Knowledge Engineering*, page 102058.

Wang, W., Wang, C., Zhu, Y., Shi, B., Pei, J., Yan, X., and Han, J. (2005). Graphminer: a structural pattern-mining system for large disk-based graph databases and its applications. In *Proceedings of ACM SIGMOD 2005*, pages 879–881.

Yang, S., Yan, X., Zong, B., and Khan, A. (2012). Towards effective partition management for large graphs. In *SIGMOD Conference*, pages 517–528.

A Knowledge Map Mining-Based Personalized Learning Path Recommendation Solution for English Learning

Duong Thien Nguyen[1,2][a] and Thu Minh Tran Nguyen[1,2,*][b]
[1]*Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam*
[2]*Viet Nam National University, Ho Chi Minh City, Vietnam*
21C12004@student.hcmus.edu.vn, ntmthu@fit.hcmus.edu.vn

Keywords: Knowledge Graph, Personalized Learning Path, Recommendation, English Learning, Graph Database.

Abstract: Recommendation systems (RS) have been widely utilized across various fields, particularly in education, where smart e-learning systems recommend personalized learning paths (PLP) based on the characteristics of learners and learning resources. Despite efforts to provide highly personalized recommendations, challenges such as data sparsity and cold-start issues persist. Recently, knowledge graph (KG)-based RS development has garnered significant interest. KGs can leverage the properties of users and items within a unified graph structure, utilizing semantic relationships among entities to address these challenges and offer more relevant recommendations than traditional methods. In this paper, we propose a KG-based PLP recommendation solution to support English learning by generating a sequence of lessons designed to guide learners effectively from their current English level to their target level. We built a domain KG architecture specifically for studying English certification exams, incorporating key concept classes and their relationships. We then researched and applied graph data mining algorithms (GAs) to create an effective PLP recommendation solution. Using consistent experimental conditions and a selected set of weights, along with our collected dataset, we evaluated our solution based on criteria such as accuracy, efficiency, stability, and execution time.

1 INTRODUCTION

The amount of data has increased dramatically along with the internet's quick development. Users find it challenging to select what interests them from a wide range of options due to the information overload. RS has been developed to enhance the user experience and aid in making decisions. Personalized recommendations are generated by RS based on user behaviour and preferences, which increases user engagement and happiness on a variety of online platforms, such as music recommendations, movie recommendations, learning path recommendations, online shopping recommendations, etc. (Q. Guo et al., 2020). Despite significant progress, creating a RS specifically suited to provide suitable content remains a challenge. Accurately predicting user and content characteristics and their complex interrelationships is one of these issues. Thus, researchers' attention has been drawn in recent years to the introduction of KG as side information in the RS. A heterogeneous graph with nodes signifying entities and edges denoting relationships between entities is called a KG. To comprehend the relationships between objects, items and their properties can be mapped into the KG. Additionally, users and user-side data may be included in the KG, improving the accuracy of capturing user preferences and relationships with items (Q. Guo et al., 2020).

RS has also benefited academic sectors in many ways since it has driven the creation of smart learning systems. The aims, interests, and abilities of each learner are the learning criteria that these techniques adjust PLP to. Utilizing data-driven monitoring to make sure that learners' parameters are fulfilled, they change the content and order of learning materials, marking the shift from a one-size-fits-all approach to customized learning methodologies (M. Abed, 2023).

Among today's subjects of study, we give special attention to foreign language learning since it plays a

[a] https://orcid.org/0009-0002-2906-8423
[b] https://orcid.org/0009-0009-6961-3976
* Corresponding author

Nguyen, D. and Nguyen, T.
A Knowledge Map Mining-Based Personalized Learning Path Recommendation Solution for English Learning.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 48-59
ISBN: 978-989-758-716-0; ISSN: 2184-3228

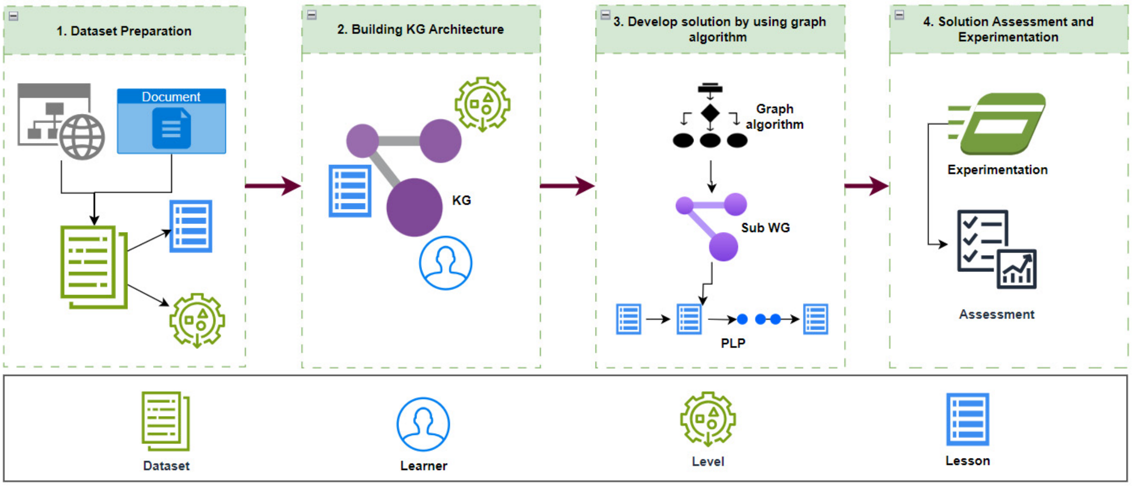

1. Dataset Preparation
Document
2. Building KG Architecture
KG
3. Develop solution by using graph algorithm
Graph algorithm
Sub WG
PLP
4. Solution Assessment and Experimentation
Experimentation
Assessment
Dataset
Learner
Level
Lesson

Figure 1: An overview flowchart illustrating the execution process for the proposed solution.

critical role in job application, study and research, travel, participation in global interchange, etc. (Ilyosovna, N. A., 2020). According to a Statista report published in 2023, English is presently the most common language in the world, with almost 1.5 billion users (E Dyvik, 2023), and certification of competency in English with four primary skills—listening, speaking, reading, and writing—is also frequently expected in job applications and university output standards. As a result, to demonstrate their ability to use English fluently, many individuals must study and prepare for tests to get international English credentials such as TOEIC (M. Schedl, 2010), IELTS, TOEFL (GH Sulistyo, 2009), and others. Learning these qualifications is now much more convenient, owing to the support of smart English learning applications and systems such as Duolingo, Elsa, and others, which allow learners to study more successfully while saving money and time (X. Fan et al., 2023). According to our survey results, these applications generally guide learners to learn in a pre-set sequence based on their goals and current level; however, to guide learners along a suitable learning path (LP) that meets their other personalized requirements, such as time, cost, progress, and learning outcomes, they are also being researched to apply the PLP recommendation (PLPR) models to advise learners on a suitable LP and fulfil the aforementioned aims.

According to that motivation, this study proposes a PLPR solution for English learners using GAs on the KG architecture, which we have developed in the English learning domain. As per the process illustrated in Fig. 1, the proposed solution will proceed through four primary phases of development. Initially, a dataset pertaining to the format, content, and assessment methods of international English certifications and associated exams will be compiled by referencing many websites and official publications. In the second phase, a KG architecture will be constructed to present key concepts about English proficiency levels, knowledge, and skills that are needed, as well as the lessons that correlate to those levels based on the built dataset and the learners' learning requirements. To optimize execution time, in the third phase, we will next create a weighted subgraph (WG) based on the learner's requested learning information in the KG. This created WG will only contain entities and weighted relationship edges related to the target level, as well as the lessons or competencies the learner possesses that correspond to the current level. Next, leveraging the GAs from Neo4j's GDS library (Hodler et al., 2022), we recommend the most effective initial PLP for learners. To identify the optimal set of weights for our solution that meets all LP assessment criteria, we conducted extensive experiments with various weight configurations. Subsequently, employing consistent experimental conditions and a selected set of weights, along with our dataset, we evaluated our solution based on criteria including accuracy, efficiency, stability, and execution time.

This paper is organized as follows: Section 1 outlines our work and its contributions. Section 2 provides an evaluation of the current state of the art in the research field. The KG architectural development process is described in depth in Section 3. Section 4 describes the steps involved in creating a PLPR solution for learners. Section 5 describes the experiments, evaluations, and data collection methods. Finally, Section 6 concludes with suggestions for future research directions.

2 RELATED WORKS

A lot of approaches have been put forth by researchers to increase learning efficiency, and one of the most cutting-edge areas of study these days is developing systems to recommend LPs to e-learners as a chain of learning materials. As a result, numerous studies have been conducted to create RSs for LP recommendations that use semantic dependency links between learning objects (LOs) and learning materials that are simultaneously stored on a variety of data types to recommend LPs to learners, then utilize data mining models to arrange learning materials into learner-recommended LPs. These RS systems do (D. Shi et al., 2020), however, still adhere to the notion of a single learning path that is applicable to all learners and are not actually tailored to the unique learning characteristics of each learner, which results in the recommendation of LPs to learners with limited suitability.

As the research by D. Shi et al. (2020) illustrates, KG has been employed recently for LP recommendation as a prominent research domain since it may eliminate ambiguities in learning content and learner’s learning characteristics descriptions. Motivated by this feature, a few researchers attempted to develop learning systems for KG-based LP recommendation and were successful in resolving the issues raised. Huang and Xiangli (2011) used AI, data mining, and database technology to create a PLP recommendation system (PLP-RS). By fine-tuning learner models with learning history data, it improves specialized services and assesses improvement using customized Knowledge Structural Graphs. Zhang et al. (2023) provide a PLP-RS for e-learning that uses a KG structure by creating a multidimensional course KG and applying graph convolutional networks (GCN) to properly represent learner preferences. The algorithm recommends ideal courses based on both learner preferences and the significance of learning resources, decreasing the need for manual planning and increasing learner satisfaction. Shi et al. (2020) construct a multidimensional KG by connecting learning elements semantically. Their algorithms provide customized LP creation and suggestions, meeting each learner’s unique e-learning demands. Static code analysis is used by H. Yin et al. (2021) to build a structural KG program for open-source projects. Through depth-first and Dijkstra search algorithms, their deep learning model, which integrates this with multi-source data and an LP recommendation mechanism, helps developers quickly learn important functions. Using GCN on Junior High School English exercises, Y. Sun et al. (2021) in the field of English education generate individualized KG for pupils, creating PLP with the aid of Prim and Kruskal algorithms. In their work on computer-assisted vocabulary acquisition, F. Sun et al. (2020) develop a recommendation engine for Chinese vocabulary learning materials utilizing a KG. Hanyu Shuiping Kaoshi (HSK) three-level language resources and ten types of relations are integrated into the system, which was created using Protégé, Apache Jena, and Python. Chen et al. (2021) use course similarity computation and pre-knowledge annotation to automate the creation of Massive Open Online Courses on KG. They use rule-based and machine learning techniques to classify courses, improve TF-IDF computation, and build a network that integrates knowledge and course nodes. The knowledge network and learner data are then used to provide personalized suggestions. Z. Yan et al. (2023) suggest a technique that makes use of a course knowledge network to suggest customized activities. The method entails building the graph using deep knowledge tracing, producing individual knowledge structure diagrams, and producing a Q-matrix from learners’ responses. The model chooses tailored assignments based on factors such as complexity, individuality, and variance, which is consistent with constructivist learning theory.

The aforementioned studies all share the same goal of investigating LP recommendation models for every learner utilizing KG by incorporating concepts such as goals, learner behaviors, LOs, and learning resources, etc., along with their interrelationships, into the architecture of the KG. It has been demonstrated that this method outperforms conventional ones in personalized recommendation outcomes. However, based on the research of M. Abed et al. (2023), we think that other aspects of the learner's learning characteristics, such as the learner's current level of knowledge and skills, desired learning time and cost, etc., must be considered to recommend a more appropriate PLP for each learner. Moreover, little study has been done on learning foreign languages like English. Our study focuses on using KG-based data modelling and processing to develop a solution that recommends a PLP for English language learners. This solution will account for the learners' current knowledge and skills, target level, and desired learning time, enabling them to achieve their goals efficiently within the shortest possible time while adhering to an appropriate learning path.

Section 3 will provide a detailed presentation of the steps involved in developing the KG architecture as well as the proposed solution for this research.

3 DOMAIN KG CONSTRUCTION

We examined the vocabulary topics, grammar themes, scoring scale, format, assessed skill, and evaluation criteria of the TOEIC, IELTS, and TOEFL test components to develop a KG architecture for presenting the concept and learning material along with their relation in preparation for the English certification examinations, as shown in Fig. 1 for the second phase. Furthermore, in accordance with European norms, we investigated the Common European Framework of Reference (CEFR) (B. North et al., 2019) to assess the link between certificate scores and English competencies.

As far as we know, people who want to get an international standard English certificate have to first complete a competence exam and receive the certificate along with a score demonstrating their ability. The score on these certificates does not indicate whether the individual passed or failed, but it does demonstrate their level of English ability. The outcomes can be transferred to the CEFR to standardize English proficiency levels across European and other regional nations. As a result, to manage learners' test information for international English certificates, the KG architecture will include a *Level class* that is focused on storing score information from the current English certificate of the learner and the target score of the certificate that the learner hopes to attain in the future. Each certificate's score information, together with qualification information based on the associated CEFR framework, will be saved as a benchmark to examine the correlation of English proficiency to scores between various certificate types.

Besides*,* success in international English certificate exams necessitates skills in speaking, listening, reading, and writing, as well as mastery of vocabulary and grammar knowledge. Therefore, the Competency class contained in the KG architecture will cover all the necessary skills and knowledge. However, we will construct specific pronunciation and vocabulary knowledge in a separate Lesson class since we understand that this knowledge is only tested in certain parts of the exam. This personalized approach guarantees that important abilities are covered in every segment of the test.

Moreover, learners must fully comprehend the sorts of questions, subjects, and settings that will be posed in each part of the exam. Combine knowledge of grammar, vocabulary, pronunciation, and English abilities to create the ideal test-taking plan. That is, for each level of English that a learner wishes to achieve, the learner must be provided with knowledge from specific lessons on clearly understanding the structure, question type, topic context, strategies, and test-taking experience in that skills test, as well as knowledge from related grammar and vocabulary lessons. As a result, in the KG architecture, an extra Lesson class will be created to manage information about lesson entities that must be learned to pass the exams. Our proposed comprehensive KG, which is represented in Fig. 2, consists of three fundamental concept classes: Level, Competency, and Lesson

For the Level concept class, it will include entity categories such as *Current_Score* and *Target_Score*, which indicate the learners' current score via the *HAS_CURRENT_SCORE* relationship and target score via the *WANT_TARGET_SCORE* connection for the same certificate type with the same properties: *score, certificate*. These certificate's score entity nodes will be referenced to the CEFR competence framework entity nodes, which have properties such as *from_score*, *to_score*, and *certificate*, via the

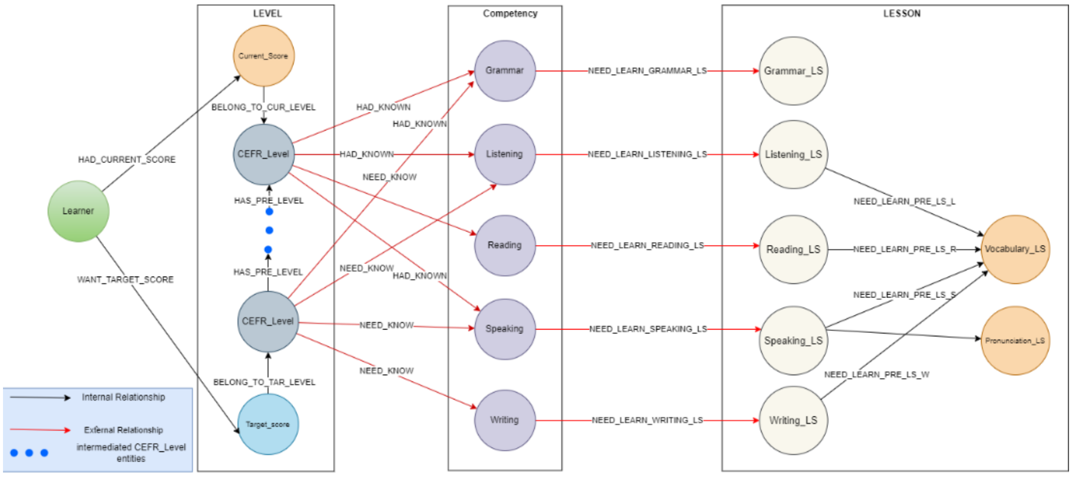


Figure 2: Complete KG architecture utilized in the proposed solution.

internal relationships *BELONG_TO_CUR_LEVEL* and *BELONG_TO_TAR_LEVEL*. Additionally, a *HAS_PRE_LEVEL* relationship will connect CEFR_Level nodes, e.g., a 'B1' level will precede 'A2' according to CEFR.

The Lesson concept class will manage entities comprising main lesson content related to grammar, listening, reading, speaking, and writing skills for each test section. Every lesson has common properties, including *title*, *category*, and *study time* (in days). Additionally, certain preparatory lessons with assigned pronunciation or vocabulary knowledge must be completed before advancing to the main lessons via system linkages like *NEED_LEARN_PRE_LESSON_L,* etc.

Finally, the Competency concept class will signify the learner's current proficiency level on the English certificate and identify acquired skills or knowledge through the *HAD_KNOWN* external relationship with entities like grammar, listening, reading, speaking, and writing lessons. Correspondingly, in alignment with the learner's target level, it outlines the requisite skills and knowledge through the *NEED_KNOW* relationship. Each skill and knowledge entity within the *Competency* class denotes the associated lessons required from the *Lesson* class, establishing external relationships like *NEED_LEARN_GRAMMAR_LS, NEED_LEARN_LISTENING_LS*, and others.

4 A SOLUTION FOR PLPR

4.1 Description of PLPR Problem

The main goal of our proposed solution is to recommend an appropriate PLP for each learner as they go toward preparing for international English certifications like the TOEFL, IELTS, and TOEIC (which include the Speaking-Writing and Listening-Reading combinations). The scores that correspond to the learners' current certificates, information about their English proficiency or lessons that correspond to their current level (which will be raised for the learner to choose based on our developed dataset), the desired study time, and the score that corresponds to the desired certificate are the first inputs of the solution. These inputs will be stored in the KG architecture (as shown in phase 2 of Fig. 1). Our system will then produce an initial PLP for each learner as indicated in phase 3 of Fig. 1, which will include a list of lessons to be learned and progress the learners from their current English level to their target level while accommodating their desired learning schedule.

4.2 End-to-End Solution Processing

As was indicated in Part 3, KG would house all data pertaining to the learning characteristics of learners as well as data on the acquisition and evaluation of English certifications. Simultaneously, we want to provide solutions using a novel approach that is simple to implement while maintaining natural logic and science. Because of this, we have examined and assessed GAs according to several factors, including the KG architecture, the issue that has to be addressed along with the intended outcomes at each stage of the solution's execution, and the fundamentals of how each algorithm works. In particular, we select the graph traversal algorithm BFS (section 4.2.1) for stage 1 of the solution in order to be able to create a subgraph with only entities connected to the learner's target-level entity. The PageRank algorithm is then combined with the LPA_NI algorithm in stage 2 of the solution with the aim of determining the importance of each lesson entity on WG and clustering these entities into clusters corresponding to the list of lessons to be learned from the current level to the target level, then merging into the original LP (section 4.2.2). Lastly, we use the Min Weighted Sum (MWS) approach to assess and determine which LP is the most appropriate as a recommended PLP for learners and meet the optimization objectives in stage 3 (section 4.2.3). Fig. 3 will provide details of the processing flow for each stage, precisely as follows:

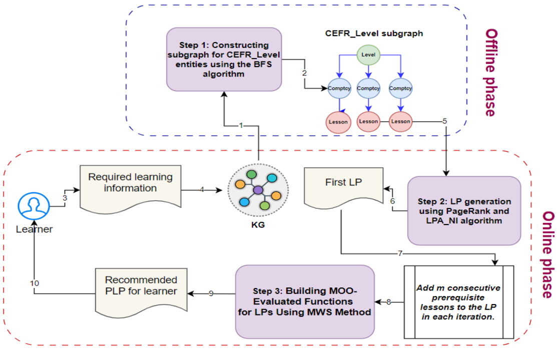

Step 1: Constructing subgraph for CEFR_Level entities using the BFS algorithm
CEFR_Level subgraph
Level
Comptcy
Comptcy
Comptcy
Lesson
Lesson
Lesson
Offline phase
1
2
5
Required learning information
3
4
KG
Learner
First LP
6
Step 2: LP generation using PageRank and LPA_NI algorithm
Online phase
7
Recommended PLP for learner
10
9
Step 3: Building MOO-Evaluated Functions for LPs Using MWS Method
8
Add m consecutive prerequisite lessons to the LP in each iteration.

Figure 3: The execution flow with applied algorithms.

Step 1: Constructing a subgraph for CEFR_Level entities using the BFS algorithm: This initial step involves offline processing to traverse the KG using the BFS algorithm (S. Huan, 2014). The goal is to generate subgraphs for each CEFR_Level entity. By doing so, we create a comprehensive list of all entity categories that are directly or indirectly connected to each CEFR_Level entity. This approach reduces the number of entity interactions, thereby optimizing the execution time for subsequent steps.

Step 2: LP generation using PageRank and the LPA_NI algorithms: This step occurs during the online processing phase. It starts by transforming the CEFR_Level entity subgraph into a WG tailored to the learner's target level, based on information provided by the learner. Next, the PageRank algorithm (C. Tulu, 2020) is employed to evaluate the relevance of each node within the WG. Following this, the LPA_NI algorithm (Zhang, 2017) groups significant nodes into clusters, reflecting the competencies and lessons required for each CEFR_Level entity in the WG. By merging these clusters and sorting them in ascending order according to the CEFR_Level entity values within each cluster, the initial learning path comprising the primary lessons is obtained.

Step 3: Building Multi-Objective Optimization (MOO)-Evaluated Functions for LPs Using the MWS Method: This step will also be completed online. The LP made in step 2 is to keep adding *m* significant nodes in the WG as prerequisite lessons as nodes in the Vocabulary or Pronunciation entity category and then utilize the developed evaluation function to gauge the LP's satisfaction at each *k*th iteration by using the MWS method. Then, as the PLP to counsel the learner, select the LP that produces the most optimal outcome while meeting all stated optimization objectives. In the following sections, we will present the details of these main processing steps.

4.2.1 Constructing Subgraph for CEFR_Level Entities

The implementation procedure for step 1 is detailed in Algorithm 1.

Algorithm 1: Constructing subgraph for each CEFR Level entities in KG.

```
Input:
    ▪ G (V, E): The KG includes V vertices and E relationship edges.
    ▪ LVL = {LVL_i | i = $\overline{1,n}$}: set of the i-th CEFR_Level entity denoted as LVL_i contained in G (V, E).
    ▪ n: number of elements in the LVL set.
Output: LV_EN_i (set of entities related to each i-th CEFR_Level entity).
LV_EN_i ← ∅ , i ← 1
while i ≤ n:
    Apply the BFS with each LVL_i as the source vertex → Obtain a set containing k nodes {v_1, v_2, …, v_k}
    LV_EN_i ← {v_1, v_2, …, v_k}
    i ← i + 1
End while
```

For example, after executing this algorithm, as shown in Fig. 4, we obtain a subgraph of the CEFR_Level node with the value "CEFR_B2," representing learner X's target level. This subgraph includes nodes related to the learner's current level, their existing competencies, the skills they need to acquire, and the lessons that they might have to learn.

4.2.2 LP Generation Using PageRank and LPA_NI Algorithms

To clearly explain the implementation process in step 2, we introduce the notations outlined in Table 1 and describe the two primary tasks. The first task involves using the PageRank algorithm, detailed in Algorithm 2. Additionally, we use Eq. 1 (C. Tulu et al., 2020) to calculate the PageRank score (PR_score) for each node in the WG derived from the subgraph defined in step 1:

$$PR(i) = (1-d) + d\sum_{j\to i}\frac{W_{ji}PR(j)}{\sum_k W_{kj}} + PR'(i) \qquad (1)$$

Algorithm 2: Determine each node's significance within the WG.

```
Input: TAR, CUR, subgraph of TAR as G' (v, e), CPT_HAD, LS_KNOWN.
Output: IPT set.
$WG \leftarrow G'$
For each edge e point to node u in WG:
    If (u ∈ CPT_HAD)||(u ∈ LS_KNOWN)
        e.weight ← 0
    Else e.weight ← 1
For each node u in WG:
    If (u == CEFR_Level entity)
        PR_score(u) ← 1
    Else PR_score(u) ← 0
While not converged:
    For each node u in WG:
        PR_old ← PR_score(u)
        Using Eq.1 to calculate PR_score(u)
        If |PR_score(u) - PR_old| < threshold:
            break loop
IPT = {u | PR_score(u) > 0 and sort by PR_score(u) decreasing}
```

Note that in Eq. 1 (E. Turan et al., 2020), *PR(i)* denotes the PR_score calculated for each node *i* in the *LV_EN* set during the current iteration, while *PR'(i)* represents the existing PR_score of node *i* from the previous iteration, indicating the spread of points among related nodes. *PR(j)* refers to the current PR_score of nodes *j* in the *LV_EN* set linked to node *i*. The weight of the edge from node *j* to node *i* is denoted as W_{ji}. Similarly, W_{kj} represents the weight of nodes *k* in the *LV_EN* set, pointing away from node *j*. The damping factor *d*, set as 1, reflects the probability of the learner accessing node *i* from node *j*, ensuring

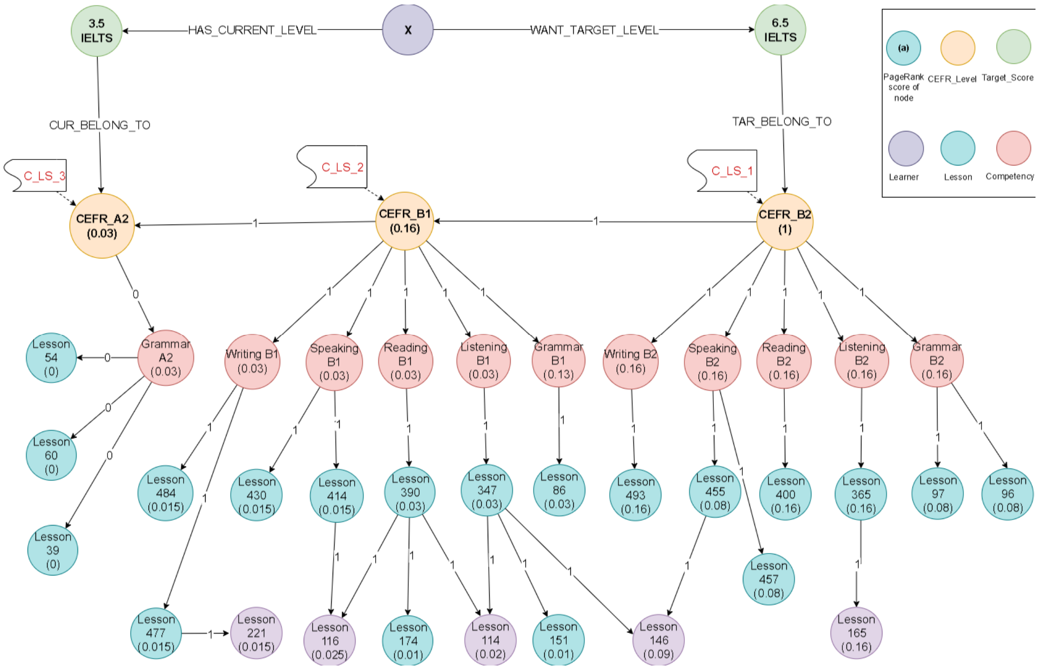


Figure 4: Weighted subgraph for the learner's target level completed after executing step 1 and 2.

Table 1: The meaning of the signs used in Step 2.

Signs	Meaning
LV_EN	Set of entities (competence, previous CEFR level, lesson) related to the learner's target CEFR level.
TAR	English proficiency according to the CEFR framework on the target certificate that the learner wants to achieve.
CUR	English level on the current certificate according to the CEFR framework that the learner currently has.
CPT_HAD	Set of competencies that the learner already has. Equivalent to a competency number belonging to CUR.
LS_KNOWN	Set of lessons that the learner has learned before (lessons that the learner can optionally learn) belongs to the competencies of CUR.
EN_IPT	Set of CEFR_Level, Competency, and Lesson entities has decreasing importance to learners according to their PR score, which is greater than 0.
INTM_LV	The set contains intermediate CEFR_Level nodes between TAR and CUR.
CL_EN_u	The u^{th} cluster contains nodes with the same label after each label propagation step.
LN_LS_u	The set contains only lesson entities filtered from the corresponding CL_EN_u clusters.

learning from node j to node i for pairs with $W_{ji} = 1$. For instance, in Fig. 4, based on the learner's input data, each edge pointing to a node in the subgraph is given a weight value of either 0 or 1. The subgraph will be transformed into the WG following this weight assignment procedure. Once Algorithm 2 has run on this WG and assigned a PR_score to each node, we will add these nodes to the *EN_IPT* set in decreasing order of their PR_scores. Fig. 5 presents the *EN_IPT* set as an example.

Based on the WG architecture designed in Fig. 4, when learner X wants to achieve a TAR (e.g., level "B2") from a CUR (e.g., level "A2"), learners must also achieve the Competencies of the intermediate

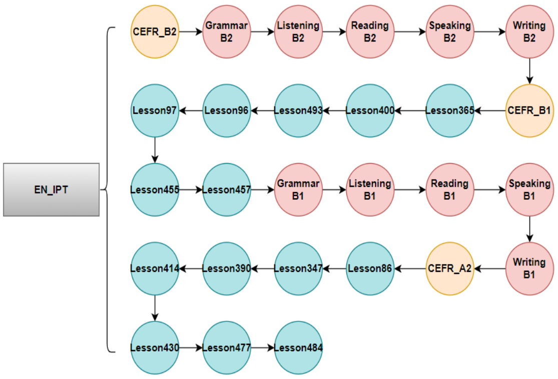


Figure 5: Entity nodes included in the EN_IPT set.

levels (INTM_LV) (for example, “B1”). To guarantee that learner X studies enough lessons for the needed Competencies from CUR to TAR, the second task in this step will cluster the most critical lessons to learn (according to the PR_score of each node in the WG) into each cluster at each level of proficiency. LPA, a well-liked clustering algorithm (Čížková, K. 2022), uses graph design to build a label propagation mechanism for random nodes. Nevertheless, we will use the method developed by Zhang et al., which is called LPA_NI, to boost second task efficiency. When propagated, LPA_NI has been demonstrated to provide superior clustering results over regular LPA based on node importance and label influence. Eqs. 2 and 3 (Zhang et al., 2017) are used by the LPA_NI for this step, where *LI (i, lb)* denotes the label's influence *(lb)* on node *i, d(j)* denotes the outdegree of node *j*, $N^{l}(i)$ denotes the set of labels *lb* surrounding node *i*, c_i denotes the most influential label that will be assigned to node *i*, and *l_max* denotes the sets of the maximum number of labels.

Algorithm 3: Building the first LP.

```
Input:  WG of TAR, CUR, EN_IPT, MaxIter
        (Maximum number of execution loops)
Output: LN_LP.
  1: Initialize seedLabel for CEFR_Level nodes in
     WG.
  2: t ← 1
  3: For each node x ∈ EN_IPT:
  4:      Assign label of most represented connected
          node.
  5: If connected nodes' labels to x are all different:
  6:        Calculate viral influence using Eq. 2.
  7: Choose label satisfying Eq. 3 to update node x.
  8: If t = MaxIter or labels of node x match majority
     connected nodes' labels:
  9:      Assign nodes x to CL_EN_1, CL_EN_2, ...,
          CL_EN_k with specified labels.
 10:      End.
 11: Else
 12:      t ← t + 1;
 13:      Repeat steps 3 – 10.
 14: For each CL_EN_1, CL_EN_2, ..., CL_EN_k:
 15:      Initialize LN_LS_u (u = $\overline{1,k}$) containing
          Lesson entities for each cluster.
 16: Create set LN_LP = LN_LS_1 ∪... ∪ LN_LS_u
     containing required lessons.
```

The following algorithm 3 describes in detail the idea of this task. Furthermore, in accordance with the example shown in Figure 6, the nodes on the WG will be split into two clusters, *C_LS_1* and *C_LS_2*, following the completion of algorithm 3. Next, entities of the type of Lesson will be chosen from each cluster to create the appropriate *LN_LS_1* and *LN_LS_2* additional clusters. Finally, we will combine the entities in the aforementioned two clusters and rearrange them in the order of rising PR_score values to construct an initial LP, known as the *LN_LP* set, which will include the key lessons to be learned from the current level to the target level.

$$LI(i, lb) = \sum_{j \in N^{l}i} \frac{PR(j)}{d(j)} \qquad (2)$$

$$c_i = \underbrace{argmax\, LI(i, lb)}_{lb \in l\_max} \qquad (3)$$

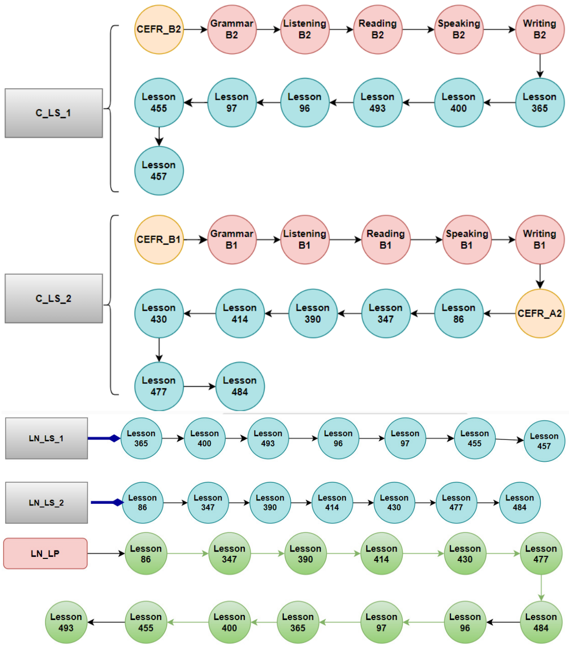


Figure 6: The process of building the LN_LP set in step 2.

4.2.3 Building a MOO Function Using the MWS Method

Not only should the PLP that is recommended to learners be a collection of lessons that are taught in a sequential manner and cover the competencies that the learner needs to master, but it should also contain a number of prerequisite lessons, or lessons that must be studied prior to studying the main lessons that are directly taught to achieve competencies. The current solution will be to provide an LP so that learners only need to learn a minimal amount of vocabulary, covering as many required main lessons as possible, as there is currently no specific statistical report on the amount of vocabulary required to be learned at each level of English certification exams.

In light of these remarks, in this step, our solution will develop an evaluation function based on the MWS method (N. Gunantara, 2018) to assess each *LN_LP's* optimization objectives in each iteration.

We set parameter *m* as a fixed number of consecutive prerequisite lessons taken from the *PRE_LS* set and then added to the existing *LN_LP* in each iteration. The proposed weights for each objective to be optimized in the evaluation functions are described in Table 2. Finally, based on the MWS formula, utilizing information from the weight set and value function for each objective (refer to table 2), let *x* represent the existing *LN_LP* in the k^{th} iteration. The MWS formula for the *LN_LP* evaluating function is expressed as in Eq. 4., which states that the LP with the lowest overall optimization score for all objectives will be deemed to be the most optimum LP when each LP in each loop has four goals that need to be optimized and each goal has a weight indicating the attached priority. The implementation procedure of step 3 is shown in Algorithm 4, and the phases are illustrated in Fig. 7 to illustrate how they are carried out. Specifically, at every k^{th} iteration, we will progressively add one required lesson to the LP that existed in the *(k-1)* th iteration. Concurrently, we utilize Eq. 4 to determine the evaluated score for each LP. In the end, only the LP found in the fifth loop will be chosen as the PLP to recommend to the learner since it fulfills Eq. 4.

Table 2: The weights and value functions of objectives.

Weight	Function	Meaning
w_1	$f_1(x)$	Maximize the number of competency entity types present in the LN_LP set.
w_2	$f_2(x)$	Minimize the number of prerequisite lesson entities (which are vocabulary or pronunciation lessons) learned enough for the required lessons in LN_LP.
w_3	$f_3(x)$	Minimize the inverse sum of the PageRank (PR) scores of lessons in LN_LP.
w_4	$f_4(x)$	Minimize the number of lessons left over in LN_LP after being evaluated.

Algorithm 4: Building the completed LP as PLP.

```
Input: WG, LN_LP, EN_IPT, m
Output: LN_LP.
  Initialize PRE_LS = ∅.
  For each node u in WG:
    If ((u == Vocabulary entity || u == Pronunciation
    entity) && u ∈ EN_IPT:
        PRE_LS ← PRE_LS ∪ {u}.
  Initialize LN_LP_L = {LN_LP}.
  While (PRE_LS ≠ ∅)
      Last_LN_LP = GetLastElement (LN_LP_L).
      Add m Lessons entities category from
      LN_PRE_LS to Last_LN_LP.
      Calculate Evaluation Score for Last_LN_LP
      using MWS with Eq.4.
      Add Last_LN_LP to LN_LP_L.
      |PRE_LS| = |PRE_LS| - m.
  Select the best LN_LP from LN_LP_L based on
    optimal evaluation score.
```

$$minF(x) = \sum_{i=1}^{4} f_i(x).w_i \mid \begin{cases} \sum_{i=1}^{4} w_i = 1 \\ w_3 > 0 \\ 0 \le w_i \le 1\ v\acute{o}i\ i = \overline{1,4} \end{cases} \tag{4}$$

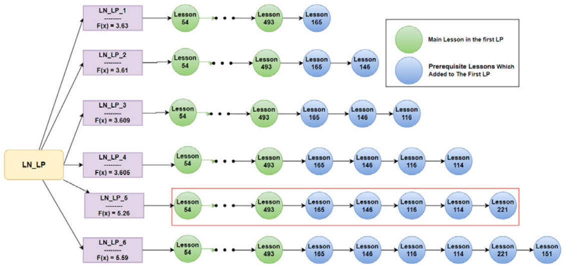


Figure 7: Development of the complete LN_LP in step 3.

5 EXPERIMENTATION AND EVALUATION

We conducted the experimentation process by considering the learner's aspiration to advance from the lowest current level and all proficiency levels of the learner's knowledge and skills, which are not yet there, to the highest target level (aligned with the CEFR competency framework: 'TOEIC (L-R)-C1', 'TOEIC (S-W)-C1', IELTS-C2', 'TOEFL-C2'). Specifically, we focused on step 3 of the solution, varying the chosen weight sets and adding a fixed number of consecutive prerequisite lessons *(m = 5)* to the LP in each k^{th} iteration. Using Eq. 4 and the completed dataset, we identified the optimal weight set for this step. We then compare approaches using the combined PageRank algorithm with the traditional LPA algorithm (PR_LPA for short), and the approach used in solution development applies the PageRank algorithm combined with the LPA_NI method (PR+LPA_NI for short) to assess the performance, stability, accuracy, and efficiency of our proposed solution (M. Abed et al., 2023) (Nabizadeh et al., 2020). The identical experimental dataset and weight set that were established following the experiment will be used for this comparison.

5.1 Dataset Building

There is currently hardly any standardized dataset that announces the learning content and skills required for

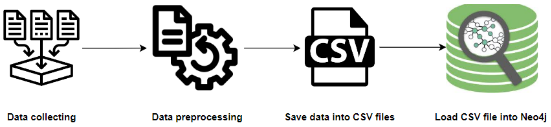


Figure 8: The process of building experimental dataset.

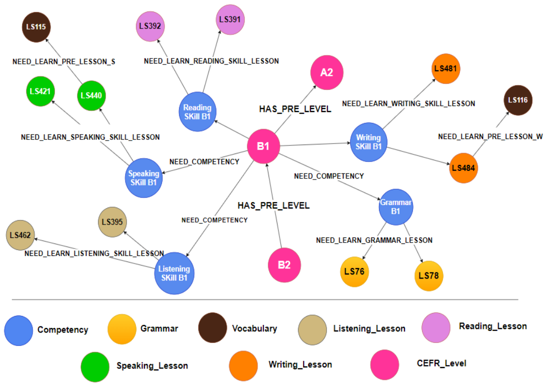


Figure 9: A part of the nodes and their relations in KG.

Table 3: Statistics on the number of entities and relationships in the KG.

Entities	Amount	Relations	Amount
		NEED_KNOWN / HAD_KNOW	63
CEFR_Level	22	NEED_LEARN_GRAMMAR	98
Competency	49	NEED_LEARN_LISTEN_SKILL	102
		NEED_LEAN_READING_SKIL	68
Grammar_LS	98	NEED_LEAN_SPEAKING_SKL	115
Listening_LS	79	NEED_LEARN_WRITING_SKIL	54
Pronunciation_LS	7	NEED_LEAN_PRE_LESSON_L	103
Reading_LS	54	NEED_LEARN_PRE_LESSON	59
Speaking_LS	91	NEED_LEAN_PRE_LESSON_S	66
Vocabulary_LS	124	NEED_LEAN_PRE_LESSON_W	12
Writing_LS	39	NEED_LEN_PRONUNCIATION	7

these English certificates according to each level, according to our survey conducted on various websites, official reference documents, and the organizations that organize these exams. Therefore, we followed the procedure outlined in Fig. 8 to produce a data set appropriate for the experimental and assessment phases.

Steps 1 and 2: Data Collection and Processing: We gather information on English certification exam formats, knowledge matter, and evaluation standards from the official homepage of ETS, the British Council, and some standard documents about these exams. Convert this information into English lesson units, detailing certification levels, required competencies, and specific lessons in grammar, vocabulary, pronunciation, listening, speaking, reading, and writing. Transform the collected data into entities, relationships, and properties matching the KG architecture.

Steps 3 and 4: Saving reprocessed data as a CSV file and importing it into Neo4j: Create CSV files containing entity and relationship data from step 2. Import these files into Neo4j using its import function to generate a comprehensive graph database schema aligned with the KG architecture. The number of entities and relationships in the KG architecture is presented in Table 3, and a part of the data set in the KG architecture is shown in Fig. 9. A full experimental dataset is now available on Kaggle.

5.2 Experimental Results and Evaluation Findings

Following the experimentation method outlined above, we discovered that all of the sets of weights tested indicated that the number of lessons was adequate to meet the necessary knowledge, skills, and lessons. Additionally, we discovered that the number of lessons—the number of prerequisite lessons that are redundant in the PLP recommended—remained unchanged in all four types of English qualification certificates. Simultaneously, the LP's evaluating function score tends to drop while the objectives' weights exhibit a significant value difference. This implies that when the objectives' weights are nearly equal, the best LP will be guaranteed when the optimal goals are deemed nearly equally important. Ultimately, we concluded that the set of weights *{w1 = 0.28, w2 = 0.27, w3 = 0.25, w4 = 0.2}* is the most ideal one to employ for this solution since it fits the requirements of the evaluation function as in Eq. 4.

As previously said, we compare the accuracy, efficiency, stability, and performance of the two approaches to the solution PR+LPA_NI and PR+LPA to assess our solution. Effectiveness is illustrated by presenting a PLP with scores from the evaluating function that conforms to the requirements in Eq. 4 and has the lowest score. Accuracy is determined by the number of lessons in PLPs that are sufficient for the number of Competency types required, and the two values of this quantity must be smaller or equivalent to the number of entities in the Lesson and Competency classes in the original KG architecture. The constancy of the PLP output across several runs with the same input data is known as stability, and the suggested PLP is used to assess the performance.

When it comes to efficiency, Fig. 10 demonstrates that the PLP's evaluating score recommended by the solution for implementation in the PR+LPA_NI or PR+LPA approach consistently satisfies Eq. 4, yet the PR+LPA_NI approach almost produces the optimal

Table 4: The percentage of Lesson entities that meet the Competency entities needed to learn in the recommended PLP.

Type of Recommendation (1)	**Number of Competency entities required (2)**	**Number of Lesson entities to learn (3)**	**Number of Lesson entities in PLP (4)**	**Number of Competency entities learned in PLP (5)**	**Competency entity rate is met** **(6) = (5)/ (2)**	**Lesson entity rate is met** **(7) = (4)/ (3)**
TOEIC L-R (A1-C1)	12	198	161	11	91,67%	81,31%
TOEIC S-W (A1-C1)	12	161	124	11	91,67%	77,02%
IELTS (A1-C2)	20	309	272	19	95%	88,03%
TOEFL (A1 – C2)	16	148	111	15	93,76%	75%

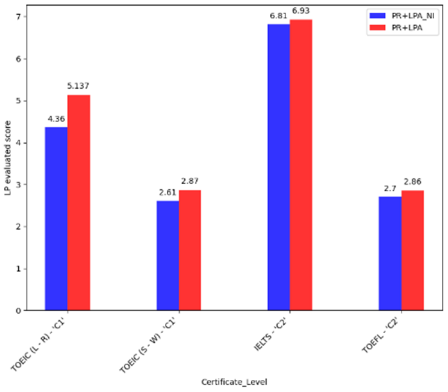


Figure 10: Evaluation scores of the recommend PLP on two algorithms.

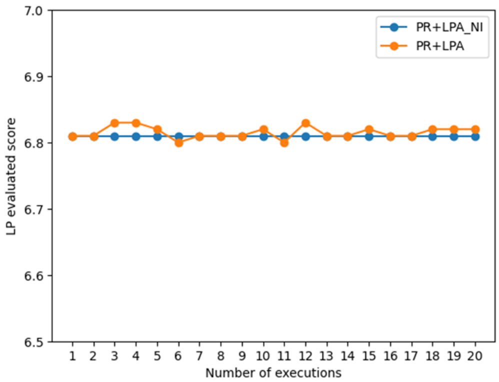


Figure 11: Execution time when executing on two algorithms for making PLP in “IELTS - C2”.

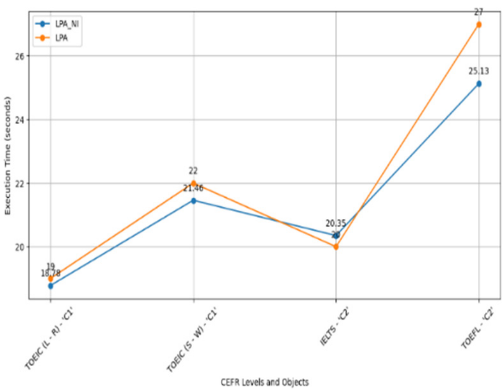


Figure 12: Results when executed on two algorithms in multiple executions.

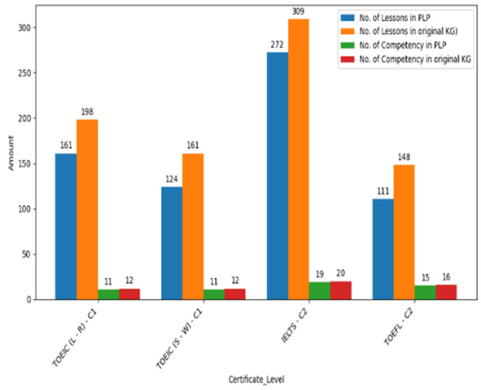


Figure 13: Number of lessons and competencies in PLP of PR+LPA_NI solution.

PLP evaluating score when compared to PR+LPA. Simultaneously, we examine the example with input data for the "IELTS-C2" certificate to recommend PLP to learners, as illustrated in Fig. 11. The solution implemented using the PR+LPA_NI approach yields the PLP evaluation function score nearly unchanged through multiple executions with the same input data in comparison to the PR+LPA approach, and the outcomes are also comparable when applied to other certificate types. Moreover, Fig. 12 indicates that PLP, as recommended by the PR+LPA_NI technique, has an approximately faster execution time than PR+LPA. Finally, when considering accuracy, based on Fig. 13 and Table 4, the solution proposed when developed in the direction of PR+LPA_NI or the PR+LPA approach all recommends being PLP for the proportion of Lesson entities almost learning enough for the required Competency entities, and the number of entities is smaller than the number of original KG.

Overall, the solution developed as a PR+LPA_NI approach route better met assessment requirements than the PR+LPA approach.

6 CONCLUSIONS

Our work developed a comprehensive solution for recommending PLPs in the English learning domain. First, we designed a KG architecture to represent key concept layers and their relationships for learning resources in international English certifications. Next, we utilized GAs and objective optimization techniques to generate the most suitable personalized learning paths. Through rigorous assessment and testing, our solution has proven to effectively generate PLPs that meet established evaluation standards and align with learners' consultation needs. To assist learners in completing their learning program as quickly and effectively as possible, future research will concentrate on developing an adaptive LP recommendation system (I. Katsaris, 2021) that modifies the original PLP in real-time after a predetermined amount of time by improving algorithms or technical processes for processing learners' learning progress data.

ACKNOWLEDGEMENTS

This research is partially supported by research funding from the Faculty of Information Technology, University of Science, VNU-HCM, Vietnam.

This research is funded by University of Science, VNU-HCM, Vietnam under grant number CNTT 2023-15.

This research is partially funded by the Vingroup Innovation Foundation (VINIF) under the grant number VINIF.2021.JM01.N2.

REFERENCES

Guo et al. (2020). A survey on knowledge graph-based recommender systems. IEEE Trans. Knowl. Data Eng., 34(8), 3549-3568.

Mansouri, N., Soui, M., & Abed, M. (2023, Sept). Full Personalized Learning Path Recommendation: A Literature Review. In AISI (pp. 185-195). Springer.

Ilyosovna, N. A. (2020). The importance of English language. *International Journal on Orange Technologies*, *2*(1), 22-24.

Dyvik, E. (2023). The most spoken languages worldwide 2023. *Statista. Retrieved.*

Fan, X., Liu, K., Wang, X., & Yu, J. (2023). Exploring mobile apps in English learning. *Journal of Education, Humanities and Social Sciences, 8*, 2367-2374.

Hodler, A. E., & Needham, M. (2022). Graph data science using Neo4j. In *Massive Graph Analytics* (pp. 433-457). Chapman and Hall/CRC.

Shi, D., Wang, T., Xing, H., & Xu, H. (2020). A learning path recommendation model based on a multidimensional knowledge graph. *Knowledge-Based Systems, 195*, 105618.

Huang, X. (2011). Study of personalized E-learning system based on knowledge structural graph. *Procedia Engineering*, *15*, 3366-3370.

Zhang, X., Liu, S., & Wang, H. (2023). Personalized learning path recommendation based on knowledge graph and graph convolutional network. *Int. J. Software Eng. Knowl. Eng.*, 33(01), 109-131.

Yin, H., Sun, Z., Sun, Y., & Huang, G. (2021). Automatic learning path recommendation for open-source projects using deep learning on knowledge graphs. In *2021 IEEE 45th Annual COMPSAC*, pp. 824-833.

Sun, Y., Liang, J., & Niu, P. (2021). Personalized recommendation of English learning based on knowledge graph and graph convolutional network. In ICAI Security (pp. 157-166). Springer.

Sun, F., Yu, M., Zhang, X., & Chang, T. W. (2020). A vocabulary recommendation system based on knowledge graph for Chinese learning. In 2020 IEEE 20th ICALT (pp. 210-212).

Chen, H., Yin, C., Fan, X., Qiao, L., Rong, W., & Zhang, X. (2021). Learning path recommendation for MOOC platforms based on a knowledge graph. In *KSEM 2021* (Vol. 14, pp. 600-611). Springer.

Yan, Z., Hongle, D., Lin, Z., & Jianhua, Z. (2023). Personalization exercise recommendation framework based on knowledge concept graph. *Computer Science & Information Systems, 20*(2).

Huang, S., Cheng, J., & Wu, H. (2014). Temporal graph traversals: Definitions, algorithms, and applications. *arXiv preprint arXiv:1401.1919*.

Turan, E., Arslan, E., Tülü, Ç., & Orhan, U. (2020). A comparison of graph centrality algorithms for semantic distance. *Lapseki Meslek Yüksekokulu Uygulamalı Araştırmalar Dergisi, 1*(2), 61–70.

Zhang, X. K., Ren, J., Song, C., Jia, J., & Zhang, Q. (2017). Label propagation algorithm for community detection. *Physics Letters A, 381*(33), 2691-2698.

Gunantara, N. (2018). A review of multi-objective optimization: Methods and its applications. Cogent Engineering, 5(1), 1502242.

Čížková, K. (2022). Comparing two community detection algorithms and their applications on human brains.

North, B., & Piccardo, E. (2019). Developing new CEFR descriptor scales and expanding the existing ones. *Zeitschrift Fremdsprachenforschung, 30*(2), 142-160.

Katsaris, I., & Vidakis, N. (2021). Adaptive e-learning systems through learning styles: A review of the literature. *Advances in Mobile Learning Educational Research, 1*(2), 124-145.

Nabizadeh, A. H., Leal, J. P., Rafsanjani, H. N., & Shah, R. R. (2020). Learning path personalization and recommendation methods: A survey of the state-of-the-art. *Expert Systems with Applications*, *159*, 113596.

Positive-Unlabeled Learning Using Pairwise Similarity and Parametric Minimum Cuts

Torpong Nitayanont[a] and Dorit S. Hochbaum[b]
Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA, U.S.A.
torpong_nitayanont@berkeley.edu, dhochbaum@berkeley.edu

Keywords: Positive-Unlabeled Learning, Binary Classification, Pairwise Similarity, Parametric Minimum Cut.

Abstract: Positive-unlabeled (PU) learning is a binary classification problem where the labeled set contains only positive class samples. Most PU learning methods involve using a prior π on the true fraction of positive samples. We propose here a method based on Hochbaum's Normalized Cut (HNC), a network flow-based method, that partitions samples, both labeled and unlabeled, into two sets to achieve high intra-similarity and low inter-similarity, with a tradeoff parameter to balance these two goals. HNC is solved, for all tradeoff values, as a parametric minimum cut problem on an associated graph producing multiple optimal partitions, which are nested for increasing tradeoff values. Our PU learning method, called *2-HNC*, runs in two stages. Stage 1 identifies optimal data partitions for all tradeoff values, using only positive labeled samples. Stage 2 first ranks unlabeled samples by their likelihood of being negative, according to the sequential order of partitions from stage 1, and then uses the likely-negative along with positive samples to run HNC. Among all generated partitions in both stages, the partition whose positive fraction is closest to the prior π is selected. An experimental study demonstrates that *2-HNC* is highly competitive compared to state-of-the-art methods.

1 INTRODUCTION

Positive-unlabeled (PU) learning is a variant of binary classification where labeled samples only come from the positive class. Each unlabeled sample could either belong to the positive or negative class. PU learning is related to the one-class learning problem in which the model is trained solely on the positive labeled set, but unlabeled samples are not utilized (Khan and Madden, 2014). PU learning is also related to semi-supervised learning, where unlabeled samples are used in addition to the labeled set of samples from both classes, giving better performances than one-class learning methods (Lee and Liu, 2003; Li et al., 2010). PU learning is a special case of semi-supervised learning where no negative labeled samples are provided.

PU learning arises in contexts where negative samples are difficult to verify or obtain, and when the absence of positive label does not always imply that the sample is negative. In personalized advertising (Yi et al., 2017; Bekker and Davis, 2020), each advertisement that is clicked is a positive sample. However, an unclicked advertisement is regarded as unlabeled as it could either be uninteresting (*negative*) or interesting but overlooked (*positive*). In the identification of malignant genes (Yang et al., 2012; Yang et al., 2014a), a limited set of genes have been verified to cause diseases (*positive*) while many other genes have not been evaluated (*unlabeled*). Other domains include fake reviews detection (Li et al., 2014; Ren et al., 2014) and remote sensing (Li et al., 2010).

A natural way to deal with the absence of negative labeled samples is to identify unlabeled samples that are likely negative, and train a traditional classifier using the positive labeled set and the likely-negative unlabeled set (Liu et al., 2002; Li and Liu, 2003). Another common approach is to train a classifier on a modified risk estimator, in which each unlabeled sample can be regarded as positive and negative with different weights. This idea has been adopted in different learning methods such as neural network models (Du Plessis et al., 2014; Du Plessis et al., 2015; Kiryo et al., 2017), and random forest (Wilton et al., 2022) with a modified impurity function. Most of these methods rely on the prior information of the fraction of positive samples, π, in the dataset.

The method that we propose here is based on a network flow-based method called Hochbaum's Nor-

[a] https://orcid.org/0009-0002-6976-1951
[b] https://orcid.org/0000-0002-2498-0512

Nitayanont, T. and Hochbaum, D.
Positive-Unlabeled Learning Using Pairwise Similarity and Parametric Minimum Cuts.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 60-71
ISBN: 978-989-758-716-0; ISSN: 2184-3228

malized Cut (HNC) (Hochbaum, 2010). HNC partitions samples into two sets to achieve high intra-similarity within sets and low inter-similarity between the two, with a tradeoff parameter that balances the two goals. The problem was shown in (Hochbaum, 2010) to be solved, for all values of the tradeoff parameter, as a minimum cut problem on an associated graph. This method was previously used as in binary classification where both positive and negative labeled samples are available (Yang et al., 2014b; Baumann et al., 2019). HNC is applicable in PU learning since it does not require labeled samples from both classes. Moreover, it makes use of unlabeled samples through their similarities with labeled samples and among themselves, making it advantageous when labeled data is limited.

As a transductive method, HNC predicts labels only for the given unlabeled samples. This is different from inductive methods that make predictions for any unlabeled samples, whether they are the given unlabeled samples, or unseen, separate set of unlabeled samples. Indeed, HNC can be extended and used as an inductive classifier.

The main contribution of this work is a new method for PU learning that utilizes the unique features of HNC in two stages, called *2-HNC*. Stage 1 generates multiple partitions of data samples, corresponding to different tradeoff values, efficiently, with a parametric cut procedure. We infer from the sequence of partitions in stage 1 the likelihood of unlabeled samples to be negatively labeled. Based on this, stage 2 generates a set of likely-negative unlabeled samples and apply HNC using both the positive samples and the likely-negative samples. Among all partitions generated in both stages, the one whose fraction of positive samples is closest to the given prior π is selected as the prediction for unlabeled samples.

Additional and independent contribution here is the method of extracting likely-negative samples from the unlabeled set using results of stage 1. This method has potential uses in settings other than PU learning. Another contribution of this work is the consideration of the intra-similarities of both positive and negative prediction sets, in data partitioning. This is in contrast to past uses of HNC, such as in (Baumann et al., 2019; Spaen et al., 2019; Asín Achá et al., 2020), where the scenario considered was to maximize the intra-similarity of the positive prediction set only.

We show via experiments on real data that 2-HNC outperforms leading methods, which include two standard benchmarks, *uPU* (Du Plessis et al., 2014; Du Plessis et al., 2015) and *nnPU* (Kiryo et al., 2017), as well as a recent state-of-the-art tree-based method, *PU ET* (Wilton et al., 2022).

2 RELATED WORKS

The main challenge of PU learning is the lack of negative labeled samples. A number of methods utilize a preprocessing step to identify a set of unlabeled samples that are likely to be negative prior to training a traditional binary classifier. For instance, the Spy technique (Liu et al., 2002) selects a few positive labeled samples as *spies* and include them in the unlabeled set, all of which are treated as negative. With a binary classifier trained on this data, unlabeled samples with lower posterior probability than the spies are considered likely to be negative. The Rocchio method (Li and Liu, 2003) marks unlabeled samples that are closer to the centroid of unlabeled samples than that of positive labeled samples as likely negative. (Lu and Bai, 2010) used Rocchio to also expand the positive labeled set when a small labeled set is given.

Another common approach in recent works is to train a model based on an empirical risk estimator, modified in the context of PU learning. (Du Plessis et al., 2014; Du Plessis et al., 2015) proposed *uPU*, an unbiased risk estimator for PU data on which neural network models are trained. (Kiryo et al., 2017) mitigates the overfitting nature of uPU via a non-negative risk estimator in their state-of-the-art method known as *nnPU*. There are also works on other classifiers, besides deep learning models, that apply this similar idea such as a random forest model called *PU ET* (Wilton et al., 2022), in which the impurity function is modified for PU data. PU ET gives competitive results, especially on tabular data type where deep learning PU methods are not always effective.

There are methods, other than the above, which rely on pairwise similarities between samples. In label propagation method of (Carnevali et al., 2021), a graph representation of the data is constructed with edge weights that reflect pairwise similarities. The likelihood of being negative for each unlabeled sample is inferred based on its shortest path distance on the graph to the positive labeled set. Labels are then propagated from the positive and likely-negative unlabeled samples to the remaining unlabeled ones. (Zhang et al., 2019) presented a maximum margin-based method that penalizes similar samples that are classified differently. While methods like (Carnevali et al., 2021; Zhang et al., 2019) utilize graph representation of the data as well as pairwise similarities, a network-flow based approach, which is a closely related area, has never been utilized in PU learning.

Hochbaum's Normalized Cut or HNC (Hochbaum, 2010) has been used in binary classification, where labeled samples from both classes are given. It was shown to be competitive in many applications (Baumann et al., 2019; Spaen et al., 2019; Yang et al., 2014b). In this work, we devise a variant of HNC for PU learning, called *2-HNC*. We compare 2-HNC to the following benchmarks: *uPU* (Du Plessis et al., 2014), *nnPU* (Kiryo et al., 2017) and *PU ET* (Wilton et al., 2022). uPU and nnPU are selected as standard PU learning benchmarks. nnPU exhibited competitive performance consistently, mostly on image and text data. PU ET, a recent state-of-the-art method, demonstrated leading performance, particularly on tabular data where it outperformed deep learning models. Similar to most PU methods, the fraction of positive samples in the data, or π, is given as a prior information for 2-HNC and the benchmark methods.

3 PRELIMINARIES, NOTATION AND HNC

3.1 Notations

Given a dataset V with a set of positive labeled samples L^+ and a set of unlabeled samples U, which is a mixture of positive and negative samples, the goal is to predict the label, or class, of each sample in U. We formalize the PU-learning task as a graph problem.

Let the directed graph $G = (V, A)$ represent the data with V, the set of vertices that corresponds to samples in the data, and $A = \{(i, j)|i, j \in V, i \neq j\}$ the set of arcs that connect each sample pair. Arcs (i, j) and (j, i) that connect i and j carry the same capacity weight w_{ij}, which reflects the symmetry of pairwise similarity of i and j.

3.2 Hochbaum's Normalized Cut (HNC)

Given a dataset, with the set of samples V and pairwise similarities w_{ij} for $i, j \in V$, the goal of HNC is to find a partition of V to two non-empty sets S and $\bar{S}$ that optimizes the tradeoff between two objectives: high *intra*-similarity within the set S and small *inter*-similarity between S and its complement $\bar{S}$. We denote their inter-similarity by $C(S, \bar{S})$, defined as $\sum_{i\in S, j\in\bar{S}} w_{ij}$. The intra-similarity within S is defined as $C(S, S) = \sum_{i,j\in S,i<j} w_{ij}$. HNC, with a tradeoff parameter $\mu \geq 0$, is the following problem:

$$\text{(HNC+)}\quad \underset{\varnothing\subset S\subset V}{\text{minimize}}\quad C(S, \bar{S}) - \mu\, C(S, S) \qquad (1)$$

Because of the symmetry between S and $\bar{S}$, the problem can be alternatively presented for the tradeoff between the intra-similarity within $\bar{S}$ and the inter-similarity between it and its complement.

$$\text{(HNC-)}\quad \underset{\varnothing\subset S\subset V}{\text{minimize}}\quad C(S, \bar{S}) - \mu\, C(\bar{S}, \bar{S}) \qquad (2)$$

One might consider a variant of HNC that incorporates both intra-similarities, $C(S, S)$ and $C(\bar{S}, \bar{S})$, as a more generalized version of both HNC+ (1) and HNC- (2). This variant, with two tradeoff weights $\alpha \geq 0$ and $\beta \geq 0$, is given as problem (3) below.

$$\underset{\varnothing\subset S\subset V}{\text{minimize}}\quad C(S, \bar{S}) - \alpha\, C(S, S) - \beta\, C(\bar{S}, \bar{S}) \qquad (3)$$

However, as proved in the next lemma, problem (3) is equivalent to either HNC+ or HNC-, depending on the relative values of α and β.

Lemma 3.1. *Problem (3) is equivalent to HNC+ (1) when* $\alpha \geq \beta$ *for* $\mu = \frac{\alpha-\beta}{1+\beta}$, *and is equivalent to HNC- (2) when* $\alpha < \beta$ *for* $\mu = \frac{\beta-\alpha}{1+\alpha}$.

Proof. $C(V, V)$ is a constant, which we denote by W_V, and is equal to $C(S, \bar{S}) + C(S, S) + C(\bar{S}, \bar{S})$ for any nonempty $S \subset V$. Hence, the objective function of (3) can be written as $C(S, \bar{S}) - \alpha\, C(S, S) - \beta(W_V - C(S, \bar{S}) - C(S, S)) = (1 + \beta)(C(S, \bar{S}) - \frac{\alpha-\beta}{1+\beta} C(S, S)) - \beta W_V$. Minimizing this function is equivalent to solving (1) with the tradeoff $\mu = \frac{\alpha-\beta}{1+\beta} \geq 0$ when $\alpha \geq \beta$.

Alternatively, the objective function of (3) can be written as $C(S, \bar{S}) - \alpha(W_V - C(S, \bar{S}) - C(\bar{S}, \bar{S})) - \beta\, C(\bar{S}, \bar{S}) = (1 + \alpha)(C(S, \bar{S}) - \frac{\beta-\alpha}{1+\alpha} C(S, S)) - \alpha W_V$. Hence, minimizing this objective is equivalent to solving (2) with $\mu = \frac{\beta-\alpha}{1+\alpha} \geq 0$ when $\alpha < \beta$. □

Therefore, instead of solving (3) where the two intra-similarities are shown explicitly, it is sufficient to consider either HNC+ or HNC- depending on whether we put more weight on the intra-similarity of S, or of $\bar{S}$. We note that in prior applications of HNC to binary classification, e.g. (Yang et al., 2014b; Baumann et al., 2019), the model was the one that considered the intra-similarity in S only, as in HNC+.

Applying HNC in binary classification, when labeled samples from both classes are given, the goal is to partition a data that consists of the positive and negative labeled sets, L^+ and L^-, as well as the unlabeled set, U, into S and $\bar{S}$, and predict the labels of unlabeled samples in U accordingly. In previous works, e.g. (Yang et al., 2014b; Baumann et al., 2019), the labeled sets are used as *seeds* and either HNC+ or HNC- is solved with the restriction that $L^+ \subseteq S \subseteq V \setminus L^-$. Unlabeled samples in the optimal S^* and $\bar{S}^*$ are predicted positive and negative, respectively.

4 2-HNC: A TWO-STAGE METHOD FOR PU LEARNING

In this section, we describe the *2-HNC* method where HNC is applied in two stages in PU learning where only the positive labeled set L^+ and the unlabeled set U are given. We then show how the optimization problems in 2-HNC are solved as parametric minimum cut problems on associated graphs.

4.1 2-HNC for PU Learning

The *2-HNC* method consists of two stages. In stage 1, we solve HNC- using only the given positive labeled set. In stage 2, we utilize the likely-negative samples extracted from the unlabeled set based on the result of the first stage, prior to solving HNC+ using both the positive labeled set and the likely-negative set. The output solution is the one data partition, among those that were generated in both stages, that has the fraction of positive samples closest to the ratio π, given as prior.

4.1.1 Stage 1: Solving HNC- with Positive Labeled Samples

The given positive labeled set L^+ is used as the seed set for the set S in HNC+ and HNC-. Since no negative labeled samples are provided, $L^- = \varnothing$, that is, no seed sample is required to be in $\bar{S}$. The seed set constraint imposed on HNC+ and HNC- is then $L^+ \subseteq S$.

Without a seed set for $\bar{S}$, HNC+ is not well defined: the optimal solution to HNC+ is always $(S^*, \bar{S}^*) = (V, \varnothing)$ for any tradeoff $\mu \geq 0$. That is, HNC+ has only the trivial solution in which all unlabeled samples are predicted to be positive. HNC+, however, will be used in stage 2 when likely-negative samples are available.

On the other hand, HNC- , with only positive labeled samples, gives non-trivial data partitions for various values of the tradeoff parameter.

The optimal data partition for HNC- is dependent on the tradeoff μ. We solve HNC-, under the constraint $L^+ \subseteq S$, for all tradeoff $\mu \geq 0$ as a parametric minimum cut problem on an associated parametric graph. For $\mu = 0$, the optimal partition $(S^*, \bar{S}^*)$ is $(V, \varnothing)$. As μ increases, the optimal partition gradually changes, for some μ, until μ reaches a sufficiently large value, at which $(S^*, \bar{S}^*)$ is $(L^+, V \backslash L^+)$. The result of the associated parametric minimum cut problem is a sequence of data partitions: $(S_1^*, \bar{S}_1^*), (S_2^*, \bar{S}_2^*), \ldots, (S_q^*, \bar{S}_q^*)$, that correspond to increasing values of μ. Here, q is the number of different partitions in the parametric minimum cut solution, and can be different for different data. This sequence of partitions for increasing values of μ, in fact, is nested. That is, $\bar{S}_1^* \subseteq \bar{S}_2^* \subseteq \cdots \subseteq \bar{S}_q^*$. We discuss the procedure of solving HNC- as a parametric minimum cut problem, as well as *the nested cut property* in Section 4.2. Stage 1 ends here with the data partition sequence, that is the optimal solution to HNC- for different tradeoff values, as an output.

4.1.2 Stage 2: Solving HNC+ with Positive Labeled Samples and Likely-Negative Unlabeled Samples

Solving HNC- in stage 1 does not require negative labeled samples and gives us, for each tradeoff μ, a partition of data samples into the positive prediction set S^* and the negative prediction set $\bar{S}^*$. However, HNC- only considers the scenario where the intra-similarity of the negative prediction set $\bar{S}$ is given higher importance than that of the positive prediction set S. Here, we consider HNC+, before combining the results from both stages as a final step of 2-HNC.

To handle the issue of HNC+ being ill-defined in the absence of the negative labeled set, as discussed in Section 4.1.1, we add to the problem the seeds for $\bar{S}$. We select the set of samples that are likely to be negative, or L^N, from the unlabeled set U as the seed set for $\bar{S}$. The random sampling procedure to form L^N, called *SelectNeg*, is based on the results of stage 1.

SelectNeg takes as input the sequence of optimal data partitions $(S_1^*, \bar{S}_1^*), (S_2^*, \bar{S}_2^*), \ldots, (S_q^*, \bar{S}_q^*)$, which are the results of solving HNC- for all $\mu \geq 0$, for increasing values of μ, in stage 1. The nested sequence $\bar{S}_1^* \subseteq \bar{S}_2^* \subseteq \cdots \subseteq \bar{S}_q^*$ starts from $\bar{S}_1^* = \varnothing$ and expands until $\bar{S}_q^* = V \backslash L^+$, which is the largest possible since we require L^+ to be in S_q^*. The implication of the nestedness is that, for an unlabeled sample that is predicted negative for a particular μ, it is also predicted negative for any larger value of μ.

We consider unlabeled samples that belong to the negative prediction set $\bar{S}^*$ for small μ as likely to be negative. As μ increases from zero, these samples are predicted negative before other unlabeled samples. Formally, for an unlabeled sample $i \in U$, we denote $q_i = \max\{\gamma \mid i \in S_\gamma^*\}$ as the index of the last partition in the sequence where sample i is still in the positive prediction set. $\eta(i) = |S_{q_i}^*|$ is the number of samples that are predicted negative at the same or larger values of tradeoff μ. A large $\eta(i)$ implies that sample i is more likely to be predicted negative than a large number of samples. In our sampling method SelectNeg, the probability that unlabeled sample i is selected as likely-negative is $\eta(i) / \sum_{j \in U} \eta(j)$. The number of likely-negative samples to be selected, or the size of the set L^N, is chosen in this work to be equal to the number of the positive labeled samples, $|L^+|$.

Once the likely-negative set L^N is formed, we use the positive labeled set L^+ and the likely-negative set L^N as the seed sets for S and $\bar{S}$, respectively, and solve HNC+ with the seed set constraint $L^+ \subseteq S \subseteq V \backslash L^N$. As a result, the output from stage 2 is another sequence of data partitions, which are the optimal solutions to HNC+ for all nonnegative tradeoff μ.

4.1.3 Combining Results from Both Stages

Among all data partitions generated in both stages, we select the partition whose positive fraction, computed as $\frac{|S^*|}{|V|}$ for a partition $(S^*, \bar{S}^*)$, is closest to the prior π. Unlabeled samples in S^* of the selected partition are predicted positive, and those in $\bar{S}^*$ negative.

4.2 Solving Parametric Minimum Cut Problems in 2-HNC

We mentioned in the previous subsection that 2-HNC involves solving HNC+ and HNC- as parametric minimum cut problems on associated graphs. We first explain how the two problems are solved for a single tradeoff $\mu \geq 0$ as minimum cut problems, in Subsection 4.2.1. In 2-HNC, we solve them for all tradeoffs $\mu \geq 0$, prior to selecting one partition from all that are generated. We describe how this is done as parametric minimum cut problems in Section 4.2.2. The nested cut property of the partition sequence as a result of stage 1 is also discussed here.

4.2.1 Solving HNC+ and HNC- for a Tradeoff Parameter μ as a Minimum Cut Problem

HNC+ and HNC- are special cases of *monotone integer programs*, (Hochbaum, 2002; Hochbaum, 2021), and as such can be solved as a minimum cut problem on an associated graph, which is a mapping from the integer programming formulation of both problems (Hochbaum, 2010). This is because any monotone integer programming problem can be solved as a minimum cut problem on an associated graph, the construction of which is a mapping from the formulation (Hochbaum, 2002; Hochbaum, 2021).

Using the standard formulations of HNC+ and HNC-, in the associated graph, there is a node for each sample, and a node for each pair of samples. As a result, the size of this graph is quadratic in the size of the data. However, there are alternative formulations that are "compact", (Hochbaum, 2010), in that the associated graph has number of nodes equal to the number of samples, $|V|$, only. The alternative formulations are shown for HNC+ and HNC- in the following lemma.

Lemma 4.1. *HNC+ is equivalent to the following problem:*

$$\underset{\varnothing \subset S \subset V}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in S} d_i \qquad (4)$$

and HNC- is equivalent to

$$\underset{\varnothing \subset S \subset V}{\text{minimize}} \; C(S,\bar{S}) - \lambda \sum_{i \in \bar{S}} d_i \qquad (5)$$

where $\lambda = \frac{\mu}{\mu+2}$ *and* $d_i = \sum_{j \in V \backslash \{i\}} w_{ij}$ *for* $i \in V$.

Proof. $C(S,S) = \sum_{i,j \in S, i<j} w_{ij} = \frac{1}{2} \sum_{i \in S} \sum_{j \in S \backslash \{i\}} w_{ij}$ since $w_{ij} = w_{ji}$. $\sum_{i \in S} \sum_{j \in S \backslash \{i\}} w_{ij} = \sum_{i \in S} (\sum_{j \in V \backslash \{i\}} w_{ij} - \sum_{j \in \bar{S}} w_{ij}) = \sum_{i \in S} d_i - C(S,\bar{S})$. Hence, $C(S,S) = \frac{1}{2}(\sum_{i \in S} d_i - C(S,\bar{S}))$

We rewrite the objective of HNC+ as $C(S,\bar{S}) - \frac{\mu}{2}(\sum_{i \in S} d_i - C(S,\bar{S})) = (1 + \frac{\mu}{2})(C(S,\bar{S}) - \frac{\mu}{\mu+2} \sum_{i \in S} d_i)$. Hence, HNC+ can be solved by minimizing (4): $C(S,\bar{S}) - \lambda \sum_{i \in S} d_i$, with $\lambda = \frac{\mu}{\mu+2}$. The equivalence of HNC- and (5) can be shown similarly by rewriting $C(\bar{S},\bar{S})$ in HNC- as $\frac{1}{2}(\sum_{i \in \bar{S}} d_i - C(S,\bar{S}))$. □

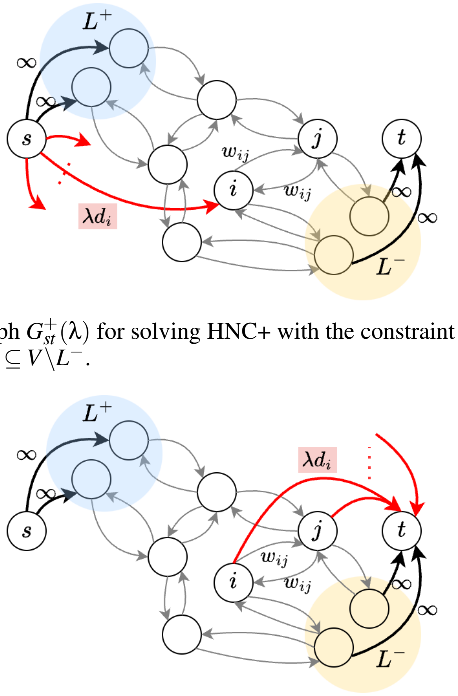
When both labeled sets L^+ and L^- are given, the seed set constraint is $L^+ \subseteq S \subseteq V \backslash L^-$. Under this constraint, the solution to HNC+ for a tradeoff μ, which is now solved via (4) with a tradeoff $\lambda = \frac{\mu}{\mu+2}$, is obtained from the minimum cut solution of the associated graph, $G^+_{st}(\lambda)$. Let $(\{s\} \cup S^*, \{t\} \cup \bar{S}^*)$ denote the minimum cut solution of $G^+_{st}(\lambda)$. Then, $(S^*, \bar{S}^*)$ is the optimal solution to HNC+. The proof provided in (Hochbaum, 2010) is omitted here.

The construction of $G^+_{st}(\lambda)$ for (4), with the constraint $L^+ \subseteq S \subseteq V \backslash L^-$, is illustrated in Figure 1a and described as follows: We add to graph G, described in Section 3.1, source node s and sink node t, and connect s to all nodes of samples in L^+ with arcs of infinite capacity. Similarly, nodes in L^- are connected to t with arcs of infinite capacity. In addition, all unlabeled sample nodes, $i \in V \backslash (L^+ \cup L^-)$, or equivalently $i \in U$, have arcs from s to i of capacity λd_i.

Let $(\{s\} \cup S^*, \{t\} \cup \bar{S}^*)$ be the minimum cut solution of $G^+_{st}(\lambda)$, then we predict unlabeled samples in S^* are positive, and those in $\bar{S}^*$ negative.

HNC- may also be used for binary classification and can be solved similarly, via (5) for a tradeoff $\lambda = \frac{\mu}{\mu+2}$, as a minimum cut problem on the associated graph, $G^-_{st}(\lambda)$, illustrated in Figure 1b. The only difference between $G^+_{st}(\lambda)$ and $G^-_{st}(\lambda)$ is that, in the latter, each $i \in V \backslash (L^+ \cup L^-)$ is connected to t, rather than s, with capacity of λd_i.

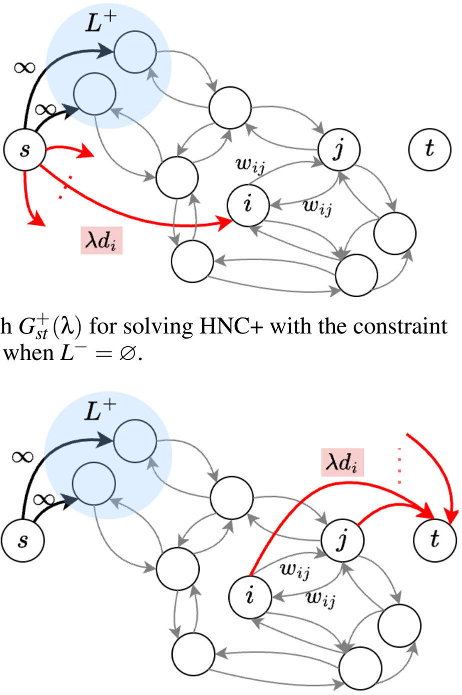
In PU learning, negative labeled samples are not given and therefore $L^- = \varnothing$. HNC+ and HNC- in



(a) Graph $G^+_{st}(\lambda)$ for solving HNC+ with the constraint $L^+ \subseteq S \subseteq V \backslash L^-$.

(b) Graph $G^-_{st}(\lambda)$ for solving HNC- with the constraint $L^+ \subseteq S \subseteq V \backslash L^-$.

Figure 1: Associated graphs with HNC+ and HNC- formulations, when labeled samples from both classes are given. Nodes in the middle, outside the blue and yellow shaded areas, correspond to unlabeled samples in U.



(a) Graph $G^+_{st}(\lambda)$ for solving HNC+ with the constraint $L^+ \subseteq S$, when $L^- = \varnothing$.

(b) Graph $G^-_{st}(\lambda)$ for solving HNC- with the constraint $L^+ \subseteq S$, when $L^- = \varnothing$.

Figure 2: Graphs on which we solve HNC+ and HNC- as minimum cut problems, in PU learning where negative labeled samples are not provided.

this context are then solved, for a tradeoff λ, as minimum cut problems on the graphs in Figure 2a and 2b, which are $G^+_{st}(\lambda)$ and $G^-_{st}(\lambda)$ where $L^- = \varnothing$. As explained in Section 4.1.1, HNC+, with $L^- = \varnothing$, has a trivial solution for all $\lambda \geq 0$. This is also reflected in the minimum cut of $G^+_{st}(\lambda)$ (Figure 2a) with $L^- = \varnothing$, which is $(\{s\} \cup V, \{t\})$, as t is disconnected from other nodes. Hence, in stage 1, we solve only HNC- using the graph $G^-_{st}(\lambda)$ in Figure 2b. Once the likely-negative samples are used as seed samples in stage 2 (Section 4.1.2), HNC+ can be solved using the graph $G^+_{st}(\lambda)$ in Figure 1a.

4.2.2 Solving HNC+ and HNC- for All Tradeoff Values with a Parametric Minimum Cut Procedure

Graphs $G^+_{st}(\lambda)$ and $G^-_{st}(\lambda)$ are *parametric flow networks* in that the capacities of source-adjacent and sink-adjacent arcs ((s,i) and (i,t) for $i \in V$) are monotone non-increasing and non-decreasing with the parameter value (λ), or vice versa. For instance, $G^+_{st}(\lambda)$ in Figure 1a, has source-adjacent capacities that can only increase with λ, and sink-adjacent capacities that are fixed. The minimum cuts in a parametric flow network are solved for all values of the parameter in the complexity of a single minimum cut procedure using the parametric cut (flow) algorithm, (Gallo et al., 1989; Hochbaum, 1998; Hochbaum, 2008). The first is based on the push-relabel algorithm, and the latter two on the HPF (pseudoflow) algorithm.

For our method, 2-HNC, HNC- with no negative seed for $\bar{S}$ ($L^- = \varnothing$) and HNC+ with the seed set $L^- = L^N$ for $\bar{S}$ are solved for *all* nonnegative tradeoff λ in stage 1 and 2, respectively, with a parametric cut procedure.

In stage 1, HNC- with $L^- = \varnothing$ is solved on the parametric graph $G^-_{st}(\lambda)$ in Figure 2b. As explained in Section 4.1.1, the result is a sequence of minimum cuts, or data partitions, for increasing values of μ (and also λ), with the *nestedness* property that motivates how we select likely-negative samples.

Nested Cut Property. *(Gallo et al., 1989; Hochbaum, 1998; Hochbaum, 2008): Given a parametric flow graph $G(\lambda)$, where, as the parameter λ increases, the capacities of the source-adjacent, sink-adjacent and other arcs are non-increasing, non- decreasing and constants, respectively, and a sequence of values $\lambda_1 < \lambda_2 \ldots < \lambda_q$, then the corresponding minimum cut partitions, $(S^*_1, \bar{S}^*_1), (S^*_2, \bar{S}^*_2), \ldots, (S^*_q, \bar{S}^*_q)$,* *satisfy* $\bar{S}_1^* \subseteq \bar{S}_2^* \subseteq \cdots \subseteq \bar{S}_q^*$.

Since the parametric graph $G_{st}^-(\lambda)$, in Figure 2b, is a parametric flow graph, it follows that the nested cut property applies. Let the sequence of partitions according to the parametric minimum cut of $G_{st}^-(\lambda)$ for increasing λ, $\lambda_1 < \lambda_2 \ldots < \lambda_q$, be $(S_1^*, \bar{S}_1^*)$, $(S_2^*, \bar{S}_2^*), \ldots, (S_q^*, \bar{S}_q^*)$. It follows that $\bar{S}_1^* \subseteq \bar{S}_2^* \subseteq \cdots \subseteq \bar{S}_q^*$. This nested data partitions sequence, that is the output of stage 1, is then used in stage 2 (Section 4.1.2) to find the likely-negative set, L^N. An example of the nested sequence is shown in Figure 3.

At the end of stage 2, we obtain the predictions of unlabeled samples by selecting one partition, from all partitions that are generated in stage 1 and 2, whose positive fraction is closest to the prior π.

A general drawback of using minimum cut in very dense graphs is that the solution tends to not favor "balanced" partitions. In a balanced partition, there is a constant fraction $f < 1$ of nodes on one side, and the number of edges between the two sets in the partition is $fn(1-f)n$, which is $O(n^2)$. In that case, even if many edges in the partition have small capacities, their sheer number makes the capacity of such cuts much higher than cuts that contain a small number of nodes on one side. In the graphs we study, all pairwise similarities are evaluated. Therefore, such graphs are complete and dense. The standard approach to obtaining meaningful cut partitions is to apply **graph sparsification**. There are many approaches for graph sparsification in the context of semi-supervised learning, as studied by (de Sousa et al., 2013). Among the approaches evaluated therein, we select the method that was shown to give the best performance, which is the k-nearest neighbor (kNN) sparsification (Blum and Chawla, 2001) where samples i and j are connected only if i is among the k nearest neighbors of j, or vice versa. This results in a graph representation $G = (V, E)$ where E is the set of similar samples according to the kNN sparsification.

5 IMPLEMENTATION OF 2-HNC

This section includes the specification of several implementation details. First, we give a brief description of the parametric minimum cut solver used in this work. Second, we describe the choice of k in the k-nearest neighbor graph sparsification method, as mentioned in the previous section. Finally, we specify the pairwise similarity measure between pairs of samples.

5.1 Parametric Minimum Cut Solver

Solving HNC, via (4) and (5), for all nonnegative tradeoff λ as a parametric minimum cut problem can be done using the pseudoflow algorithm, given by (Hochbaum, 2008) as a *fully* parametric minimum cut solver that identifies all tradeoff values where optimal partitions change as the tradeoff increases. In this work, we use an implementation[1] of the pseudoflow algorithm that is a *simple* parametric minimum cut solver. It takes as input the list of values of λ for which we solve for the minimum cut of $G_{st}^+(\lambda)$ and $G_{st}^-(\lambda)$. The λ values we use are $\{0, 0.001, 0.002, \ldots, 0.500\}$. The simple parametric minimum cut solver finds the minimum cuts for all the listed λ values, efficiently, in the complexity of a single minimum cut procedure.

5.2 Graph Sparsification

As described in Section 4.2.2, we apply the kNN sparsification to $G_{st}^+(\lambda)$ and $G_{st}^-(\lambda)$ on which we solve the parametric minimum cut problem. For each data, we use multiple values of k and find the partitions for all of them prior to selecting one for the prediction. For data of size less than 10000, we use $k \in \{5, 10, 15, 20, 25\}$. For larger data, we use $k \in \{5, 10\}$.

The procedure to select a data partition from those generated by all k's is as follows: For each k, we find the parametric minimum cut on the kNN-sparsified graph and select the partition whose positive fraction is closest to π as the *candidate* partition. Among the candidate partitions from all k, we choose the one with the largest k that has its positive fraction within 2% from π. Larger k is preferred since it maintains more pairwise information. If no candidate partition has positive fraction within 2% from π, we choose the one with the fraction closest to π.

Here, only smaller values of k are evaluated on large data. This is because, as discussed in Section 4.2.2, large datasets, with dense graph representation, often have highly unbalanced cuts. These large datasets benefit from a higher degree of sparsification. Hence, smaller k's are applied.

5.3 Pairwise Similarity Computation

Given H-dimensional vector representations of samples i and j, $x_i, x_j \in \mathbb{R}^H$, we compute their distance d_{ij} as a Euclidean distance between x_i and x_j. The pairwise similarity w_{ij} is then computed using the

[1] https://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html

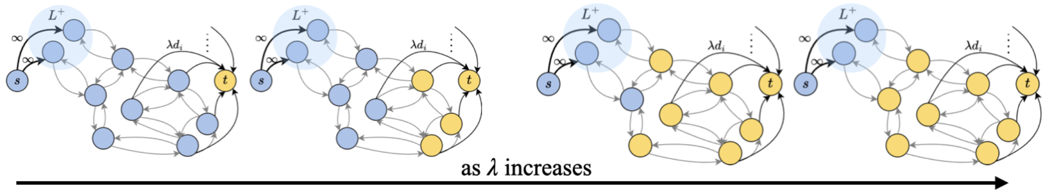


Figure 3: An example of a nested sequence of data partitions as a result of solving the parametric minimum cut problem in stage 1 of 2-HNC, illustrated on the graph $G_{\overline{st}}(\lambda)$. The sets of nodes in blue and yellow are the sets of positive and negative predictions, respectively, for increasing tradeoff values λ.

Gaussian kernel, which is commonly used in methods that rely on pairwise similarities (Jebara et al., 2009; de Sousa et al., 2013; Baumann et al., 2019), as $w_{ij} = exp(-d_{ij}^2/2\sigma^2)$. We use $\sigma = 0.75$ for data with less than 10000 samples. For larger datasets, we use $\sigma = 0.25$. Again, large datasets require a higher degree of graph sparsification. Hence, a smaller σ is applied so that similarities of distant pairs are brought closer to zero, for the same effect as the sparsification technique discussed in Section 4.2.2 and 5.2.

In addition to the standard Euclidean distance, we also use a weighted Euclidean distance as an alternative: $d_{ij} = \sqrt{\sum_{h=1}^{H} \rho_h (x_{ih} - x_{jh})^2}$ where $\rho = [\rho_1, \dots, \rho_H]$ is the weight for the feature vector of size H. ρ is scaled so that $\sum_{h=1}^{H} \rho_h = H$. We use *the feature importance* from a random forest-based PU learning method (Wilton et al., 2022) as the weight ρ. Features with high importance contribute to high impurity reduction at tree node splits in the random forest.

We refer to 2-HNC with the unweighted Euclidean distance as 2-HNC(EU) and the variant with feature importance as 2-HNC(FI).

6 TIME COMPLEXITY ANALYSIS

Let N denote the data size, that is, $N = |L^+| + |U|$. Scikit-Learn implementation using the k-d tree data structure for kNN sparsification and distance computation runs in $O(N \log N)$ (Pedregosa et al., 2011). The similarity weights computation takes $O(N)$ time since there are $O(N)$ pairs remain after sparsification.

The pseudoflow algorithm, known as HPF or Hochbaum's PseudoFlow, solves the parametric minimum cut problem in the complexity of a single minimum cut procedure (Hochbaum, 2008). The complexity of HPF on a graph with n nodes and m arcs, denoted by $T(n,m)$, depends on the implementation. For instance, (Hochbaum and Orlin, 2013) provides a version of HPF that runs in $O(mn \log(\frac{n^2}{m}))$. Since the number of nodes in the graphs of both stages are at most N. The numbers of arcs are at least $2kN$ and at most $4kN$ due to the kNN sparsification. Hence, solving HNC in both stages runs in $O(N^2 \log N)$. This runtime dominates other steps. Therefore, the time complexity of 2-HNC is $O(N^2 \log N)$.

7 EXPERIMENTS

We evaluate 2-HNC with benchmark methods on real data. The test for the methods' robustness against the misspecification of the prior π is also included.

7.1 Datasets

Datasets are listed in Table 1, with the number of all samples, labeled and unlabeled samples, the number of features and the fraction of positive samples (π) of each data. All datasets are from the UCI ML Repository (Kelly et al.,), except for *CIFAR10* from Keras (Chollet et al., 2015), and *20News* and *MNIST* from Scikit-learn (Pedregosa et al., 2011). Samples in each dataset are assigned labels (positive vs negative) as follow: *Vote*: {Democrat} vs {Republican}, *Obesity*: {Obesity Type I, II and III} vs {Insufficient, Normal, Overweight}, *Mushroom*: {Edible} vs {Poisonous}, *20News*: {alt., comp., misc., rec.} vs {sci., soc., talk.}, *Letter*: {A-M} vs {N-Z}, *CIFAR10*: {bird, cat, deer, dog, frog, horse} vs {airplane, automobile, ship, truck}, *MNIST*: {1,3,5,7,9} vs {0,2,4,6,8}. Following (Kiryo et al., 2017; Wilton et al., 2022), we use a pre-trained GloVe word embedding (Pennington et al., 2014) to map each document in *20News* to a 300-dimension vector.

For each dataset, except for *Vote*, we randomly sample 10% of the positive samples (with the number rounded to the nearest hundred) as the positive labeled set L^+. All the remaining samples are used as unlabeled samples, or the set U. For *Vote*, as a small dataset, we randomly select 40 samples as the positive labeled set. We run the experiments 5 times, with different sampling of labeled samples.

As described in the introduction, 2-HNC is a

Table 1: Datasets: 10% of positive samples are randomly selected as labeled samples. The unlabeled set consists of negative samples and the remaining 90% of positive samples. π is the fraction of positive samples in each dataset.

Name	# Samples	# Labeled	# Unlabeled	# Feature	π
Vote	435	40	395	16	0.61
Obesity	2111	100	2011	19	0.46
Mushroom	8124	400	7724	112	0.52
20News	18846	1000	17846	300	0.56
Letter	20000	1000	19000	16	0.50
CIFAR10	60000	3600	56400	3072	0.60
MNIST	70000	3500	66500	784	0.51

Table 2: Classification accuracy (%) average (and standard error) across 5 runs of both variants of 2-HNC and benchmark methods. Number in bold for each data is the highest accuracy.

Data	uPU	nnPU	PU ET	2-HNC(EU)	2-HNC(FI)
Vote	51.60 (1.42)	84.08 (6.80)	92.51 (2.93)	90.33 (0.46)	**94.99** (1.22)
Obesity	85.92 (7.23)	91.92 (1.23)	92.74 (0.66)	89.64 (1.78)	**96.54** (1.18)
Mushroom	87.92 (4.54)	98.94 (0.64)	99.35 (0.66)	99.67 (0.18)	**99.85** (0.09)
20News	58.83 (1.23)	70.90 (0.73)	84.89 (0.41)	76.63 (0.54)	**86.03** (1.46)
Letter	81.33 (1.97)	87.50 (0.76)	86.21 (0.55)	**88.88** (1.51)	87.92 (2.09)
CIFAR10	43.00 (0.01)	**87.98** (0.65)	81.55 (0.11)	78.46 (0.85)	77.81 (0.41)
MNIST	72.65 (1.85)	94.25 (0.91)	95.30 (0.12)	**96.44** (0.07)	94.87 (1.17)

Table 3: F1 score (%) average (and standard error) across 5 runs of both variants of 2-HNC and benchmark methods. Number in bold for each data is the highest F1 score.

Data	uPU	nnPU	PU ET	2-HNC(EU)	2-HNC(FI)
Vote	26.26 (11.81)	88.85 (4.67)	93.20 (3.22)	91.54 (0.40)	**95.63** (1.00)
Obesity	74.59 (13.70)	86.26 (7.75)	91.04 (0.88)	87.94 (2.24)	**94.63** (2.80)
Mushroom	85.19 (6.93)	98.91 (0.66)	99.34 (0.69)	99.67 (0.19)	**99.85** (0.09)
20News	20.34 (4.38)	70.97 (3.39)	85.81 (0.23)	78.55 (0.61)	**87.11** (1.28)
Letter	74.97 (3.26)	85.72 (1.82)	83.49 (0.73)	**88.34** (1.64)	87.42 (2.14)
CIFAR10	21.02 (10.11)	**89.44** (1.19)	83.99 (0.11)	81.05 (0.77)	80.92 (0.49)
MNIST	30.75 (7.96)	93.27 (2.13)	95.11 (0.16)	**96.27** (0.07)	94.63 (1.09)

transductive method that predicts specifically for samples in the given unlabeled set. Hence, we evaluate the models on their predictions of unlabeled samples in U that the models are trained on. The metrics that we use are the classification accuracy and F1 score, averaged over 5 experiments on each dataset.

7.2 Benchmark Methods

2-HNC is compared against the following benchmarks: uPU (Du Plessis et al., 2014; Du Plessis et al., 2015), nnPU (Kiryo et al., 2017) and PU ET (Wilton et al., 2022), as discussed in Section 2.

The choices of neural networks of uPU and nnPU are similar to (Kiryo et al., 2017): a 6-layer MLP with Softsign activation function for *20News*, a 13-layer CNN with a ReLU final layer for *CIFAR10* and *MNIST*, and a 6-layer MLP with ReLU for other datasets. For PU ET, we use the default hyperparameters as suggested in (Wilton et al., 2022). We use the available implementations[2] of these methods.

As explained in Section 5.3, we use two variants of 2-HNC: 2-HNC(EU) and 2-HNC(FI) that use unweighted and feature importance-weighted Euclidean distance, respectively.

7.3 Results

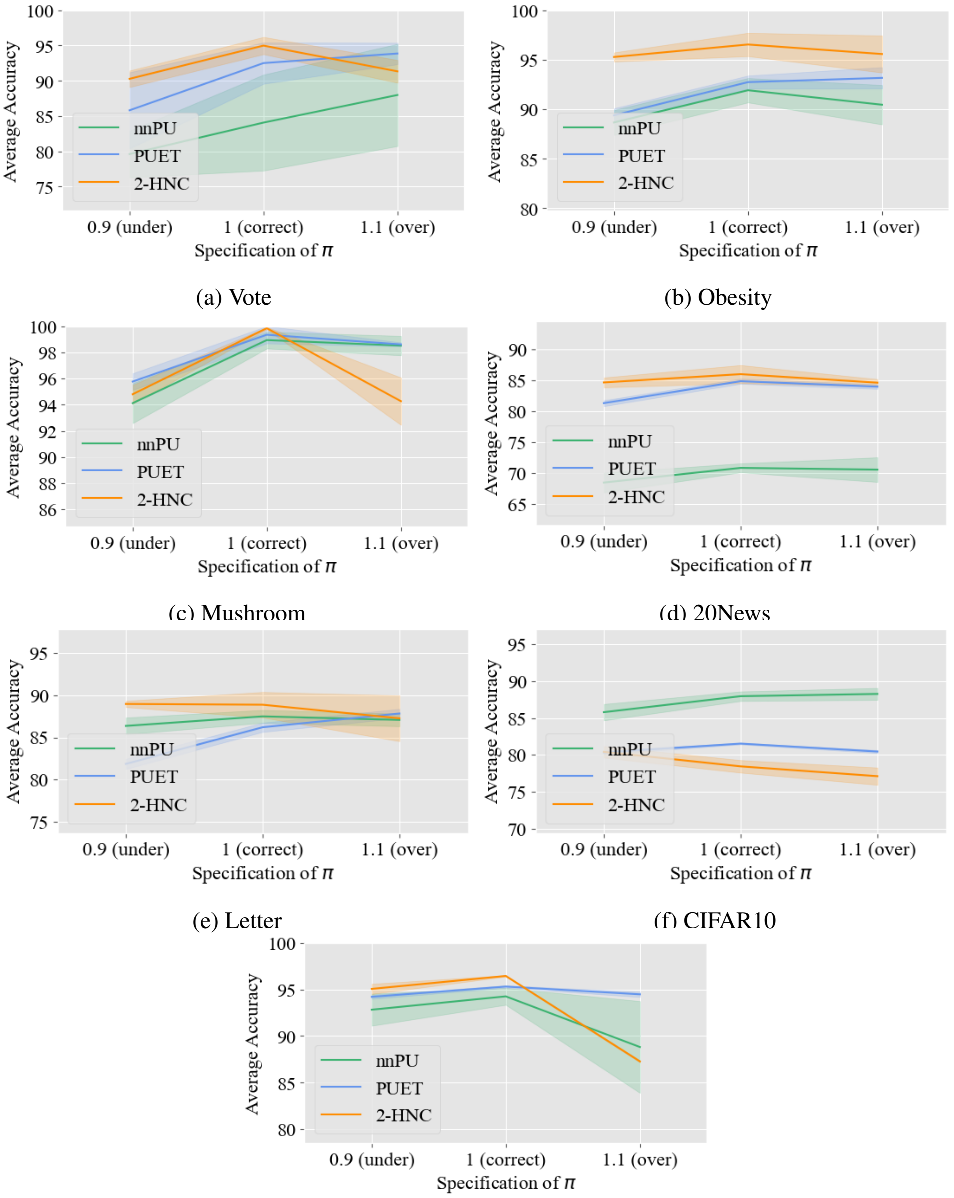
The accuracy and F1 score of both variants of 2-HNC and benchmark models are reported in Table 2 and 3. 2-HNC(FI) yields the best result on tabular data (Vote, Obesity, Mushroom) and the text data (20News). 2-HNC(EU) outperforms all methods on Letter and MNIST. However, nnPU has the best performance for CIFAR10. The relative performance of the models are similar for both accuracy and F1 score.

We also test the statistical significance of the outperformance of 2-HNC over other methods. The best

[2] *uPU, nnPU:*https://github.com/kiryor/nnPUlearning, *PU ET:*https://github.com/jonathanwilton/PUExtraTrees

Table 4: P-values for the t-test on the performances of 2-HNC and the best benchmarks. (*) denotes p-values where HNC outperforms with high statistical significance ($\alpha = 0.05$). P-values on CIFAR10 are not shown as 2-HNC does not give the highest performance on CIFAR10.

Data	Best 2-HNC variant	Best benchmark	P-values: accuracy	P-values: f1 score
Vote	2-HNC(FI)	PU ET	0.0903	0.0875
Obesity	2-HNC(FI)	PU ET	0.0017*	0.0119*
Mushroom	2-HNC(FI)	PU ET	0.0312*	0.0302*
20News	2-HNC(FI)	PU ET	0.1025	0.0465*
Letter	2-HNC(EU)	nnPU	0.0251*	0.0269*
MNIST	2-HNC(EU)	PU ET	1.2584e-5*	8.4961e-5*



(a) Vote

(b) Obesity

(c) Mushroom

(d) 20News

(e) Letter

(f) CIFAR10

(g) MNIST

Figure 4: Average accuracy (with the shaded regions as error bars) of each PU learning method when the prior of the positive fraction π is misspecified, compared to the results when the correct π is provided.

variant between 2-HNC(EU) and 2-HNC(FI), is compared to the best among the three benchmarks for each dataset. P-values of the paired t-tests are reported in Table 4. 2-HNC outperforms other methods with high statistical significance (significance level of 0.05) on most data for both metrics. The exceptions are accuracy and F1 score on Vote, where p-values are 0.0903 and 0.0875, and accuracy on 20News, with p-values of 0.1025. Despite that, these p-values still demonstrate the statistical significance level of around 0.1.

7.4 Sensitivity Analysis

We evaluate the models' sensitivity to the misspecification of the prior of positive fraction π. For each dataset, with true positive fraction π_0, we over-specify and under-specify the prior using $\pi = 1.1\pi_0$ and $\pi = 0.9\pi_0$, respectively. Results are shown in Figure 4 with *uPU* omitted for clarity of the plots as uPU achieves lowest accuracy in all cases. In this analysis, we use the better variant of 2-HNC for each data, according to the result from the previous subsection.

As shown in Figure 4, 2-HNC exhibits higher robustness than other methods when π is under-specified, for all datasets except Mushroom and CIFAR10. For CIFAR10, 2-HNC yields similar performance as PUET, where the two methods have the accuracy of 80.43 ± 0.77 and 80.35 ± 0.25, respectively. Moreover, the rate of accuracy decline for 2-HNC is lower than that of nnPU and PUET on many datasets such as Vote, Obesity, 20News and Letter.

When π is over-specified, 2-HNC is not as robust as other methods. On data such as Obesity and 20News, the improvements become smaller.

8 CONCLUSIONS

Our PU learning method called 2-HNC is a two-stage variant of a network flow-based Hochbaum's Normalized Cut that was previously used in binary classification with labeled samples of both classes. The output of 2-HNC is the partition of samples into the positive and negative prediction sets.

Both stages of 2-HNC generate nested sequences of data partitions for varying tradeoffs between the inter-similarity of the positive and negative prediction sets, and the intra-similarity within sets, solved as parametric minimum cut problems. Stage 1 puts more weights on the intra-similarity of the negative prediction set, whereas stage 2 emphasizes on the positive one. Stage 2 utilizes the set of likely-negative unlabeled samples, determined by the order in which unlabeled samples enter the negative prediction set in the nested sequence of stage 1. A partition whose positive fraction approximates the prior π most closely is selected as the predictions for unlabeled samples.

Experiments on real datasets demonstrate that 2-HNC outperforms benchmark methods in terms of accuracy and F1 scores, as well as better robustness to the under-specification of the prior π.

Future research directions include methods that learn accurate pairwise similarities measure based on the PU data as the current similarity measure is unsupervised. Another potential direction is the selection of likely-negative samples from the unlabeled set. While an approach based on the nested partition sequence is employed in this work, other techniques are also worth further investigation.

ACKNOWLEDGEMENTS

This research was supported in part by the AI Institute NSF Award 2112533. We would also like to thank William Pham for his help with the literature review.

REFERENCES

Asín Achá, R., Hochbaum, D. S., and Spaen, Q. (2020). Hnccorr: combinatorial optimization for neuron identification. *Annals of Operations Research*, 289:5–32.

Baumann, P., Hochbaum, D. S., and Yang, Y. T. (2019). A comparative study of the leading machine learning techniques and two new optimization algorithms. *European journal of operational research*, 272(3):1041–1057.

Bekker, J. and Davis, J. (2020). Learning from positive and unlabeled data: A survey. *Machine Learning*, 109(4):719–760.

Blum, A. and Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts.

Carnevali, J. C., Rossi, R. G., Milios, E., and de Andrade Lopes, A. (2021). A graph-based approach for positive and unlabeled learning. *Information Sciences*, 580:655–672.

Chollet, F. et al. (2015). Keras. https://keras.io.

de Sousa, C. A. R., Rezende, S. O., and Batista, G. E. (2013). Influence of graph construction on semi-supervised learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pages 160–175. Springer.

Du Plessis, M., Niu, G., and Sugiyama, M. (2015). Convex formulation for learning from positive and unlabeled data. In *International conference on machine learning*, pages 1386–1394. PMLR.

Du Plessis, M. C., Niu, G., and Sugiyama, M. (2014). Analysis of learning from positive and unlabeled data. *Advances in neural information processing systems*, 27.

Gallo, G., Grigoriadis, M. D., and Tarjan, R. E. (1989). A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55.

Hochbaum, D. S. (1998). The pseudoflow algorithm and the pseudoflow-based simplex for the maximum flow problem. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 325–337. Springer.

Hochbaum, D. S. (2002). Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *European Journal of Operational Research*, 140(2):291–321.

Hochbaum, D. S. (2008). The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations research*, 56(4):992–1009.

Hochbaum, D. S. (2010). Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):889–898.

Hochbaum, D. S. (2021). Applications and efficient algorithms for integer programming problems on monotone constraints. *Networks*, 77(1):21–49.

Hochbaum, D. S. and Orlin, J. B. (2013). Simplifications and speedups of the pseudoflow algorithm. *Networks*, 61(1):40–57.

Jebara, T., Wang, J., and Chang, S.-F. (2009). Graph construction and b-matching for semi-supervised learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 441–448.

Kelly, M., Longjohn, R., and Nottingham, K.

Khan, S. S. and Madden, M. G. (2014). One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374.

Kiryo, R., Niu, G., Du Plessis, M. C., and Sugiyama, M. (2017). Positive-unlabeled learning with non-negative risk estimator. *Advances in neural information processing systems*, 30.

Lee, W. S. and Liu, B. (2003). Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455.

Li, H., Chen, Z., Liu, B., Wei, X., and Shao, J. (2014). Spotting fake reviews via collective positive-unlabeled learning. In *2014 IEEE international conference on data mining*, pages 899–904. IEEE.

Li, W., Guo, Q., and Elkan, C. (2010). A positive and unlabeled learning algorithm for one-class classification of remote-sensing data. *IEEE transactions on geoscience and remote sensing*, 49(2):717–725.

Li, X. and Liu, B. (2003). Learning to classify texts using positive and unlabeled data. In *IJCAI*, volume 3, pages 587–592. Citeseer.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. (2002). Partially supervised classification of text documents. In *ICML*, volume 2, pages 387–394. Sydney, NSW.

Lu, F. and Bai, Q. (2010). Semi-supervised text categorization with only a few positive and unlabeled documents. In *2010 3rd International conference on biomedical engineering and informatics*, volume 7, pages 3075–3079. IEEE.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Ren, Y., Ji, D., and Zhang, H. (2014). Positive unlabeled learning for deceptive reviews detection. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 488–498.

Spaen, Q., Asín-Achá, R., Chettih, S. N., Minderer, M., Harvey, C., and Hochbaum, D. S. (2019). Hnccorr: A novel combinatorial approach for cell identification in calcium-imaging movies. *eneuro*, 6(2).

Wilton, J., Koay, A., Ko, R., Xu, M., and Ye, N. (2022). Positive-unlabeled learning using random forests via recursive greedy risk minimization. *Advances in Neural Information Processing Systems*, 35:24060–24071.

Yang, P., Li, X., Chua, H.-N., Kwoh, C.-K., and Ng, S.-K. (2014a). Ensemble positive unlabeled learning for disease gene identification. *PloS one*, 9(5):e97079.

Yang, P., Li, X.-L., Mei, J.-P., Kwoh, C.-K., and Ng, S.-K. (2012). Positive-unlabeled learning for disease gene identification. *Bioinformatics*, 28(20):2640–2647.

Yang, Y. T., Fishbain, B., Hochbaum, D. S., Norman, E. B., and Swanberg, E. (2014b). The supervised normalized cut method for detecting, classifying, and identifying special nuclear materials. *INFORMS Journal on Computing*, 26(1):45–58.

Yi, J., Hsieh, C.-J., Varshney, K. R., Zhang, L., and Li, Y. (2017). Scalable demand-aware recommendation. *Advances in neural information processing systems*, 30.

Zhang, C., Ren, D., Liu, T., Yang, J., and Gong, C. (2019). Positive and unlabeled learning with label disambiguation. In *IJCAI*, pages 4250–4256.

Efficient Visualization of Association Rule Mining Using the Trie of Rules

Mikhail Kudriavtsev[1][a], Andrew McCarren[2][b], Hyowon Lee[2][c] and Marija Bezbradica[3][d]
[1]*Centre for Research Training in Artificial Intelligence (CRT-AI), Dublin City University, Dublin, Ireland*
[2]*Insight Centre for Data Analytics, Dublin City University, Dublin, Ireland*
[3]*Adapt Research Centre, Dublin City University, Dublin, Ireland*
mikhail.kudriavtsev@dcu.ie

Keywords: Association Rule Mining, Data Visualization, Trie of Rules, FP-tree, Frequent Pattern Tree, Cognitive Load, Visualization Efficiency, Data Mining Techniques.

Abstract: Association Rule Mining (ARM) is a popular technique in data mining and machine learning for uncovering meaningful relationships within large datasets. However, the extensive number of generated rules presents significant challenges for interpretation and visualization. Effective visualization must not only be clear and informative but also efficient and easy to learn. Existing visualization methods often fall short in these areas. In response, we propose a novel visualization technique called the ”Trie of Rules.” This method adapts the Frequent Pattern Tree (FP-tree) structure to visualize association rules efficiently, capturing extensive information while maintaining clarity. Our approach reveals hidden insights such as clusters and substitute items, and introduces a unique feature for calculating confidence in rules with compound consequents directly from the graph structure. We conducted a comprehensive evaluation using a survey where we measured cognitive load to calculate the efficiency and learnability of our methodology. The results indicate that our method significantly enhances the interpretability and usability of ARM visualizations.

1 INTRODUCTION

Association Rule Mining (ARM) is a popular technique in data mining and machine learning that aims to uncover interesting and meaningful relationships within large datasets (Agrawal et al., 1993). These relationships, expressed as ”association rules,” provide valuable insights for decision-making across various domains, such as market basket analysis, healthcare, and fraud detection (Shaukat Dar et al., 2015). However, ARM can produce a vast number of rules, making it difficult to interpret them effectively. Therefore, effective visualization techniques are crucial to help analysts and domain experts make sense of the discovered rules and extract valuable knowledge.

Current visualization approaches for ARM results struggle with significant limitations when displaying a large number of rules while retaining essential information. Existing solutions often either provide incomplete information, limiting the ability to fully interpret and explore the rules, or produce overly large and cluttered charts that are challenging to navigate (Fister et al., 2023; Jentner et al., 2019; Fernandez-Basso et al., 2019). These limitations result in ineffective information display, hindering the practical utility of ARM in real-world applications where understanding complex patterns quickly and accurately can be essential.

In response to these challenges, we developed a novel visualization technique named the ”Trie of Rules.” Our approach addresses the problem of ineffective information display by capturing a wealth of information and maintaining a manageable size when dealing with large datasets. Additionally, it reveals implicitly hidden insights such as substitute pairs or clusters of rules. The Trie of Rules method is based on an adapted Frequent Pattern Tree (FP-tree) structure, traditionally used to visualize transactions. We propose a novel way to interpret this structure to visualize association rules, making our approach both easy to learn and efficient.

A key aspect of our approach is its efficiency. We designed the Trie of Rules to enable users to complete tasks more quickly and accurately when dealing with complex datasets, while maintaining a learnability level comparable to existing methods.

[a] https://orcid.org/0000-0001-9815-5067
[b] https://orcid.org/0000-0002-7297-0984
[c] https://orcid.org/0000-0003-4395-7702
[d] https://orcid.org/0000-0001-9366-5113

Kudriavtsev, M., McCarren, A., Lee, H. and Bezbradica, M.
Efficient Visualization of Association Rule Mining Using the Trie of Rules.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 72-80
ISBN: 978-989-758-716-0; ISSN: 2184-3228

The main contributions of this paper are as follows:

- **Development of a Visualization Strategy:** We introduce an efficient visualization technique for ARM results that captures extensive information while remaining easy to interpret.
- **Comparison with Popular Methods:** We compared our method with other popular visualization techniques and demonstrated that it outperforms them in terms of efficiency. This was accomplished via a survey with 34 participants, where we measured efficiency and learnability. Our approach allows users to complete tasks more quickly and accurately, while being as easy to learn as existing methods.
- **Confidence Calculation for Compound Consequents:** The Trie of Rules approach introduces a novel property that significantly enhances further exploration of knowledge and increases speed efficiency when examining the ruleset. This feature allows the calculation of Confidence for rules with compound consequents directly from the graph structure, avoiding additional clutter on the plot and making it easier to read and interpret.

This paper is structured as follows: Section 2 provides background information on ARM and related concepts. Section 3 reviews existing visualization methods and their limitations. Section 4 details our proposed Trie of Rules methodology, including the FP-tree background and the visualization approach. Section 5 describes our evaluation methodology, survey construction, and results. Finally, Section 6 summarizes the contributions and suggests directions for future research.

2 BACKGROUND

Association Rule Mining is a data mining technique that aims to discover interesting relationships and patterns within large datasets (Agrawal et al., 1993). The fundamental concepts of ARM include association rules, ruleset, transactions, frequent set, antecedent and consequent, support, and confidence (Geng and Hamilton, 2006; Wu et al., 2010; Luna et al., 2018).

Transactions refer to the records or instances in a dataset, often representing events or actions. In retail, for example, a transaction might correspond to a customer's purchase, where each item bought constitutes a transaction item.

A **frequent set** is a subset of items that frequently occur together in transactions. The identification of frequent sets is a crucial step in ARM, and it involves finding sets of items whose occurrence surpasses a predefined minimum co-occurrence frequency threshold.

An **association rule** is a relationship or pattern that describes the co-occurrence of items in a dataset. It is typically represented as an implication of the form $A \rightarrow B$, where A is the **antecedent** and B is the **consequent**. An example of an association rule could be: *If a customer buys item X, they are likely to buy item Y*.

A **ruleset** is a collection of association rules derived from a dataset. The ruleset provides a comprehensive view of the discovered patterns and relationships within the data. Each rule in the ruleset contributes to the understanding of associations between different items.

Metrics are essential for describing association rules, with support, confidence, and lift being the most popular. However, many other metrics exist as well (Hahsler, 2024). These metrics assess the value of rules in various ways. Crucially, they describe the relationship between the antecedent and the consequent, which means they can only be applied to rules. The exception to this is support, which can also be applied to frequent sequences and is frequently used as a metric for the threshold during the mining process.

3 RELATED WORK

Visualizing ARM results is recognized as a challenging task, as indicated by surveys conducted by (Hahsler and Chelluboina, 2011; Fernandez-Basso et al., 2019; Jentner et al., 2019; Alyobi and Jamjoom, 2020; Menin et al., 2021; Fister et al., 2023). The complexity arises from the need to represent rules visually while considering the multitude of associated metrics and distinguishing between antecedents and consequents, leading to various proposed approaches.

Traditionally, rules are presented as plain tables or text-based methods due to their simplicity and familiarity. However, these methods often fail to effectively convey complex relationships, and there is much room for improvement.

Although various methods exist, they can be classified into three distinct groups: scatter plots, matrix-based methods, and graph-based methods.

The **scatter plot** approach, one of the more basic methods, was introduced by (Jr. et al., 1999). This method employs a two or three-dimensional plot (Ong et al., 2002) to depict rules as dots. Although effective in handling a high number of rules, scatter plots lack insight into the structure of rules, requiring manual examination of the text-based representation of the original dataset.

Matrix-based visualization, as presented by (Hofmann and Buhmann, 2000), places antecedent and consequent sets on axes and displays metric values at their intersections. Despite its efficiency in revealing rule components, it suffers from scalability issues, particularly as the dataset size increases. A more modern implementation is provided by (Varu et al., 2022).

An improvement to the matrix-based approach is the **grouped matrix-based visualization**, as proposed by (Hahsler et al., 2017), which alleviates size concerns by grouping similar rules. However, scalability remains a challenge.

Graph-based visualization, widely employed in ARM (Klemettinen et al., 1994; Rainsford and Roddick, 2000; Buono and Costabile, 2005; Ertek and Demiriz, 2006; Fernandez-Basso et al., 2019; Alyobi and Jamjoom, 2020; Menin et al., 2021), provides a clear representation of rule structures. However, the main problem remains how to show all the items in a rule and distinguish between antecedents and consequents. This problem leads to either excessive size of the plot or low interpretability. Current methods rely on the idea that two types of nodes exist—items and rules. Items that go into (directed edge) the rule are antecedents, and edges that go out of a rule node are consequents.

These three main categories are implemented in popular libraries such as arulesViz for R (Hahsler et al., 2017) and arules for Python (Hahsler, 2023).

In conclusion, existing ARM visualization methods exhibit limitations in terms of scalability, interpretability, and representation of rule structures. The proposed methodology in the next section aims to address these challenges by incorporating FP-tree principles to create a more effective visualization.

4 METHODOLOGY

4.1 FP-tree Background

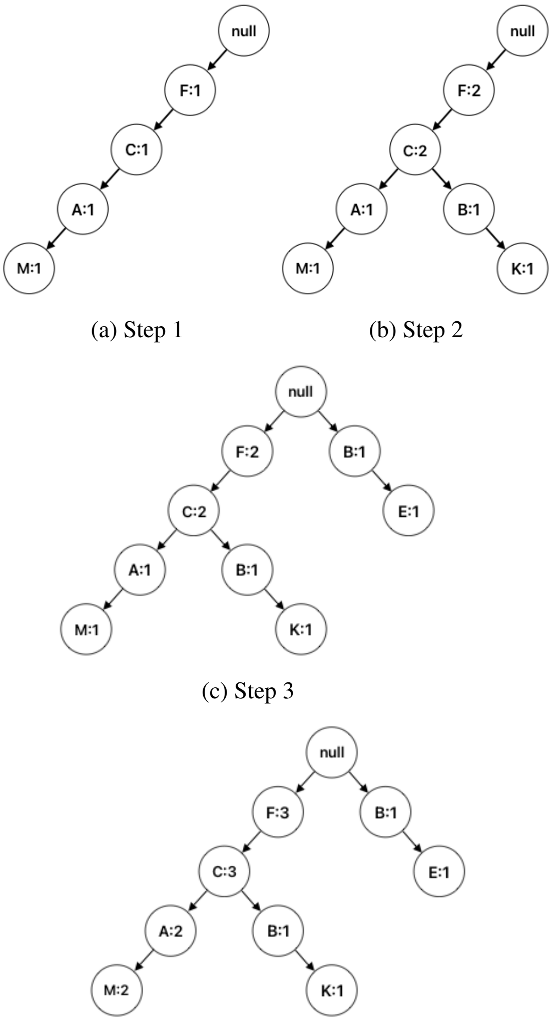
A Frequent Pattern Tree (FP-tree), also known as a **trie** or **prefix tree**, was introduced by (Han et al., 2004). It is commonly used in the rule mining process and is known for its efficiency (Bodon and Rónyai, 2003; Grahne and Zhu, 2003; Shabtay et al., 2021; Shahbazi and Gryz, 2022). This data structure is designed to compactly represent transactions by compressing the database.

An FP-tree is constructed in the following steps:

1. **Scan the Dataset:** The transaction database is scanned to determine the count of each item.
2. **Order Items:** Items in transactions are sorted in descending order of item counts.
3. **Build the Tree:** The FP-tree is built by reading each transaction and mapping it to a path in the tree, ensuring common prefixes are shared to compress the data.

Table 1: Initial Transactions.

Transaction ID	Sorted items
1	F, C, A, M
2	F, C, B, K
3	B, E
4	F, C, A, M



(a) Step 1

(b) Step 2

(c) Step 3

(d) Step 4

Figure 1: Progress of FP-tree construction from transactions in table 1.

Figure 1 demonstrates how the FP-tree structure is dynamically built using transaction from table 1, efficiently representing the frequent itemsets within the dataset.

FP-trees are particularly useful in applications where identifying frequent itemsets is crucial, such as market basket analysis, bioinformatics, and web usage mining. Their ability to efficiently handle large datasets makes them a powerful tool in data mining tasks. However, the potential of this data structure for storing association rules has not been fully explored.

4.2 Proposed Visualization Approach

To leverage the **FP-tree structure for visualizing association rules**, we propose a novel approach called the ”Trie of Rules.” This method adapts the FP-tree to effectively represent association rules, enabling users to comprehend the hierarchical relationships between items and the formation of rules while also reducing the size of the final plot by overlapping rules with common items.

Concept of Rules. In the Trie of Rules, each path from the root (Null node) to a node represents an association rule, where the nodes along the path form the antecedent, and the final node represents the consequent. Figure 2 illustrates the structure of a rule in the Trie of Rules. The item p is depicted as an element that exists in the trie but is not part of the evaluated rule $(f, c, a \to m)$. However, it can potentially become part of another rule. This structure allows users to trace hierarchical relationships between items, enhancing the interpretability and manageability of the visualization of the rules.

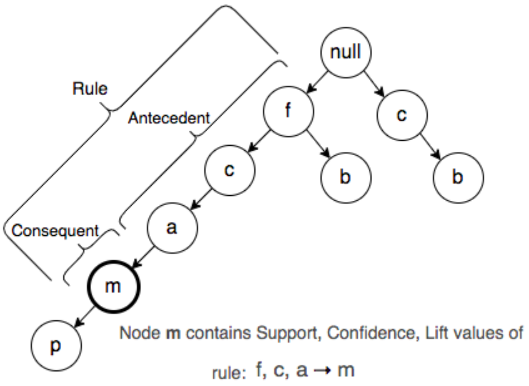


Figure 2: The structure of a rule in a Trie of Rules.

Metrics Display. Metrics are displayed through the color and size of nodes, and optionally, through the size of the caption near nodes. For instance, in Figure 4a, node size captures confidence while node color represents lift, although various other configurations are possible.

Our approach also facilitates the discovery of additional insights, such as clusters and substitute items:

- **Clusters:** Groups of items that frequently occur together can be easily identified through their shared paths in the FP-tree structure, revealing natural clusters within the data.
- **Substitute Items:** Items that can replace each other in transactions are revealed through the overlapping paths in the tree, providing insights into alternative itemsets.

4.3 Confidence for Compound Consequent

A unique feature of our approach is the ability to calculate confidence for rules with compound consequents directly from the graph structure. The confidence of a compound-consequent rule can be calculated as the multiplication of confidence values of the nodes in the consequent, as illustrated in Figure 3.

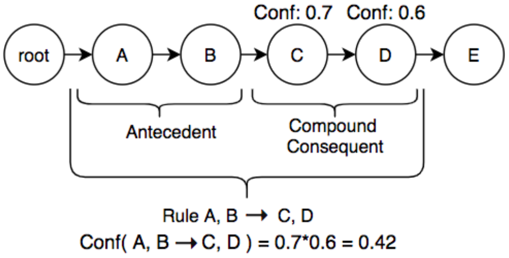


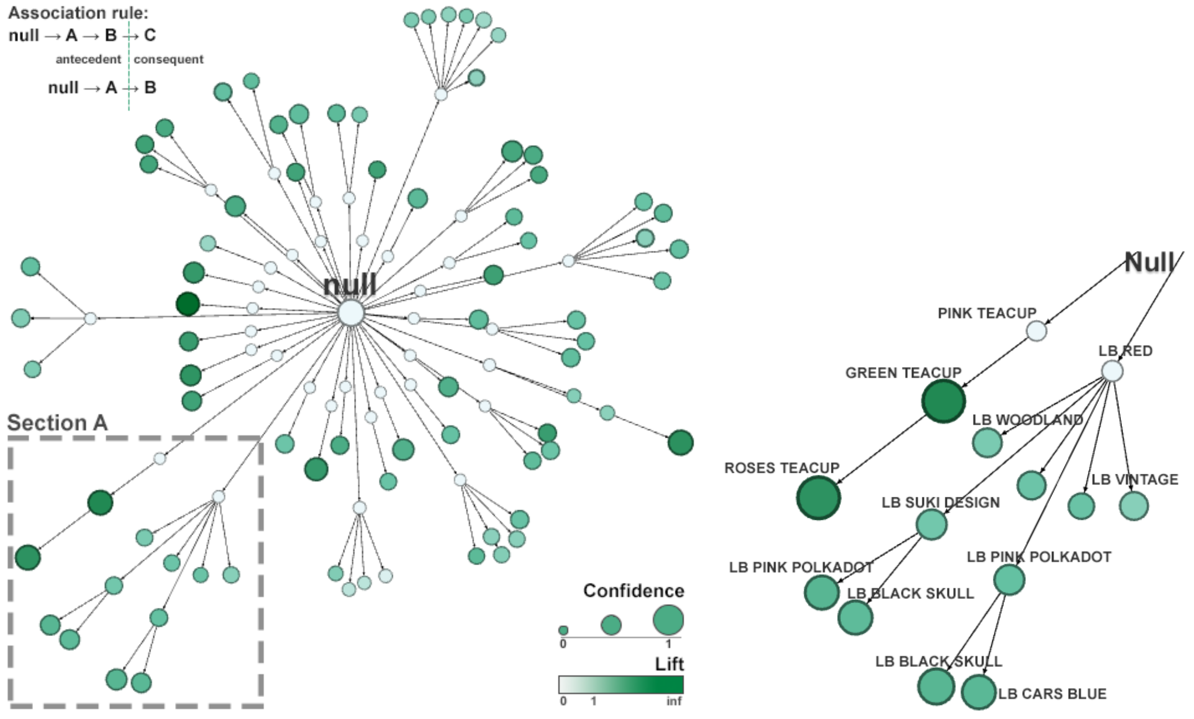
Figure 3: A rule with a compound consequent.

Although this method specifically applies to confidence, the support value for items with a compound consequent does not require additional calculation. The support of a rule $A, B, C \to D$ is equal to the support of the rule $A, B \to C, D$, as both rules refer to the same set of item occurrences within the dataset. Since the support measures the co-occurrence of items, the support for both rules remains the same. However, it is important to note that while the support is identical, the confidence differs. The confidence of $A, B \to C, D$ is based on how often C, D appear given A, B, whereas the confidence of $A, B, C \to D$ is calculated based on how often D appears given A, B, C.

The example rule in Figure 3 is part of a longer path within the trie, but we extract this portion to demonstrate that any section of the path can be taken as a rule. The figure also shows the item E, which exists in the trie but is not part of the current rule.

4.4 Case Study

For the implementation and testing of the Trie of Rules methodology, we used the ”Online Retail Logs”



(a)

(b) Section A

Figure 4: (a) Trie of Rules visualization of the ARM results for the online retail dataset without captions displayed. (b) Zoomed section A of Figure 4a. LB stands for Lunch Bag.

dataset (Chen, 2015). This dataset, characterized by its large size and sparsity, contains 3,663 unique items and 18,484 transactions. The minimum support threshold for the ARM algorithm was set to 0.015, resulting in 234 association rules. We used the FP-growth algorithm (Han et al., 2000) to process the dataset and our developed library (implementation of the Trie of Rules methodology[1]) to produce the graph file.

The resulting Trie of Rules was visualized as a graph structure using Gephi 0.9.2 (Bastian et al., 2009). The default overlay method ”Yifan Hu” (Hu, 2006) in Gephi was applied to enhance the clarity of the visualization.

Figure 4a illustrates the Trie of Rules generated from the Online Retail dataset. The visualization highlights clusters, the hierarchical structure of association rules, and substitute items, providing valuable insights into the dataset.

There are several valuable implications we can draw from exploring Figure 4b:

- The branch that starts with *LB RED* forms various rules that consist solely of Lunch Bag (LB) items of different designs: Vintage, Pink Polkadot, Cars Blue, etc. We can infer that these bags are often bought together in various designs. Based on this, we can propose selling these items as sets. Moreover, sets of color palettes can be formed based on the association rules observed in the Trie of Rules, for example, $(RED, VINTAGE)$ or $(RED, SUKI\ DESIGN, PINK\ POLKADOT)$. Given that *LB RED* starts this branch, we can imply that *LB RED* is the most popular and could be the ”default” item in these sets.
- The branch that starts with *PINK TEACUP* creates several strong rules in the dataset. The color and size of the nodes indicate high Lift and Confidence values. However, this branch forms just two rules:

1. $PINK\ TEACUP \rightarrow GREEN\ TEACUP$
2. $(PINK\ TEACUP, GREEN\ TEACUP) \rightarrow ROSES\ TEACUP$

The first rule is a sub-rule of the second. We can imply that these items are often bought together with high probability. As with the previous branch, we can propose selling these items as sets of various designs. In this case, only one color palette can be proposed: $(PINK, GREEN, ROSES)$.

[1] https://github.com/ARM-interpretation/Trie-of-rules

5 EVALUATION

Evaluating visualization approaches for Association Rule Mining (ARM) is a complex task. Previous studies have employed various methods to assess the effectiveness of visualization techniques:

- Some researchers simply invite one or two experts to provide subjective feedback on their method's effectiveness (Menin et al., 2021; Varu et al., 2022).
- Others demonstrate the utility of their visualization techniques using ”validation through awesome example” (Ong et al., 2002; Leung and Carmichael, 2009).
- Another common approach is to outline the advantages and disdavantages of the proposed methods without conducting rigorous user studies (Fernandez-Basso et al., 2019; Jentner et al., 2019; Hahsler and Chelluboina, 2011; Fister et al., 2023).

However, those methods are not considered as robust enough and objective; literature suggests using more comprehensive evaluation methodologies, such as those described by (Elmqvist and Yi, 2012), emphasising the importance of assessing cognitive load and user efficiency, especially when dealing with complex visualization tasks. Cognitive load refers to the amount of cognitive resources required to perform a task. As highlighted by (Yoghourdjian et al., 2021; Henike et al., 2020; Huang et al., 2009), it provides a quantitative measure to compare the efficiency of different visualization methods, making cognitive load a suitable metric in our study. A conceptual construct of cognitive load in the context of visualization efficiency (Huang et al., 2009) is illustrated in Figure 5.

Our evaluation focuses on measuring efficiency and learnability, similar to the approach used by (Huang et al., 2009). The evaluation process involved a carefully designed survey and tasks, structured as follows.

5.1 Survey Construction

We conducted a survey, which was approved by the ethical committee of [University Name]. The participants, 34 individuals with higher education backgrounds, completed the survey remotely on their own computers. We utilized the LimeSurvey platform to collect their responses and to record the time taken to answer each question. Participants were informed that their response times were being tracked.

Although the survey was anonymous, we ensured a diverse pool by using surveyswap.io, limiting potential participants to those with higher education in technical fields. Additionally, 14 participants were second-year computer science students from [University Name], consisting of 9 females and 5 males. This approach provided a balanced demographic, enhancing the robustness and interpretability of the results.

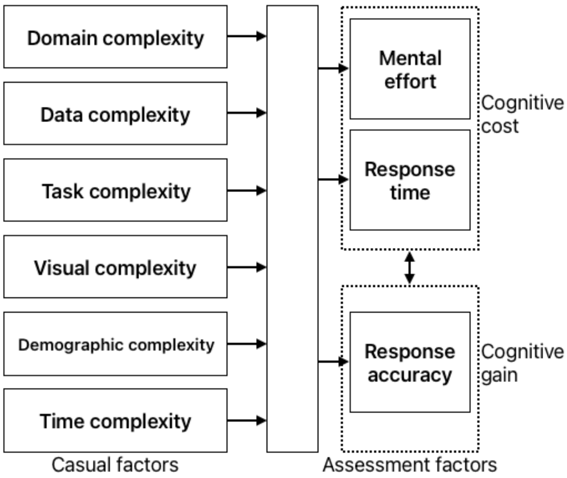


Figure 5: The construct of cognitive load for visualization understanding.

The survey took approximately 50 minutes for each participant and included four sections, one for each type of visualization: scatter plot, matrix-based, graph-based, and our proposed Trie of Rules approach. The sections were presented in a random order for each participant. At the beginning of the survey, participants were given a short introduction to ARM to ensure they could perform the given tasks.

Each section contained 9 questions:

- One introductory question to assess the ease of understanding the visualization method on a scale from 1 to 10, measuring learnability.
- Four simple questions focusing on tasks such as finding the support or confidence of a rule and identifying the rule with the maximum support or confidence.
- Four complex questions requiring deeper analysis, such as determining relationships between rules, identifying substitute items, assessing clusters, counting rules with a specific item, and finding the longest rule.

Participants were not limited in time and were asked the same questions across different visualization methods but with varying items to ensure consistency.

5.2 Measured Metrics

The following metrics were measured to evaluate the effectiveness of the visualization techniques:

- **Response Time (RT):** The time taken to complete each task. Shorter response times indicate more efficient visualizations.
- **Response Accuracy (RA):** The correctness of the answers provided. Higher accuracy indicates more effective visualizations.
- **Mental Effort (ME):** Self-reported effort on a scale of 1 to 10. Lower mental effort suggests that the visualization is easier to understand and use.

To standardize the results and facilitate a fair comparison across different visualization methods, we calculated z-scores for these metrics following the methodology proposed by (Huang et al., 2009). The z-score transformation normalizes the data by subtracting the mean and dividing by the standard deviation of the respective metric, resulting in a standardized score with a mean of 0 and a standard deviation of 1. The formula for calculating the z-score is:

$$z = \frac{X - \mu}{\sigma}$$

where X is the raw score, μ is the mean of the scores, and σ is the standard deviation.

We used the following formula for visualization efficiency:

$$E = Z_{RA} - Z_{ME} - Z_{RT}$$

In this formula, E represents the efficiency via cognitive load, Z_{RA} is the z-score for response accuracy, Z_{ME} is the z-score for mental effort, and Z_{RT} is the z-score for response time. This metric captures the trade-off between accuracy, effort, and time, providing a comprehensive measure of visualization efficiency. High efficiency is achieved when high accuracy is associated with low mental effort and short response time.

5.3 Survey Results and Analysis

The results of our evaluation are summarized in Table 2 and Table 3.

In terms of accuracy, the Trie of Rules method demonstrated better performance on complex questions (0.59) compared to the other methods (Matrix: 0.17, Graph: 0.29, Scatter: 0.23). This indicates that while the Trie of Rules may be novel and less familiar to users, its structured representation of association rules enables more accurate analysis of complex relationships. However, for simple questions, the accuracy of the Trie of Rules (0.44) was on par with the Scatter plot (0.44) and better than the Matrix (0.34) and Graph (0.20) methods. This suggests that while the Trie of Rules is effective for both simple and complex tasks, its advantage becomes more pronounced with increased complexity.

Regarding mental effort, all methods showed no significant difference, as indicated by the ANOVA test results (p-value < 0.05). This indicates that the complexity of the questions impacted time and accuracy rather than mental effort. The Scatter plot required the least effort (2.57), probably because it is the most familiar and commonly used scientific visualization method. The Trie of Rules method showed moderate mental effort (3.11), indicating that while it is a novel approach, it is not significantly more challenging to understand and use compared to existing methods.

The response time for simple questions was slightly higher for the Trie of Rules (56 seconds) compared to the other methods, with the Scatter plot being the fastest (40 seconds). This suggests that users may need more time to familiarize themselves with the Trie of Rules. However, for complex questions, the Trie of Rules (35 seconds) performed on par with the Scatter plot (35 seconds), indicating that once users become familiar with the method, they can analyze complex information just as quickly as with more traditional methods.

5.4 Discussion

The results indicate that the Trie of Rules method offers a significant advantage in terms of accuracy and efficiency, particularly for complex questions, while maintaining a moderate mental effort comparable to existing methods.

The slightly higher response time for simple questions indicates that there is a learning curve associated with the Trie of Rules. This could be due to its novel representation compared to more familiar visualization methods like the Scatter plot. However, the improved accuracy and efficiency for complex questions highlight the potential benefits of this method, especially in scenarios where users need to analyze intricate relationships within the data.

Furthermore, the findings suggest that the benefits of the Trie of Rules may become more apparent with larger datasets and more complex association rules. Future studies could explore the impact of different dataset sizes and structures on the effectiveness of the Trie of Rules. For instance, with twice the number of data points, the advantages of the Trie of Rules in handling complex information efficiently might be even

Table 2: Means of response time, accuracy, mental effort, and efficiency on simple questions.

	Trie of Rules	Matrix	Graph	Scatter
Time (sec.)	56.00	43.00	73.00	40.00
Accuracy	0.44	0.34	0.20	0.44
Effort	3.11	3.32	3.03	2.57
Efficiency	0.23	-0.46	-2.76	2.99

Table 3: Means of response time, accuracy, mental effort, and efficiency on complex questions.

	Trie of Rules	Matrix	Graph	Scatter
Time (sec.)	35.00	40.00	46.00	35.00
Accuracy	0.59	0.17	0.29	0.23
Effort	3.11	3.32	3.03	2.57
Efficiency	1.89	-1.99	-1.56	1.66

more pronounced.

Overall, the Trie of Rules method demonstrates promising potential for enhancing the interpretability and usability of ARM visualizations. By offering a structured and efficient way to represent association rules, it can help users uncover hidden patterns and relationships within large datasets, ultimately facilitating better decision-making and knowledge discovery. Future work will focus on developing software tools to facilitate the adoption of this methodology and further optimizing the user interface and experience to improve the efficiency of the visualization process.

6 CONCLUSION

Association Rule Mining is a valuable technique for uncovering hidden patterns in large datasets, and the efficiency of individuals interpreting these results is greatly influenced by the effectiveness of the visualization techniques employed. Existing visualization methods often struggle with scalability, interpretability, and the effective representation of rule structures, limiting their practical utility in real-world applications.

In this paper, we introduced a novel visualization technique called the ”Trie of Rules.” This method leverages the FP-tree structure to compactly and effectively represent association rules, addressing the common issues faced by traditional visualization approaches. Our approach not only captures a wealth of information and reveals implicit insights, such as clusters and substitute items, but also maintains manageable visualization size by overlapping common items.

We conducted a comprehensive evaluation to compare the Trie of Rules with existing visualization methods through a survey measuring cognitive load. The results demonstrated that our method outperforms others in terms of efficiency, particularly in handling complex queries, while maintaining comparable learnability.

Our findings indicate that the Trie of Rules method significantly enhances the interpretability and usability of ARM visualizations. Future work will focus on developing software tools to facilitate the adoption of this methodology and further researching how user interface and user experience can be optimized to improve the efficiency of the visualization process.

REFERENCES

Agrawal, R., Imieliński, T., and Swami, A. (1993). Mining Association Rules Between Sets of Items in Large Databases. In *ACM SIGMOD Record*, volume 22, pages 207–216.

Alyobi, M. A. and Jamjoom, A. A. (2020). A visualization framework for post-processing of association rule mining. *International Journal Transaction on Machine Learning and Data Mining*, 2020(2):83–99.

Bastian, M., Heymann, S., and Jacomy, M. (2009). Gephi: An open source software for exploring and manipulating networks.

Bodon, F. and Rónyai, L. (2003). Trie: an alternative data structure for data mining algorithms. In *Mathematical and computer modelling*, volume 38, pages 739–751.

Buono, P. and Costabile, M. F. (2005). *Visualizing Association Rules in a Framework for Visual Data Mining*, pages 221–231. Springer Berlin Heidelberg, Berlin, Heidelberg.

Chen, D. (2015). Online Retail. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5BW33.

Elmqvist, N. and Yi, J. S. (2012). Patterns for visualization evaluation. *ACM International Conference Proceeding Series*, (October 2012).

Ertek, G. and Demiriz, A. (2006). A framework for visualizing association mining results. In *Computer and*

Information Sciences – ISCIS 2006, pages 593–602. Springer Berlin Heidelberg.

Fernandez-Basso, C., Ruiz, M. D., Delgado, M., and Martin-Bautista, M. J. (2019). A comparative analysis of tools for visualizing association rules: A proposal for visualising fuzzy association rules. In *Proceedings of the 11th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT 2019)*, pages 520–527. Atlantis Press.

Fister, I., Fister, I., Fister, D., Podgorelec, V., Fister, I., and Salcedo-Sanz, S. (2023). A comprehensive review of visualization methods for association rule mining: Taxonomy, challenges, open problems and future ideas. *Expert Systems with Applications*, 233(June):120901.

Geng, L. and Hamilton, H. J. (2006). Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38(3):9–es.

Grahne, G. and Zhu, J. (2003). Efficiently using prefix-trees in mining frequent itemsets. *Proc. of the 1st IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, pages 236–245.

Hahsler, M. (2023). ARULESPY: Exploring Association Rules and Frequent Itemsets in Python. (Raschka 2018).

Hahsler, M. (2024). A Probabilistic Comparison of Commonly Used Interest Measures for Association Rules.

Hahsler, M. and Chelluboina, S. (2011). Visualizing Association Rules: Introduction to the R-extension Package arulesViz. Technical Report February.

Hahsler, M., Chelluboina, S., and Hornik, D. (2017). Visualizing association rules: Introduction to the r-extension package arulesviz. *Journal of Statistical Software*, 83(1).

Han, J., Pei, J., and Yin, Y. (2000). Mining frequent patterns without candidate generation. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 29(2):1–12.

Han, J., Pei, J., Yin, Y., and Mao, R. (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87.

Henike, T., Kamprath, M., and Hölzle, K. (2020). Effecting, but effective? How business model visualisations unfold cognitive impacts. *Long Range Planning*, 53(4).

Hofmann, T. and Buhmann, J. M. (2000). Multidimensional scaling and data clustering. In *Advances in Neural Information Processing Systems*, pages 459–466.

Hu, Y. (2006). The Mathematica ® Journal Efficient, High-Quality Force-Directed Graph Drawing. *Methematica Journal*, 10:37–71.

Huang, W., Eades, P., and Hong, S. H. (2009). Measuring effectiveness of graph visualizations: A cognitive load perspective. *Information Visualization*, 8(3):139–152.

Jentner, J., Heitmann, B., and Nagel, W. E. (2019). A survey on visualization for mining association rules and frequent item sets. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(2).

Jr., R. J. B., Agrawal, R., and Gunopulos, D. (1999). Constraint-based rule mining in large, dense databases. In *Proceedings of the 15th International Conference on Data Engineering*, pages 188–197.

Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A. I. (1994). Finding interesting rules from large sets of discovered association rules. In *Proceedings of the Third International Conference on Information and Knowledge Management*, pages 401–407.

Leung, C. K. S. and Carmichael, C. L. (2009). FpViz: A visualizer for frequent pattern mining. *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery, VAKD '09*, (January 2009):30–39.

Luna, J. M., Ondra, M., Fardoun, H. M., and Ventura, S. (2018). Optimization of quality measures in association rule mining: an empirical study. *International Journal of Computational Intelligence Systems*, 12:59–78.

Menin, A., Cadorel, L., Tettamanzi, A., Giboin, A., Gandon, F., and Winckler, M. (2021). ARViz: Interactive Visualization of Association Rules for RDF Data Exploration. In *Proceedings of the International Conference on Information Visualisation*, volume 2021-July, pages 13–20. IEEE.

Ong, K.-h., Ong, K.-l., Ng, W.-k., and Lim, E.-p. (2002). CrystalClear: Active Visualization of Association Rules. *ICDM'02 International Workshop on Active Mining AM2002*, (February):1–6.

Rainsford, C. P. and Roddick, J. F. (2000). Visualisation of temporal interval association rules. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1983, pages 91–96.

Shabtay, L., Fournier-Viger, P., Yaari, R., and Dattner, I. (2021). A guided fp-growth algorithm for mining multitude-targeted item-sets and class association rules in imbalanced data. *Information Sciences*, 553:353–375.

Shahbazi, N. and Gryz, J. (2022). Upper bounds for cantree and FP-tree. *Journal of Intelligent Information Systems*, 58(1):197–222.

Shaukat Dar, K., Zaheer, S., and Nawaz, I. (2015). Association rule mining: An application perspective. *International Journal of Computer Science and Innovation*, 1:29–38.

Varu, R., Christino, L., and Paulovich, F. V. (2022). ARMatrix: An Interactive Item-to-Rule Matrix for Association Rules Visual Analytics. *Electronics (Switzerland)*, 11(9).

Wu, T., Chen, Y., and Han, J. (2010). Re-examination of interestingness measures in pattern mining: A unified framework. *Data Mining and Knowledge Discovery*, 21(3):371–397.

Yoghourdjian, V., Yang, Y., Dwyer, T., Lawrence, L., Wybrow, M., and Marriott, K. (2021). Scalability of Network Visualisation from a Cognitive Load Perspective. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1677–1687.

Predicting Post Myocardial Infarction Complication: A Study Using Dual-Modality and Imbalanced Flow Cytometry Data

Nada ALdausari[1] [a], Frans Coenen[1] [b], Anh Nguyen[1] [c] and Eduard Shantsila[2] [d]
[1] *Department of Computer Science, The University of Liverpool, Liverpool, U.K.*
[2] *Institute of Population Health, The University of Liverpool, Liverpool, U.K.*
{*n.al-dausari, coenen, anh.nguyen*}*@liverpool.ac.uk, eduard.shantsila@liverpool.ac.uk*

Keywords: Post-Myocardial Infarction Complications, Flow Cytometry, Artificial Neural Networks, Data Normalisation, Imbalanced Data Handling, Binary Classification.

Abstract: Previous research indicated that white blood cell counts and phenotypes can predict complications after Myocardial Infarction (MI). However, progress is hindered by the need to consider complex interactions among different cell types and their characteristics and manual adjustments of flow cytometry data. This study aims to improve MI complication prediction by applying deep learning techniques to white blood cell test data obtained via flow cytometry. Using data from a cohort study of 246 patients with acute MI, we focused on Major Adverse Cardiovascular Events as the primary outcome. Flow cytometry data, available in tabular and image formats, underwent data normalisation and class imbalance adjustments. We built two classification models: a neural network for tabular data and a convolutional neural network for image data. Combining outputs from these models using a voting mechanism enhanced the detection of post-MI complications, improving the average F1 score to 51 compared to individual models. These findings demonstrate the potential of integrating diverse data handling and analytical methods to advance medical diagnostics and patient care.

1 INTRODUCTION

Cardiovascular Disease (CVD) remains one of the leading causes of mortality (Bhatnagar et al., 2015; Centers for Disease Control and Prevention, 2022), significantly impacting global health trends. Reports from the National Center for Health Statistics highlight that between 2019 and 2021, CVD was a major cause of death in the US (Murphy et al., 2021). Similarly, the British Heart Foundation identifies CVD as more prevalent than cancer in the UK (Bhatnagar et al., 2015), underscoring its severity as a health concern. Among the various types of CVD, myocardial infarction (MI), commonly known as a heart attack, presents particularly complex challenges. It occurs when blood flow to part of the heart is obstructed, resulting in heart muscle damage (Thygesen et al., 2012). Post-MI, patients face significant risks, including heart failure and increased mortality; about 20% of those suffering an acute MI die within the first year, with a substantial portion of these deaths occurring after the initial 30 days (Qing Ye, 2020). This array of adverse outcomes after a MI is collectively referred to as Major Adverse Cardiac Events (MACE) (Clinic, 2022).

Recent medical studies have explored potential predictors for post-MI complications (Boidin et al., 2023; Shantsila et al., 2013; Shantsila et al., 2019), including dynamic changes in specific subsets of white blood cells, particularly those expressing CD14 and CD16 markers. High levels of CD14 and CD16 white blood cells are associated with higher occurrences of MACE, making these counts useful for predicting post-MI complications and managing patient recovery. However, analysing these cells is challenging due to the complexity of interactions and the need for manual calibration in flow cytometry. Additionally, small sample sizes limit the generalisation of findings and focusing solely on cell subsets may overlook other critical factors. These challenges underscore the need for further research and improved methodologies to enhance predictive accuracy and improve patient outcomes.

Recent deep learning efforts have focused on using patient data such as age, gender, lifestyle, and

[a] https://orcid.org/0009-0003-3014-059X
[b] https://orcid.org/0000-0003-1026-6649
[c] https://orcid.org/0000-0002-1449-211X
[d] https://orcid.org/0000-0002-2429-6980

ALdausari, N., Coenen, F., Nguyen, A. and Shantsila, E.
Predicting Post Myocardial Infarction Complication: A Study Using Dual-Modality and Imbalanced Flow Cytometry Data.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 81-90
ISBN: 978-989-758-716-0; ISSN: 2184-3228

isolated biomarker data, typically reflecting the degree of myocardial damage (e.g., troponins) (Mohammad et al., 2022; Khera et al., 2021; Li et al., 2023; Oliveira et al., 2023; Piros et al., 2019; Ghafari et al., 2023; Newaz et al., 2023; Saxena et al., 2022). These studies have incorporated a broad range of features, not solely blood cells, and none have applied convolutional neural networks (CNNs). This paper aims to bridge the gap between traditional medical research and the deep learning community by incorporating white blood cell data into predictive models. We present a deep learning approach to analysing flow cytometry data to predict post-MI complications, addressing two significant technical challenges: the dual modality of the data and the imbalanced nature of the available flow cytometry data. Overcoming these challenges is crucial for enhancing the accuracy of predictions and improving patient outcomes after MI.

The main contributions of this paper are:

1. Developing preprocessing techniques to explore and identify data representations that significantly enhance performance outcomes.
2. Designing and implementing two neural network models to effectively manage the multi-modality inherent in the dataset.
3. Investigating and assessing various balancing techniques to achieve an equitable distribution of samples across different classes.
4. Employing diverse evaluation methodologies to identify the most effective balancing technique, ensuring robust model performance.

2 RELATED WORK

Previous studies have focused on applying machine and deep learning techniques to predict MI mortality and hospital admissions due to complications. These studies, detailed in various research papers, have utilised a range of machine learning algorithms, dataset sizes, and features (Mohammad et al., 2022; Khera et al., 2021; Li et al., 2023; Oliveira et al., 2023; Piros et al., 2019; Ghafari et al., 2023; Newaz et al., 2023; Saxena et al., 2022).

CVD Datasets. Studies on predicting CVD complications have employed many datasets and features to enhance model accuracy. These datasets vary significantly in size, with some studies using smaller datasets of approximately 1,000 to 1,700 patients (Oliveira et al., 2023; Ghafari et al., 2023; Newaz et al., 2023; Saxena et al., 2022). In comparison, others utilised much larger datasets, including those exceeding 100,000 patients (Mohammad et al., 2022; Khera et al., 2021; Li et al., 2023; Piros et al., 2019). Common features across these studies encompass patient demographics such as age and gender, medical history, lifestyle factors, clinical markers like troponin levels, and diagnostic test data such as ECG results (Newaz et al., 2023). Larger datasets typically include more detailed and diverse features, such as in-hospital treatment details and discharge medications. The variety of features used underscores the importance of diverse data in improving the predictive power of machine learning models for CVD complications.

Machine Learning for Post-MI Complications Analysis. Various machine learning algorithms have been employed in these studies, yielding notable successes. Commonly used algorithms include Logistic Regression, Support Vector Machine, Random Forest, XGBoost, and Artificial Neural Networks. Smaller datasets, ranging from 1,000 to 1,700 patients (Oliveira et al., 2023; Ghafari et al., 2023; Newaz et al., 2023; Saxena et al., 2022), often utilised combinations of Support Vector Machine, Logistic Regression, k-nearest neighbours, and Naive Bayes, achieving high accuracy and robust performance metrics. Larger datasets, such as those with over 100,000 patients (Mohammad et al., 2022; Khera et al., 2021; Li et al., 2023; Piros et al., 2019), typically employed more sophisticated algorithms like XGBoost and Artificial Neural Networks, demonstrating their effectiveness with high accuracy and strong performance scores. Overall, XGBoost and Artificial Neural Networks consistently emerged as top-performing models across various studies, highlighting their capability to handle diverse and complex datasets to predict cardiovascular disease complications accurately. These studies emphasise the importance of selecting appropriate algorithms tailored to the dataset size and feature complexity to optimise prediction.

While previous studies have concentrated on employing machine and deep learning models trained on general patient data, this paper diverges by explicitly focusing on blood cell data, mainly white blood cells. White blood cells are pivotal in the context of cardiovascular damage and repair. This focus not only introduces a novel dataset for machine learning applications but also aligns with medical research, as highlighted in previous studies (Shantsila et al., 2011; Shantsila et al., 2019), underscoring the critical role of white blood cells in cardiovascular health. This approach bridges a gap between machine learning methodologies and medical insights, providing a unique perspective on predicting post-MI complication.

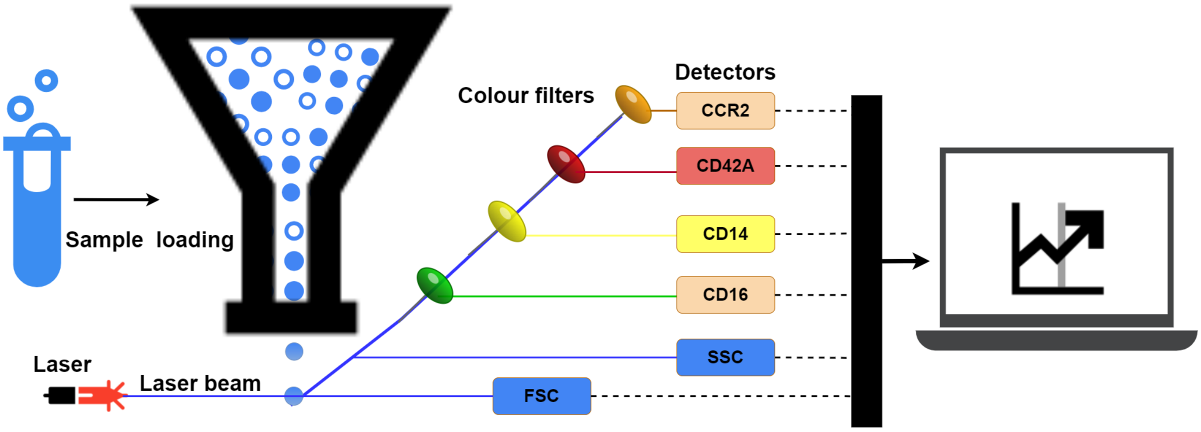


Figure 1: Collect Data by flow cytometry.

3 FLOWCYTO-MI: THE FLOW CYTOMETRY POST-MI COMPLICATION DATASET

3.1 Data Collection

There are multiple risk factors predictive of mortality after the diagnosis of MI, including those based on imaging (echocardiography) and blood tests (troponin levels) (Reddy et al., 2015). This paper focuses on predicting MI complications using blood pathology data from flow cytometry. This technology uses lasers at a sequence of white blood cells moving in a directed fluid stream to generate light signals, causing them to emit light at different wavelengths. Colour filters play a crucial role in this process. They separate the emitted fluorescence light into distinct wavelength bands, allowing only specific wavelengths of light to pass through while blocking others. For example, a filter might permit green light to pass while blocking light of other wavelengths (such as blue or red). Following filtration, the light reaches the detectors, which measure the intensity of the filtered light. Detectors measure the scatter of light and fluorescence emission concerning each cell. Scatter is measured along the laser signal path Forward Scatter (FSC) and at a 90-degree angle to the path Side Scatter (SSC). FSC measures cell size, while SSC measures cell complexity or granularity. The fluorescence helps identify the surface expression (density) of various types of molecules found on the surface of a blood cell. These surface expressions indicate multiple cell functions, labelled by the Cluster of Differentiation (CD) protocol. The data collected by the detectors is then processed and quantified using sophisticated software, converting the raw light intensity measurements into meaningful numerical values. Figure 1 illustrates this process.

The collective effect of these measures is that they allow the separation of individual cells by plotting pairs of features. Figure 2 provides an example of such a plot, generated using FlowJo (FlowJo, 2024), a software system that supports analysing data obtained through flow cytometry. The figure plots FSC density on the x-axis and SSC density on the y-axis. The colours used in the figure indicate cell density: blue and green for low density, red and orange for high density, and yellow for medium density (FlowJo, 2024). The white area in the bottom left corner, which does not have any data, shows electronic noise and tiny particles smaller than cells, thus it is not included in the data collection. The image data is characterised by dimensions of 611 × 620 × 4, denoting the height and width (in pixels) and the RGBA (Red, Green, Blue, and Alpha) values.

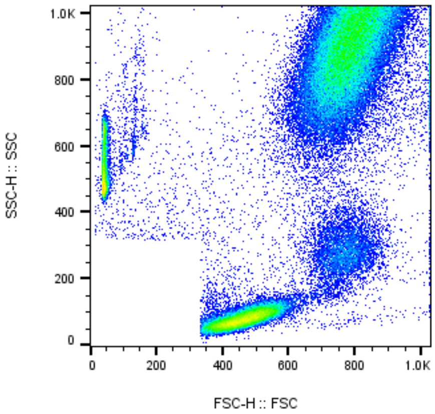


Figure 2: Example density plot (FSC against SSC) generated using Flowjo(FlowJo, 2024).

This paper collected flow cytometry data for 246 patients from several hospitals in Birmingham, UK, including City Hospital, Sandwell General Hospital, Heartlands Hospital, and Queen Elizabeth Hospital, from November 2009 to November 2012. For each patient, the data was provided in two formats: (i) a tabular data file (one line per cell) and (ii) a Portable Network Graphics (PNG) image.

3.2 Data Statistics

In Figure 3, we illustrate the dataset distribution, which includes 195 instances from class 0 (patients without post-MI complications) and 51 instances from class 1 (patients with post-MI complications, heart failure, or death).

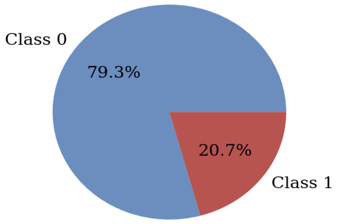


Figure 3: Distribution of dataset.

The tabular data comprised six attributes (columns). The first two were the FSC and SSC values (see section 3.1), and the remaining four were counts of particular surface molecules indicating proteins of various kinds labelled using the CD protocol (CD16, CD14, CD42a, and CCR2)(FlowJo, 2024). Figure 4 shows the range, median, and variability of each feature in the dataset. Features like FSC, SSC, and CD16 AF488 have higher medians and broader distributions, while CD14-PE, CD42a-PerCP, and CCR2-APC show lower medians with significant variability, highlighted by numerous outliers.

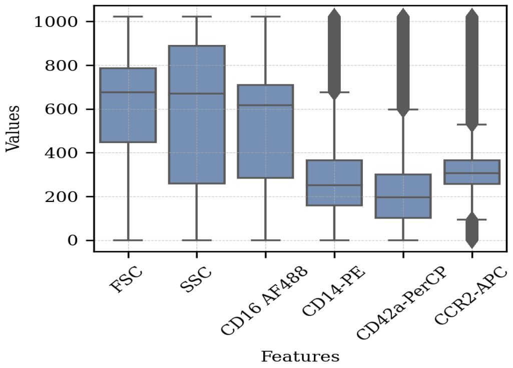


Figure 4: Features Distribution.

The tabular files also varied in length (number of records/rows) because flow cytometry does not always process the same number of cells. Figure 5 shows the median number of rows per patient is similar for both classes, around 100,000, with a slightly larger interquartile range for Class 0. Additionally, there are significant outliers in both classes, with some patients having up to 400,000 rows.

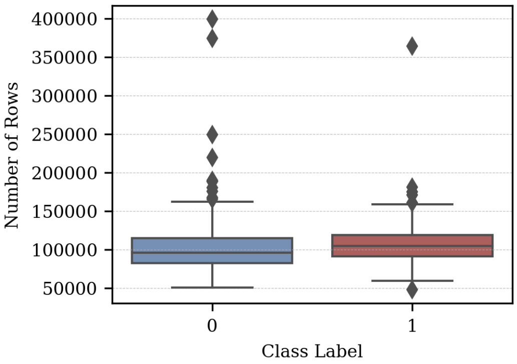


Figure 5: Number of Rows.

4 POST MYOCARDIAL INFARCTION COMPLICATION PREDICTION

The work presented in this paper is directed at using neural networks to predict post-MI complications. The use of neural networks was influenced by the observation that previous research has demonstrated that neural networks are robust and practical techniques for classification (Zhang, 2000). In addition, numerous medical diagnosis applications have shown significant success by utilising neural networks (Zhou and Jiang, 2003).

To build a machine learning model that would work with such dual-modality data, there were two options: (i) use some form of unifying representation and build a single model, or (ii) build individual models, one for each modality and combine the result (for example, by voting). The first was used in the case of Aldosari et al.(2022) in the context of electrocardiogram (ECG) and patient data to predict the likelihood of CVD. This required features to be extracted from each data format to unify the data representation that could be constructed. The disadvantage was that the feature extraction process could result in information loss. When building separate models for each modality, the disadvantage is that it is assumed that each modality is entirely independent of the others when this might not be the case. Given the challenge of extracting features from the blood cell data, the second option was to construct two models. However, in this paper, we tried to avoid the disadvantage of information separation by combining the two models, one for the tabular comma-separated values (CSV) data and one for the image data, with a voting method Algorithm 1.

```
Data: Tabular dataset $T$, Image dataset $I$, Number
      of folds $K = 5$, Tabular model $M_T$, Image
      model $M_I$.
Result: Comprehensive average results with
        statistical significance analysis
Initialise results list $R$;
for fold f from 1 to K do
    Split $T$ and $I$ into stratified training,
     validation, and test sets;
      • Normalise T features using RobustScaler;
      • Normalise I images using ToTensor();
      • Apply data augmentation in I;
    Apply data balancing methods to Train_T and
    Train_I:
      • Use Random Over-Sampling, Random
        Under-Sampling and SMOTE;
      • Use Geometric Transformation, Focal
        Loss and Random Under-Sampling;
    Train Models:
      • Perform hyperparameter tuning for both
        $M_T$ on Train_T and $M_I$ on Train_I, tuning
        parameters such as learning rate, batch
        size, and number of epochs, with cross-
        validation ;
      • Train both $M_T$ on Train_T and $M_I$ on Train_I
        using their respective best hyperparameters;
      • Evaluate $M_T$ on Test_T and $M_I$ on Test_I,
        saving detailed metrics (precision, recall,
        F1-score) to $R_T$ and $R_I$ respectively;
    Combine results $R_T$ and $R_I$:
      • Compute the weighted average of
        predictions based on validation
        performance;
      • If biased, default to class 1;
      • Save combined results $R_f$ with all
        detailed metrics;
    Append $R_f$ to $R$;
end
Calculate comprehensive average results $R$:
  • Average of all metrics (precision, recall, F1-score);
return Comprehensive average results with
  statistical significance analysis.;
```

Algorithm 1: Cross-Validation with Dual Models for Tabular and Image Data.

Data Transformation. Each patient's dataset has a varying number of rows for tabular data but consistently includes six specific columns. We convert the data into a unified tensor format to prepare for neural network processing with PyTorch. This involves: identifying the maximum number of rows (399078), padding shorter sequences with zeros to match this length, converting each DataFrame into a PyTorch tensor, and concatenating these tensors into a master tensor. This results in a tensor format that includes the number of datasets, rows, and columns. For image data, data augmentation techniques enhance the size and quality of training datasets, improving deep-learning models (Yang et al., 2022). The applied transformations include converting to tensors, resizing images to 256x256 pixels, randomly rotating them by up to 20 degrees, and flipping them vertically with a 0.4 probability and horizontally with a 0.5 probability.

Data Splitting. The dataset was divided into a 06% training set, a 20 % validation set and a 20% testing set, following standard practice (Mpanya et al., 2021). Five-fold cross-validation was used for evaluation, partitioning the dataset into five folds and running training and testing five times. Stratified sampling ensured equal class distribution across folds using Python's StratifiedKFold with five splits. This maintains a 60-20-20 split, with about 10 or 11 instances of Class 1 in the test set. Reducing to three folds increases the test set to 17 instances for Class 1 while increasing to seven folds reduces it to five instances. This affects data balance for training and testing, though variations are minor. The data distribution is shown in Table 1.

Table 1: Data distribution in training, validation, and testing sets.

Fold	Training data		Valid. data		Test data		Total
	0	1	0	1	0	1	
1	156	40	39	10	39	11	246
2,3,4,5	156	41	39	11	39	10	246

Data Normalisation. Data normalisation ensures that each attribute contributes equally numerically (García et al., 2015), which enhances classification performance, especially in medical data classification (Jayalakshmi and Santhakumaran, 2011; Singh and Singh, 2020). In tabular flow cytometry data, varying feature ranges required normalisation. The RobustScaler method was applied (Izonin et al., 2022), which uses the median and Interquartile Range (IQR) for scaling, as shown in Equation 1:

$$X' = \frac{X - X_{\text{med}}}{IQR} \tag{1}$$

where X' is the normalised attribute, X_{med} is the median, and *IQR* is the Interquartile Range. In PyTorch, `transforms.ToTensor()` normalizes RGBA values from $[0,255]$ to $[0,1]$ by dividing by 255 and storing the data as a tensor.

Data Balancing. The available data predominantly consists of myocardial infarction (MI) patients without post-MI complications, resulting in an imbalance, with a higher prevalence of patients without complications. The dataset comprises 195 instances from class 0 (no MI complications) and 51 instances from class 1 (MI complications), creating a 4:1 ratio. Various techniques were employed on both tabular and image data to address this imbalance. For tabular data (Zhang et al., 2023; Khushi et al., 2021), random over-sampling generated additional records for the minority class, random under-sampling reduced the majority class records, and SMOTE (Synthetic Minority Oversampling Technique) augmented the minority class using synthetic data created through interpolation. This process involves the existing minority class samples and their nearest neighbours, with K set to 5, to ensure that the number of records in the minority class matches those in the majority class.For image data, balancing strategies involved geometric transformation, random under-sampling similar to the tabular data approach, and focal loss, which modulates cross-entropy loss to focus on minority examples by down-weighting easy examples and emphasising hard-to-classify ones. The weighting factor α is set to 0.80, calculated by the ratio of majority class samples to total samples on the training set, emphasising the minority class. The focusing parameter γ is set to 2, ensuring the model focuses on hard-to-classify examples (Lin et al., 2017). Augmentation conducted through horizontal and vertical axis flipping improves the model's ability to recognise patterns regardless of position. Consequently, two additional images for each image in the minority class could be generated this way.The use of random under-sampling for both tabular and image data effectively reduced the majority class without impacting the minority class, ensuring that all information from Class 1 was preserved, which is critical for accurate modeling. Additionally, with each patient contributing approximately 100,000 rows (see Figure 5), there remained ample data to train the model effectively despite the reduction in the majority class.

Model Generation. This paper addresses the challenge of data's dual-modality by employing two neural network models, each characterised by distinct architectures and design patterns tailored to its data type, with the predictions from both models combined to improve overall performance. The tabular data neural network comprises a standard feed-forward neural network consisting of a sequence of layers organised into four blocks. The first block, `fla_block1`, the flatten layer, transforms the input to 2394468, which is the product of 399078 and 6, the input size. The second block, `lin_block2`, includes linear, batch normalisation, Rectified Linear Unit (ReLU) activation, and dropout layers. The next block, `lin_block3`, is the same as block 1. The last block, `classifier_block4`, includes a linear layer, as shown in Figure 6. Implementation was conducted using the Python PyTorch library. This model employed cross-entropy loss to measure the disparity between predicted class probabilities and the actual class labels. The loss, which falls between 0 and 1, indicates the model's accuracy and aims to minimise it as much as possible (PyTorch, 2024). The model's parameters were also updated during training using the Adam Optimiser, which has a learning rate of 1e-3 and a batch size of 8.

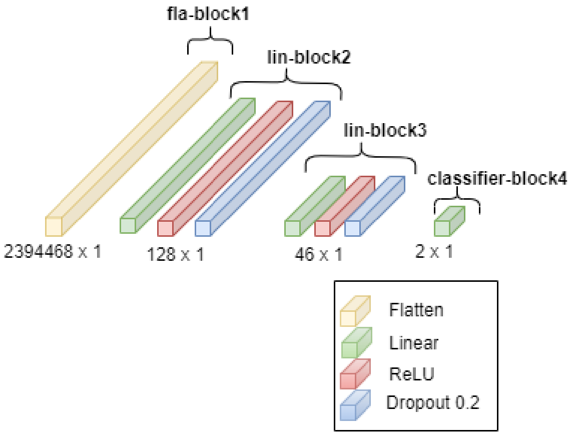


Figure 6: Architecture for the Tabular Data Feed Forward Neural Network.

The image data neural network model was a Convolutional Neural Network (CNN). This model was organised into three components, referred to as `conv_block1`, `conv_block2`, and `classifier_block3`. The first two convolutional blocks comprised several layers, including convolutional layers, batch normalisation, and pooling layers. The role of the `classifier_block3` is to take the output from the convolutional layers, flatten it, and then pass it through a fully connected (linear) layer for making classification predictions. The architecture of the CNN model is illustrated in Figure 7. All input images were resized to (256, 256). The model employs binary cross-entropy loss, typically utilised for binary classification tasks. This loss function compares the predicted logits and target labels. Similar to the previous model, it employs the Adam optimiser.

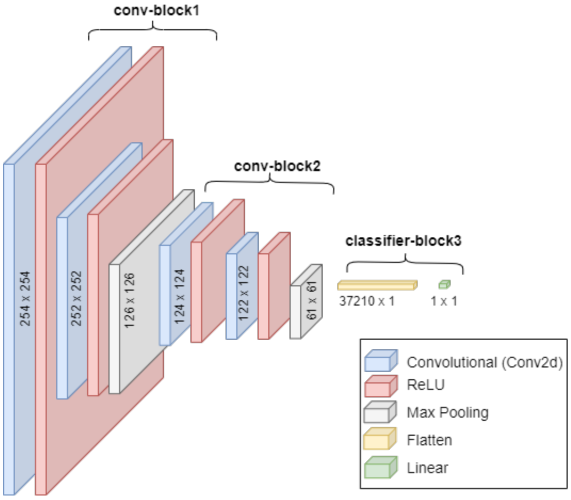


Figure 7: Architecture for the Image Data CNN.

The classifications from these two models were combined using voting (Bin Habib and Tasnim, 2020; Géron, 2017). A straightforward method to enhance the classifier performance is combining the predictions from multiple classifiers and selecting the class with the highest number of votes. In case of a tie-break situation, class 1 was selected, as in medical diagnosis, it is considered more critical to avoid missing an actual illness than to diagnose someone as ill incorrectly. This form of ensemble classifier is known as a complex voting classifier.

5 RESULTS

This section evaluates the proposed process using FlowCyto-MI data. The evaluation metrics adopted were as follows (Zhou, 2020):

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3}$$

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{4}$$

The same data splitting for training, validation, and testing was used in both models, which handle tabular and image data, respectively. The number of data points used in each epoch is shown in Table 1. For each fold, only the test set was used for evaluation, without using any data from the training or validation sets. After experimenting with different epochs, the proposed model's results are presented in Tables 2 and 3.

Note that results are presented using each of the imbalanced data techniques and no technique (Baseline).

Evaluation Data Balancing. Considering the tabular data, an inspection of Table 2 indicates that without data balancing, the tabular model performed exceptionally well for class 0 (patients without post-MI complications), achieving an F1 score of 87. However, class 1 (patients with post-MI complications) recorded an F1 score of 0 for epochs 10 and 3 for epoch 15. All methods, including random over-sampling, random under-sampling, and SMOTE, performed better than the baseline for class 1. For epoch 10, the highest average F1 score, 49, was obtained with SMOTE. For epoch 15, the best average F1 score, 50, was achieved using random over-sampling. This is the best result for this model.

Table 2: Tabular data Feed Forward Neural Network Results, using 10 and 15 Epochs.

Method	Class	10 Epochs			15 Epochs		
(5 folds)		Prec.	Rec.	F1	Prec.	Rec.	F1
Baseline	0	78	97	87	79	97	87
	1	0	0	0	20	2	3
Random over-sampling	0	79	95	86	79	91	85
	1	21	8	11	31	21	15
Random under-sampling	0	79	60	62	78	62	61
	1	18	37	19	27	35	18
SMOTE	0	79	91	85	79	94	86
	1	42	10	13	13	6	8

Abbreviations: Prec.= Precision, and Rec.= Recall

Regarding the image data, the examination of Table 3 reveals that the baseline performance for class 1 was superior compared to that observed with tabular data. At 200 epochs, the highest average F1 scores recorded were 50.5. Methods such as augmentation, random under-sampling, and focal loss demonstrated improvements in the F1 score for class 1 beyond the baseline. Focal loss, applied at epochs 100 and 200, achieved an F1 score of 50.5. Because the average F1 scores are equal in the baseline and with focal loss, we compare based on the recall of class 1. In medical diagnostics, recall is often prioritised over precision as it focuses on the proportion of actual positive cases (patients with the disease) correctly identified by the model. Focal loss at epochs 100 and 200 was selected based on the recall score for class 1.

Evaluation of Combined Model. Table 4 presents the outcomes of model integration, where random over-sampling with 15 epochs was chosen for the tabular model, and focal loss with 100 and 200 epochs was selected for the image model. This setup enabled

Table 3: Image data CNN Results, using 100 and 200 Epochs.

Method	Class	100 Epochs			200 Epochs		
(5 folds)		Prec.	Rec.	F1	Prec.	Rec.	F1
Baseline	0	78	88	83	80	89	84
	1	18	9	12	18	15	17
Augmentation	0	79	64	68	78	74	70
	1	24	41	28	25	31	22
Focal Loss	0	80	71	75	79	74	77
	1	22	33	26	21	27	24
Random	0	77	48	57	78	55	64
under-sampling	1	22	53	30	19	42	26

Abbreviations: Prec.= Precision, and Rec.= Recall

two voting scenarios between these models. In the first scenario, the vote was between the final result from the best tabular model (random over-sampling with 15 epochs) and the final result from the best image model (focal loss and 100 epochs). In the case of a tie, class 1 was selected. In the second scenario, the voting process was identical, except that the image model used focal loss with 200 epochs instead of 100. The best result was achieved using random over-sampling and focal loss with 200 epochs, resulting in an average F1 score of 51. While this represents a slight improvement over the best individual results from the tabular and image models, the difference in performance compared to the CNN model with focal loss and 100 epochs is minimal. Specifically, for Class 0, both models produced nearly identical results (Precision = 80, Recall = 71, F1 = 75), and for Class 1, the difference is very slight, with the CNN model yielding Precision = 22, Recall = 33, and F1 = 26, while our integrated model achieved Precision = 22, Recall = 34, and F1 = 27. A review of Table 4 reveals that the integration strategy achieves the highest F1 score of 51, combining random over-sampling and focal loss with 200 epochs. However, while this integration approach addresses the dual modality and imbalanced nature of the data, the performance improvements are incremental rather than significant when compared to the CNN model alone. Additionally, only 34% of post-MI complications are correctly identified, with 78% of the diagnosed cases being false positives. This raises concerns about the practical applicability of the model in real-world clinical settings, where a high rate of false positives may lead to unnecessary interventions and increased costs. While the integration strategy provides a slight performance boost, further refinement is required to reduce the false positive rate and improve the model's reliability for practical use in diagnosing post-MI complications.

Table 4: Evaluation of Combined Model

Method Combination	Epochs	Class	Prec.	Rec.	F1
Random over-sampling	15	0	80	65	71
& Focal Loss	100	1	23	39	28
Random over-sampling	15	0	80	71	75
& Focal Loss	200	1	22	34	27

Abbreviations: Prec.= Precision, and Rec.= Recall

6 CONCLUSIONS

This paper presented a deep learning approach to predict post-MI complications using dual-modal imbalanced flow cytometry data, consisting of both tabular and image data. Unlike previous studies, which did not utilise blood test data at the individual cell level, our focus was on leveraging this detailed blood cell data for more accurate predictions.

To address the dual-modality issue, we developed two models: one for tabular data and one for image data. The predictions from these models were then combined to produce a final prediction. The best results were achieved using random over-sampling for the tabular data and focal loss for the image data. Our evaluation indicates that the image-based model outperforms the tabular model in predicting post-MI complications. These findings underscore the potential of using detailed blood cell data and advanced modelling techniques to improve prediction accuracy in medical diagnostics.

REFERENCES

Bhatnagar, P., Wickramasinghe, K., Williams, J., Rayner, M., and Townsend, N. (2015). The epidemiology of cardiovascular disease in the uk 2014. 101(15):1182–1189.

Bin Habib, A.-Z. S. and Tasnim, T. (2020). An ensemble hard voting model for cardiovascular disease prediction. In *2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI)*, pages 1–6.

Boidin, M., Lip, G. Y., Shantsila, A., Thijssen, D., and Shantsila, E. (2023). Dynamic changes of monocytes subsets predict major adverse cardiovascular events and left ventricular function after stemi. *Scientific reports*, 13(1):48.

Centers for Disease Control and Prevention (2022). Products - data briefs - number 456 - september 2022. National Center for Health Statistics. Accessed: 2024-03-30.

Clinic, C. (2022). Congestive heart failure. https://my.clevelandclinic.org/health/diseases/17069-heart-failure-understanding-heart-failure.

FlowJo (2024). Flowjo data analysis software. https://www.flowjo.com/solutions/flowjo. February 21, 2024.

García, S., Luengo, J., and Herrera, F. (2015). *Data preprocessing in data mining*, volume 72. Springer.

Géron, A. (2017). *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Sebastopol, CA.

Ghafari, R., Azar, A. S., Ghafari, A., Aghdam, F. M., Valizadeh, M., Khalili, N., and Hatamkhani, S. (2023). Prediction of the fatal acute complications of myocardial infarction via machine learning algorithms. *The Journal of Tehran University Heart Center*, 18(4):278–287.

Izonin, I., Ilchyshyn, B., Tkachenko, R., Greguš, M., Shakhovska, N., and Strauss, C. (2022). Towards data normalization task for the efficient mining of medical data. In *2022 12th International Conference on Advanced Computer Information Technologies (ACIT)*, pages 480–484.

Jayalakshmi, T. and Santhakumaran, A. (2011). Statistical normalization and back propagation for classification. *International Journal of Computer Theory and Engineering*, 3(1):1793–8201.

Khera, R., Haimovich, J., Hurley, N. C., McNamara, R., Spertus, J. A., Desai, N., Rumsfeld, J. S., Masoudi, F. A., Huang, C., Normand, S.-L., Mortazavi, B. J., and Krumholz, H. M. (2021). Use of Machine Learning Models to Predict Death After Acute Myocardial Infarction. *JAMA Cardiology*, 6(6):633–641.

Khushi, M., Shaukat, K., Alam, T. M., Hameed, I. A., Uddin, S., Luo, S., Yang, X., and Reyes, M. C. (2021). A comparative performance analysis of data resampling methods on imbalance medical data. *IEEE Access*, 9:109960–109975.

Li, X., Shang, C., Xu, C., Wang, Y., Xu, J., and Zhou, Q. (2023). Development and comparison of machine learning-based models for predicting heart failure after acute myocardial infarction. *BMC Medical Informatics and Decision Making*, 23(1):165.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Mohammad, M. A., Olesen, K. K. W., Koul, S., Gale, C. P., Rylance, R., Jernberg, T., Baron, T., Spaak, J., James, S., Lindahl, B., Maeng, M., and Erlinge, D. (2022). Development and validation of an artificial neural network algorithm to predict mortality and admission to hospital for heart failure after myocardial infarction: a nationwide population-based study. *The Lancet. Digital health*, 4(1):e37–e45.

Mpanya, D., Celik, T., Klug, E., and Ntsinjana, H. (2021). Predicting mortality and hospitalization in heart failure using machine learning: A systematic literature review. *IJC Heart & Vasculature*, 34:100773.

Murphy, S. L., Kochanek, K. D., Xu, J., and Arias, E. (2021). Mortality in the united states, 2020. *National Center for Health Statistics (NCHS), Data Brief Num. 427*.

Newaz, A., Mohosheu, M. S., and Al Noman, M. A. (2023). Predicting complications of myocardial infarction within several hours of hospitalization using data mining techniques. *Informatics in Medicine Unlocked*, 42:101361.

Oliveira, M., Seringa, J., Pinto, F. J., Henriques, R., and Magalhães, T. (2023). Machine learning prediction of mortality in acute myocardial infarction. *BMC Medical Informatics and Decision Making*, 23(1):1–16.

Piros, P., Ferenci, T., Fleiner, R., Andréka, P., Fujita, H., Főző, L., Kovács, L., and Jánosi, A. (2019). Comparing machine learning and regression models for mortality prediction based on the hungarian myocardial infarction registry. *Knowledge-Based Systems*, 179:1–7.

PyTorch (2024). torch.nn.CrossEntropyLoss. https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html. Febarary 20, 2024.

Qing Ye, Jie Zhang, L. M. (2020). Predictors of all-cause 1-year mortality in myocardial infarction patients. *Medicine*, 99(23).

Reddy, K., Khaliq, A., and Henning, R. (2015). Recent advances in the diagnosis and treatment of acute myocardial infarction. *World Journal of Cardiology*, 7(5):243–276.

Saxena, A., Kumar, M., Tyagi, P., Sikarwar, K., and Pathak, A. (2022). Machine learning based selection of myocardial complications to predict heart attack. In *2022 IEEE 9th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pages 1–4. IEEE.

Shantsila, E., Ghattas, A., Griffiths, H., and Lip, G. (2019). Mon2 predicts poor outcome in st-elevation myocardial infarction. *Journal of internal medicine*, 285(3):301–316.

Shantsila, E., Tapp, L. D., Wrigley, B. J., Montoro-Garcia, S., and Lip, G. Y. (2013). Cxcr4 positive and angiogenic monocytes in myocardial infarction. *Thrombosis and haemostasis*, 109(02):255–262.

Shantsila, E., Wrigley, B., Tapp, L., Apostolakis, S., Montoro-Garcia, S., Drayson, M., and Lip, G. (2011). Immunophenotypic characterization of human monocyte subsets: possible implications for cardiovascular disease pathophysiology. *Journal of Thrombosis and Haemostasis*, 9(5):1056–1066.

Singh, D. and Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, 97:105524.

Thygesen, K., Alpert, J. S., Jaffe, A. S., Simoons, M. L., Chaitman, B. R., and White, H. D. (2012). Third universal definition of myocardial infarction. *circulation*, 126(16):2020–2035.

Yang, S., Xiao, W., Zhang, M., Guo, S., Zhao, J., and Shen, F. (2022). Image data augmentation for deep learning: A survey. *arXiv preprint arXiv:2204.08610*.

Zhang, G. (2000). Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(4):451–462.

Zhang, Y., Kang, B., Hooi, B., Yan, S., and Feng, J. (2023). Deep long-tailed learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9):10795–10816.

Zhou, V. (2020). Precision, recall, and f score concepts in detail. https://regenerativetoday.com/precision-recall-and-f-score-concepts-in-details/. March 20, 2024.

Zhou, Z.-H. and Jiang, Y. (2003). Medical diagnosis with c4.5 rule preceded by artificial neural network ensemble. *IEEE Transactions on Information Technology in Biomedicine*, 7(1):37–42.

GenCrawl: A Generative Multimedia Focused Crawler for Web Pages Classification

Domenico Benfenati[a], Antonio Maria Rinaldi[b], Cristiano Russo[c] and Cristian Tommasino[d]
Department of Electrical Engineering and Information Technology (DIETI), University of Naples Federico II, Naples, Italy
{*domenico.benfenati, antoniomaria.rinaldi, cristiano.russo, cristian.tommasino*}*@unina.it*

Keywords: Web Crawling, Web Pages Classification, Generative AI, Web Topic Analysis.

Abstract: The unprecedented expansion of the internet necessitates the development of increasingly efficient techniques for systematic data categorization and organization. However, contemporary state-of-the-art techniques often need help with the complex nature of heterogeneous multimedia content within web pages. These challenges, which are becoming more pressing with the rapid growth of the internet, highlight the urgent need for advancements in information retrieval methods to improve classification accuracy and relevance in the context of varied and dynamic web content. In this work, we propose GenCrawl, a generative multimedia-focused crawler designed to enhance web document classification by integrating textual and visual content analysis. Our approach combines the most relevant topics extracted from textual and visual content, using innovative generative techniques to create a visual topic. The reported findings demonstrate significant improvements and a paradigm shift in classification efficiency and accuracy over traditional methods. GenCrawl represents a substantial advancement in web page classification, offering a promising solution for systematically organizing web content. Its practical benefits are immense, paving the way for more efficient and accurate information retrieval in the era of the expanding internet.

1 INTRODUCTION

The rapid growth of the web has led to an overwhelming amount of information, with over 5 billion web pages available online (Kunder, 2018; Bergman, 2001). Effective web page classification is crucial for various applications, including information retrieval, content recommendation, and search engine optimization. However, web content's diverse and dynamic nature, including textual and multimedia elements, presents significant challenges for traditional classification methods (Chakrabarti et al., 1999). Previous approaches to web page classification have primarily focused on analyzing textual content, often neglecting the valuable information embedded in visual elements (Rinaldi et al., 2021c; Rinaldi et al., 2021b). While some methods have attempted to incorporate multimedia data, they typically treat textual and visual content separately, failing to leverage the synergistic potential of combining these modalities (Ahmed and Singh, 2019). Furthermore, the complexity and resource-intensive nature of processing large volumes of multimedia content necessitate the development of more efficient and scalable solutions (Fernàndez-Cañellas et al., 2020). Introducing GenCrawl, a genuinely innovative, generative, multimedia-focused crawler, in response to these challenges. This novel approach, which integrates textual and visual content analysis, promises to enhance web page classification accuracy and revolutionize the field. Our work makes some primary contributions: we introduce a novel interdisciplinary approach that combines textual and visual content analysis, significantly improving the classification performance of multimedia-focused web crawlers. This comprehensive approach ensures that no aspect of web content is overlooked, enhancing the accuracy of our classification system. A conventional focused crawler relies on link-based navigation, which may not prioritize systematic data acquisition and processing. This problem limits discerning meaningful patterns, extracting valuable insights, and adapting to dynamic web content evolution. The data generated may lack refinement, potentially leading to lower data quality and hindering analysis accuracy (Kumar and Aggarwal, 2023). Ethical

[a] https://orcid.org/0009-0008-5825-8043
[b] https://orcid.org/0000-0001-7003-4781
[c] https://orcid.org/0000-0002-8732-1733
[d] https://orcid.org/0000-0001-9763-8745

Benfenati, D., Rinaldi, A., Russo, C. and Tommasino, C.
GenCrawl: A Generative Multimedia Focused Crawler for Web Pages Classification.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 91-101
ISBN: 978-989-758-716-0; ISSN: 2184-3228

and privacy considerations may also pose challenges. Adopting a more adaptive framework for web crawling strategies could enhance the effectiveness of conventional focused crawlers. The synergistic integration of deep learning techniques can enhance crawler proficiency in analyzing and categorizing web pages, ensuring seamless incorporation of diverse instances into the evolving web page classification task (K et al., 2023). This approach addresses web page classification intricacies and advances information representation and retrieval methodologies in the expansive and dynamic web-based knowledge dissemination era.

The article is organized as follows: in Section 2 a literature review is presented and discussed, putting in evidence the novelties of our approach; the system architecture and the proposed methodology for crawling strategy and web pages classification methodology are discussed in Section 3; a use case of our crawler together with experimental results are in Section 4; eventually, conclusions and future works are presented in Section 5.

2 RELATED WORKS

General-purpose and special-purpose web agents are the categories into which web crawlers fall (Bhatt et al., 2015). Instead of providing a thorough analysis of general-purpose crawlers, this section highlights relevant research on focused crawlers within the framework of web page classification. Since our framework is based on ontologies, we carefully evaluate works that utilize and conform to this approach. On the other hand, we present a summary of alternative approaches, emphasizing studies that show how well Convolutional Neural Network (CNN) features operate as general feature extractors for multimedia content retrieval tasks.

A focused crawler filters millions of pages and finds relevant resources distant from the initial batch. Machine learning approaches are used to train focused crawlers, which can be extracted from online taxonomies or manually classified datasets (Pant and Srinivasan, 2005). The seminal work on focused crawlers is (Chakrabarti et al., 1999), where the authors developed two hypertext mining software: a classifier for document relevance assessment and a distiller for identifying hypertext nodes providing multiple access points to relevant pages. Moreover, genetic algorithms show intriguing results when it comes to concentrated crawling.

Ontology-based crawlers employ unsupervised ontology learning and domain-based ontology with multi-objective optimization for improved crawling performance and selection of weighted coefficients for web pages (Hassan et al., 2017; Liu et al., 2022; Russo et al., 2020), or for improvement of visualization and document summarization (Rinaldi and Russo, 2021). Event-based crawlers utilize event models and temporal intent recognition methods, including Google Trends data, to capture and prioritize event-related information (Farag et al., 2018; Wu and Hou, 2023). Phishing detection crawlers use isomorphic graph techniques to detect phishing content by identifying subgraph similarities between web pages (Tchakounte et al., 2022). Machine learning crawlers implement LSTM and CNN for word embeddings and classification, and Attention Enhanced Siamese LSTM Networks for predicting web page relevance in specific domains like biomedical information (Shrivastava et al., 2023; Mary et al., 2022). Rule-based and specialized domain crawlers leverage rule-based approaches for obfuscating audio file crawlers in the AIR domain and incremental crawling systems for the Dark Web (Benfenati et al., 2023; Fu et al., 2010). Genetic algorithm-based crawlers utilize modified Genetic Algorithms for web page classification based on keyword feature sets (Fatima et al., 2023). Additionally, an improved genetic algorithm can increase the focused crawler's memory and precision while broadening its search area, focusing on the direct influence of textual content and topic on user information retrieval (Yan and Pan, 2018).

The strategy proposed in this article introduces a multimedia-focused crawler for web page classification, which combines textual and visual topics from text and images as in (Rinaldi et al., 2021a). It uses a supervised learning algorithm to classify web pages, including those with text and images. In the latter case, a generative model extracts and creates images, improving visual topic extraction and crawling performance. The crawler's flexibility and adaptability to different domains make it more adaptable to predefined keywords or exemplary documents. Our study compares a web page classification method using text or images with existing methods, revealing superior accuracy, recall, and greater resilience to noisy or irrelevant content. We also discuss its benefits and drawbacks and suggest future enhancement directions.

3 PROPOSED FRAMEWORK

This section describes in detail how the proposed framework is composed, indicating specifically what the components are and how they function.

Figure 1 shows a sketch of the proposed system architecture. Our system involves multiple modules for crawler tasks. It starts with an online document repository, generating crawler threads. These threads retrieve web pages, analyze structures, classify text and images, and use a model for synthetic image generation if direct comparison isn't possible. Extracted topics from text and images are combined for document classification. The crawling process then selects the next page from the repository. The crawler initialization phase begins by gathering seed URLs from an online document repository and setting up a structured workflow for the subsequent stages of the crawling process. Following this, the web page retrieval and hyperlink extraction phase constructs a network of interconnected pages, creating a comprehensive dataset that mirrors the web's interconnected nature. In the parsing phase, the crawler differentiates between main content (relevant text) and auxiliary content (unrelated text), focusing on essential elements. It also identifies visual information, adding a multimedia dimension to the understanding of web pages.

In the web page classification phase, machine learning models categorize textual and visual elements based on their topics. This classification process automates the evaluation of content relevance, enhancing system efficiency and accuracy. When discrepancies arise between textual and visual classifications, the generative phase harmonizes these interpretations. Using a Latent Diffusion Model (LDM) for image generation, it synthesizes images that align with textual descriptions, ensuring cohesive content representation. The combined topics of textual content and generated images are then used to calculate a priority value for subsequent actions, indicating each web page's significance in the crawler's exploration.

The preprocessing module is crucial for analyzing textual and visual information on web pages, using the DMOZ web collection as a data source. Despite its official closure in 2017, DMOZ remains valuable for classification tasks due to its heterogeneous nature. A screenshot of this repository is available on the Kaggle Dataset platform[1].

The text preprocessing pipeline involves HTML parsing, normalization, tokenization, stopword removal, lemmatization, removal of special characters and symbols, spell-checking, feature extraction, and vectorization. These steps ensure consistent representation, enhance topic extraction efficiency, and prepare the text for analysis.

For encoding the text of web pages, we used SBERT (Cheng et al., 2023) to create vector representations capturing semantic meanings. Clustering techniques grouped sentences into topics based on semantic similarity, extracting representative keywords and generating labels for each topic using a rule-based method. This approach allowed the identification of main themes or concepts in the web pages.

The focused crawler's image processing pipeline involves a sequential process of extracting and analyzing images from web pages, starting with HTML parsing, downloading and preprocessing images for quality enhancement, and extracting relevant features as feature vectors. The critical phase involves applying advanced computer vision techniques, explicitly utilizing the VGG19 model (Simonyan and Zisserman, 2014). VGG19 is a mighty Convolutional Neural Network (CNN) consisting of 19 layers. It has been trained on a large and diverse dataset featuring complex image classification tasks, such as ImageNet (Ridnik et al., 2021), a large image dataset consisting of 14,197,122 images, each tagged in a single-label fashion by one of 21,841 possible classes. VGG19's adaptability makes it well-suited for the task of visual topic extraction in our work, as it excels at discerning patterns and objects in the analyzed images. Figure 2 represents the described textual and visual pipelines.

3.1 Textual Topic Detection

We employ a pipeline to identify representative topics in text using Sentence-BERT (SBERT) embeddings and WordNet synsets. Text is preprocessed through lowercasing, sentence tokenization, and removal of common linguistic artifacts, such as stopwords, special characters, etc, for uniformity. SBERT embeddings encode each sentence into a high-dimensional vector space, capturing contextual relationships. The SBERT application to the preprocessed text ensure that the vector created by the model doesn't effect the noisy informations in the long text of the web pages of the dataset. K-means clustering identifies distinct topics, while WordNet synsets enrich semantic interpretation.

This method's effectiveness relies on the input text's language richness and diversity, facilitating granular content exploration.

3.2 Visual Topic Detection

The visual topic detection task utilizes multimedia components, particularly images, to determine a document's principal topic, enhancing the overall framework's performance. Our research currently focuses on recognizing multimedia descriptors to measure the

[1] https://www.kaggle.com/datasets/shawon10/url-classification-dataset-dmoz?select=URL+Classification.csv

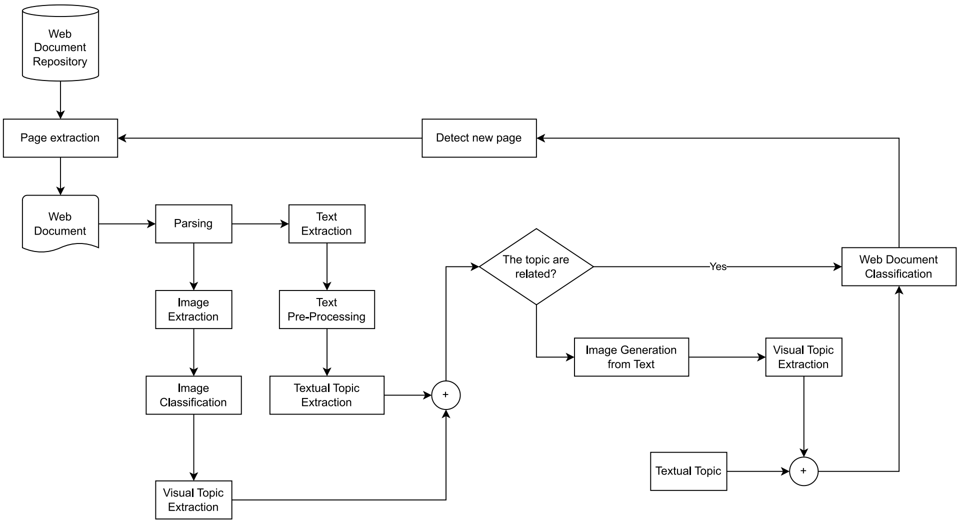


Figure 1: System architecture overview.

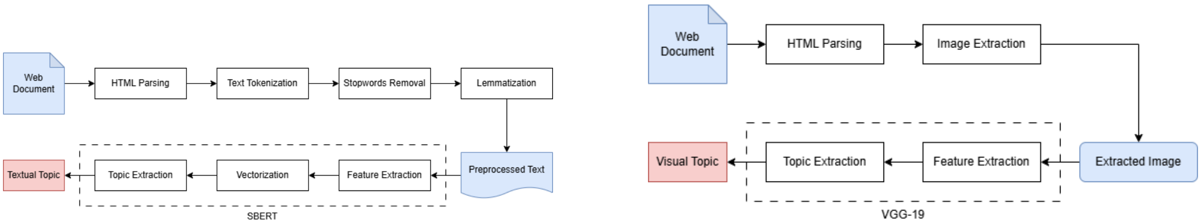


Figure 2: A detailed pipeline for textual (on the left) and visual (on the right) topic extraction and detection task.

similarity between document images and our multimedia knowledge base. Diverse descriptors, including local, global, and deep features, are evaluated (see Section 3). We employ the pre-trained VGG19 model on ImageNet, applied to the complete image, to identify and visualize relevant image regions associated with specific predictions, facilitating efficient transfer learning. This model's straightforward architecture promotes ease of interpretability and implementation.

If more than one image are detected from the web page, we need to select only the images that are more compliant with the textual topic.

3.3 Text-to-Image Generation

We incorporated a Latent Diffusion Model (LDM) into the crawling process to address the challenge of web pages lacking relevant multimedia content. This model generates high-quality images consistent with the textual description of the web page, even if no multimedia data is present initially. The Stable Diffusion (Rombach et al., 2022) serves as the latent model for translating text into images. Based on a latent diffusion approach, it is tailored for generating and manipulating images from textual prompts. The model utilizes the pre-trained CLIP ViT-L/14 text encoder (Radford et al., 2021), following Imagen's methodology (Saharia et al., 2022). The choice of Stable Diffusion over other generation techniques is justified by its superior performance, as demonstrated in Section 3.

To show that the visual topic extraction procedure works in the same way, using generated images, we chose to show the predictions made by the VGG19 model on the example Figure 3 and extract the probabilities for the first three classes predicted by the model. As shown in Table 1, the top 3 predicted classes fully reflect the content of the image, including the topics indicated within the textual prompt used to generate the image.

Table 1: Top 3 Prediction probabilities of generated image using the prompt ”a parrot that rides a bicycle”.

Top Prediction	Probability
macaw	**0.919**
bicycle-built-for-two	0.027
lorikeet	0.017

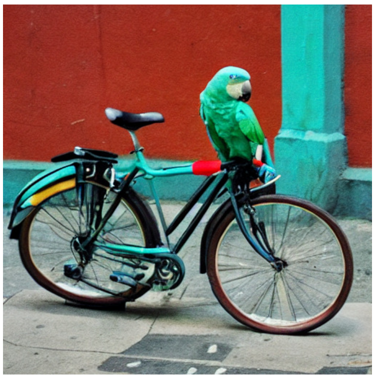


Figure 3: Image generated using the prompt ”a parrot that rides a bicycle”.

3.4 Combined Topic Extraction

This paper aims to improve classification performance by combining two classifiers: one for text-based topic detection and another for visual-based topic detection. Existing research suggests that different classifiers may provide complementary information models based on the specific patterns requiring classification (Mohandes et al., 2018; Clinchant et al., 2011).

The textual and visual classifications are combined by normalizing scores from different topic detection methods and scaling them to fit within the $[0,1]$ interval. The fusion of textual and visual classifications adopts various schemes, and in line with (Rinaldi, 2014), this study opts for the SUM operator and the Ordered Weighted Averaging (OWA) operators proposed by (Yager and Kacprzyk, 2012). These operators provide a systematic way to aggregate the results, allowing for a more robust and comprehensive integration of information from both textual and visual modalities. The SUM function stands out as one of the widely adopted techniques for linear combinations of classifiers, offering various versions such as weighted sum and average. Its prevalence in ensemble methods is attributed to its superior noise tolerance, contributing to enhanced overall performance compared to other elementary functions (Kittler et al., 1998). The versatility of the SUM function makes it particularly effective in capturing the collective decision-making power of diverse classifiers, making it a popular choice in ensemble learning approaches.

Formally an OWA operator of size n is a function $F : R_n \rightarrow R$ with a collection of associated weights $W = [w_1, ..., w_n]$ whose elements are in the unit range such that $\sum_{i=1}^{n} w_i = 1$. The function is defined as:

$$F(a_1, ..., a_2) = \sum_{j=1}^{n} w_j b_j \tag{1}$$

where b_j represent the jth value of the $\vec{a}$ vector ordered.

4 EXPERIMENTAL RESULTS

This section details the experiments conducted to evaluate the performance of key components within the proposed framework, specifically focusing on textual, visual, and combined topic detection strategies. The system outlined in this study exhibits a high degree of generalization owing to the inherent versatility of the developed modules.

4.1 Dataset Description

This study employs a parsed and pre-elaborated multimedia dataset derived from DMOZ (Sood, 2016), a renowned and extensive multilingual web directory recognized for its popularity and open-content richness. DMOZ was selected as the experimental framework to establish a real-world scenario, offering a publicly accessible and widely recognized repository for result comparisons against baseline measures.

The Scraper module compiles URLs for download, using a “*text-only*” parameter to acquire only textual components of each document. The complete DMOZ repository subset is used based on web-scraping policies and link prevalence. It is crucial to map DMOZ categories on WordNet synset and definitions and if a corresponding mapping exists for all categories, because we have to obtain a compliant representation of the synsets and the categories that we have in the dataset. In Table 2, we provide a simple association between the category extracted and the respective WordNet synset tag and the description of it. We use English documents for our experiments, automatically handled with the help of a Python library porting of the Google algorithm for language detection and for the scraping of the pages (Danilak, 2017; Hajba, 2018). Out of a total corpus comprising 12,120 documents, 10,910 were allocated for creating topic modeling models. The remaining 1,210 documents were designated as test sets for the comprehensive evaluation of the entire system. A practical test set for the system requires documents that undergo textual and visual analyses. Specifically, efforts were directed towards selecting random documents from the web directory, ensuring they possess a substantial textual component and a minimum of three images. A DOM Parser algorithm was used to identify and retain a “valid” multimedia document aligned with the system’s objectives.

Table 2: Categories with WordNet Synsets and Definitions.

Category	**Synset**	**Definition**	**Offset**
Arts	art.n.01	The products of human creativity; works of art collectively	2,743,547
Business	commercial_enterprise .n.02	The activity of providing goods and services involving financial and commercial and industrial aspects	1,094,725
Computers	computer.n.01	A machine for performing calculations automatically	3,082,979
Games	game.n.01	A contest with rules to determine a winner	455,599
Health	health.n.01	A healthy state of well-being free from disease	14,447,908
News	news.n.01	Information about recent and important events	6,642,138
Science	science.n.01	A particular branch of scientific knowledge	5,999,797
Shopping	shopping.n.01	Searching for or buying goods or services	81,836
Society	society.n.01	An extended social group having a distinctive cultural and economic organization	7,966,140
Sports	sport.n.01	An active diversion requiring physical exertion and competition	523,513

4.2 Textual Topic Detection

The process of annotating the DMOZ category shown in Table 2 makes the classification using the selected algorithms easier for comparison of the resulted topic detection task because they return no information about the DMOZ category but only about the number of topics that represent the main topic of the text corpus of the web page.

To ensure a comprehensive evaluation of our system's performance, we compare two established reference algorithms widely employed in topic detection research: Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). Latent Semantic Analysis (LSA) (Landauer et al., 1998) employs a vectorial representation approach to capture the essence of a document using the bag-of-words model. Meanwhile, Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a text-mining model rooted in statistical methodologies. By benchmarking our system against these reference algorithms, we aim to provide a robust assessment of its efficacy in comparison to established techniques in the realm of topic detection.

The three selected strategy performance for textual topic detection, compared with SBERT, are presented in Figure 4 and summarized in Table 3. The SBERT model yielded the most favorable results regarding accuracy for textual topic detection, followed by LDA, while the LSA algorithm demonstrated comparatively lower performance. This discrepancy may be attributed to the outcomes being contingent on the feasibility of mapping DMOZ categories onto the concepts within the proposed ontology, as made in Table 2. Notably, in the case of LSA, the diminished accuracy appears linked to challenges associating specific topics generated by the model with their corresponding WordNet synsets. It can be postulated that SBERT exhibits superior generalization in concept recognition and is adept at mitigating noise inherent in specific datasets. Algorithm 1 shows the pseudo-code of the procedure adopted for detecting the textual topic from the full text of the web page using SBERT.

Table 3: Accuracy score detail for textual topic detection.

Algorithm	**Accuracy**	**Num. Correct**
LSA	0.1	117
LDA	0.34	407
SBERT	**0.53**	**620**

Algorithm 1: Detect Topics.

```
function PREPROCESS_TEXT(text)
    return Tokenize, lemmatize, and remove stop words from text
end function
function DETECT_TOPICS(text, num_clusters)
    processed_text ← PREPROCESS_TEXT(text)
    embeddings ← Generate SBERT embeddings for processed_text
    clusters ← Apply k-means clustering with num_clusters to embeddings
    for each cluster do
        synset_map ← Map words to WordNet synsets in processed_text
        top_synset ← Identify most common synset in synset_map
        Associate top_synset with the cluster as the representative topic
    end
    return (detected topics, associated synsets)
end function
```

4.3 Visual Topic Detection

The task of visual topic detection harnesses multimedia components, particularly images, to identify a document's primary topic, enhancing the overall performance of our framework. Our methodology takes a comprehensive approach to visual topic detection, employing three distinct configurations based on different feature extraction methods. This thorough exploration allows us to understand the strengths and limitations of each method.

PHOG (Bosch et al., 2007) is a global feature representation method that ensures comprehensive feature extraction and accuracy through dense grid computation and local contrast normalization with overlapping regions, making it suitable for precise visual analysis tasks.

SIFT (Lowe, 2004) is a robust local feature extraction technique that identifies key points of interest in gray-scale images, providing invariant descriptors for translation, rotation, and scaling, and is widely used in computer vision tasks.

VGG19 (Simonyan and Zisserman, 2014) is a deep convolutional neural network (CNN) designed for image classification tasks, with 19 layers, including convolutional and fully connected layers. It extracts visual content representation from intermediate layers, particularly the last max pooling layer, yielding deep features suitable for topic detection.

Evaluation of the three configurations based on utilized features (presented in Figure 4 and summarized in Table 4) reveals VGG19 features exhibit the highest accuracy, followed by PHOG and SIFT. These results align with expectations, as VGG19 and PHOG are promising candidates for precise feature matching, albeit with VGG19's computational time tradeoff due to its high dimensionality.

PHOG's superior accuracy over SIFT is attributed to its global nature. However, it can also be interpreted as a local feature due to its processing of images at various scales and resolutions. Selection of the optimal feature depends on a comprehensive analysis of the combined strategy and specific application requirements.

Table 4: Accuracy score detail for visual topic detection.

Algorithm	Accuracy	Num. Correct
SIFT	0.29	471
PHOG	0.39	348
VGG19	**0.82**	**1331**

4.4 Text-to-Image Generation

In our proposed strategy, an additional step involves generating multimedia data, prompting us to identify the most suitable model for this task. We evaluate three different models: StackGAN (Zhang et al., 2017), AttnGAN (Xu et al., 2018), and Stable Diffusion (Rombach et al., 2022). StackGAN employs a hierarchical Generative Adversarial Network for multi-stage image synthesis, progressively generating high-resolution images. AttnGAN incorporates attention mechanisms to enhance fine-grained image generation by focusing selectively on relevant regions. Stable Diffusion utilizes stable diffusion processes, controlling the gradual evolution of generated images to achieve high-quality synthesis. To assess fidelity in generated images from textual descriptions, we consider several metrics addressing different aspects of image fidelity. The Fréchet Inception Distance (FID) quantifies dissimilarity between the distributions of real and generated images, offering a comprehensive assessment. Other metrics, such as R-precision, Semantic Object Accuracy (SOA), and CLIP score, evaluate image-text matching. R-precision measures visual-semantic similarity by ranking retrieval results based on image and text features. SOA assesses object detection in images, providing class (SOA-C) and image (SOA-I) averages to gauge model alignment with textual descriptions. As highlighted in (Hinz et al., 2020), not all metrics are equally suited for evaluating generative models. Metrics like FID, SOA, and CLIP score better align with human visual assessment than IS and R-precision. Thus, we focus on the FID score, SOA metric, and CLIP score as crucial metrics for evaluating the generative model's performance. In Table 5, we have reported some metrics comparisons obtained during the evaluation process. The results show that Stable Diffusion is the model that best generates visually correct images that represent the textual prompt used for generation purposes.

4.5 Combined Topic Detection

The objective of the combined topic detection is to allocate a singular score based on weights assigned to individual textual and visual topic detection classifiers. The integration employs SUM and Ordered Weighted Averaging (OWA) operators. Various weight combinations have been systematically tested for each proposed scheme, excluding the OWA schema that employs OWA operators, providing a fuzzy logic approach. This comprehensive evaluation aims to identify optimal weight configurations contributing to an effective and nuanced integration of textual and visual topic detection outputs.

We decided to define four types of combination between the textual and visual strategy: the *A* combination pertains to the tests detailed earlier, wherein visual topic detection demonstrated superior performance. The *B* combination represents an average of computed scores, while the *C* combination assigns

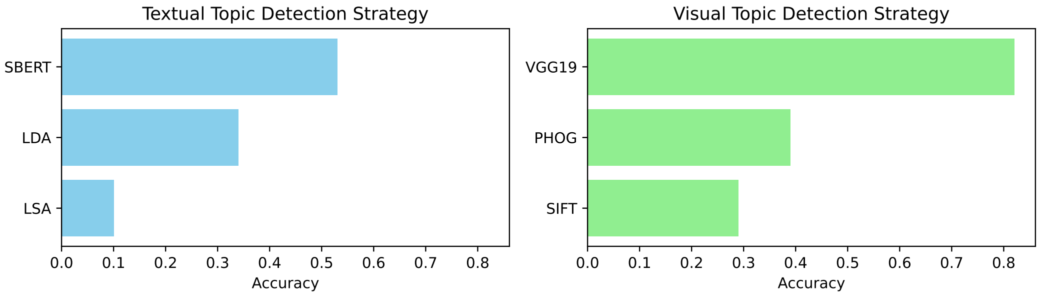


Figure 4: Accuracy comparison between the selected methods of textual topic detection (on the left) and visual topic detection (on the right).

Table 5: Metrics value comparison between GAN generative models and Stable Diffusion.

Model	FID	CLIP	SOA-C	SOA-I	R-precision
StackGAN (Zhang et al., 2017)	12.50	23.18	25.88	39.01	68.40
AttnGAN (Xu et al., 2018)	9.14	28.39	31.70	47.78	83.79
Stable Diffusion (Rombach et al., 2022)	**7.31**	**32.03**	**33.27**	**51.81**	**92.53**

greater importance to textual topic detection. Additionally, a combination was tested using OWA operators, denoted as the D scheme, with a weight vector $\vec{w} = (0.65, 0.35)$. This diverse set of combinations aims to thoroughly explore the interplay between textual and visual topic detection and identify optimal configurations for comprehensive evaluation. The testing procedure encompasses 36 combinations, employing schemes outlined in Table 6. In Figure 5, all results in classification accuracy among the described combinations are presented. Combinations incorporating visual topic detection and a fuzzy logic approach (combination D) are the most effective, particularly with deep feature-based schemas achieving high accuracy. However, textual topic detection schemes exhibit lower or similar accuracy. Combinations with equal weights for both classifiers can sometimes yield suboptimal classification accuracy.

Table 6: Weight configuration mapping for combined topic detection.

Combination	Text topic weight	Image topic weight
A	0.4	0.6
B	0.5	0.5
C	0.6	0.4
D	0.65	0.35

Visual classifiers with high accuracy significantly contribute to topic detection due to their enhanced discriminatory capabilities, particularly when images better represent specific concepts. However, terms with multiple meanings introduce uncertainty in the task. Despite their high computational demands, the SBERT model approach and VGG19 network-based visual topic detection yield the best results. The categorization process is an offline task prioritizing accuracy within the specific domain of interest.

Table 7 only summarizes the most significant experiments. The decision to choose the strategy for extracting topics from text and images and the method of combination used for classification was based on minimizing noise introduced by the image generation step, ensuring its inclusion as a variable in the evaluation step.

5 CONCLUSIONS AND FUTURE WORKS

In this work, we have proposed an innovative way to classify web documents using a generative approach and combined topic detection algorithm. We have unveiled promising directions for advancing the capabilities of an ontology-driven focused crawler. Exploring its extension to diverse conceptual domains, accompanied by a comparative assessment across various ontologies and knowledge bases, offers valuable insights into its adaptability and performance in capturing domain-specific information. Additionally, the investigation into alternative deep learning architectures, encompassing attention mechanisms, generative models, and self-supervised learning, holds the potential for augmenting the efficacy of image classification and retrieval tasks.

By evaluating different strategies and weighting schemes for merging textual and visual classifications, we have refined the model's performance. This

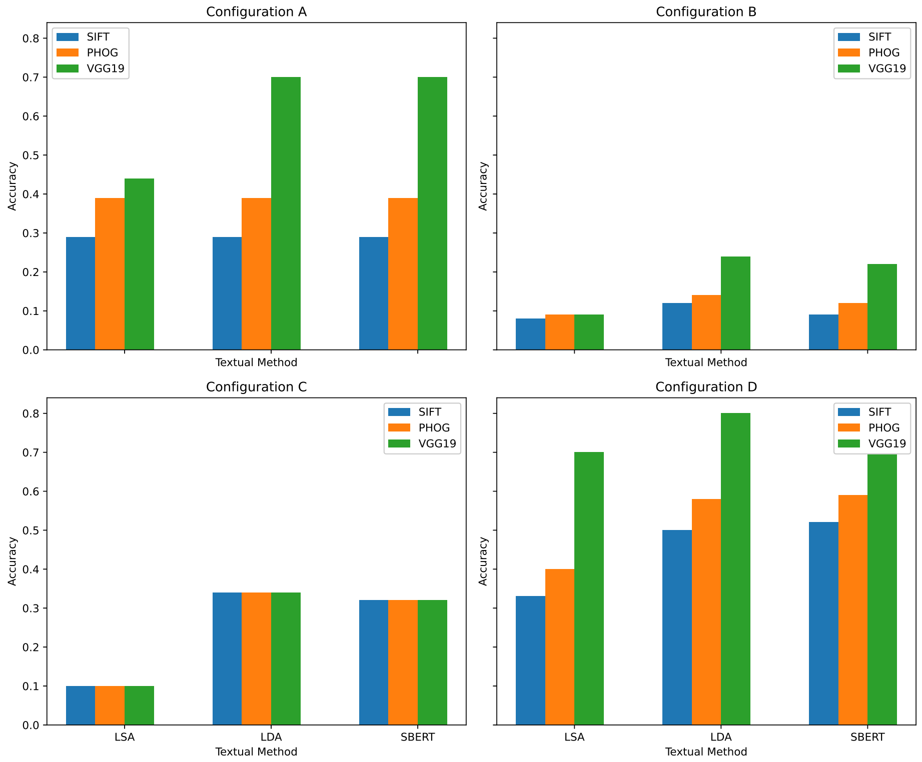


Figure 5: Accuracy performance among all the combinations selected, with all the visual and textual topic detection strategies descripted.

Table 7: Most relevant test details for combined topic detection.

Combination	Accuracy	Num. Correct	Combination	Accuracy	Num. Correct	Combination	Accuracy	Num. Correct
LSA-SIFT-A	0.29	348	LDA-SIFT-A	0.29	348	SBERT-SIFT-A	0.29	348
LSA-SIFT-B	0.08	100	LDA-SIFT-B	0.12	147	SBERT-SIFT-B	0.09	108
LSA-SIFT-C	0.10	117	LDA-SIFT-C	0.34	408	SBERT-SIFT-C	0.32	389
LSA-SIFT-D	0.33	402	LDA-SIFT-D	0.50	609	SBERT-SIFT-D	0.52	629
LSA-PHOG-A	0.39	471	LDA-PHOG-A	0.39	471	SBERT-PHOG-A	0.39	471
LSA-PHOG-B	0.09	108	LDA-PHOG-B	0.14	171	SBERT-PHOG-B	0.12	149
LSA-PHOG-C	0.10	117	LDA-PHOG-C	0.34	408	SBERT-PHOG-C	0.32	389
LSA-PHOG-D	0.40	480	LDA-PHOG-D	0.58	708	SBERT-PHOG-D	0.59	711
LSA-VGG19-A	0.44	539	LDA-VGG19-A	**0.70**	**846**	SBERT-VGG19-A	**0.70**	**846**
LSA-VGG19-B	0.09	111	LDA-VGG19-B	0.24	288	SBERT-VGG19-B	0.22	270
LSA-VGG19-C	0.10	117	LDA-VGG19-C	0.34	408	SBERT-VGG19-C	0.32	389
LSA-VGG19-D	**0.70**	**852**	LDA-VGG19-D	**0.80**	**966**	SBERT-VGG19-D	**0.80**	**965**

process has also enhanced our understanding of the robustness and sensitivity of these strategies under various conditions. Moreover, the expansion of the focused crawler framework to include multimedia content such as audio, video, and 3D models suggests a comprehensive approach to web page analysis. This expansion has the potential to significantly enhance our understanding of multimedia-rich online content, thereby improving the overall web document classification process.

ACKNOWLEDGEMENTS

This research has been partially supported by the Spoke 9 - Digital Society & Smart Cities - Centro Nazionale di Ricerca in High-Performance-Computing, Big Data and Quantum Computing, funded by European Union - NextGenerationEU - CUP: E63C22000980007. We also acknowledge financial support from the PNRR MUR project PE0000013-FAIR.

REFERENCES

Ahmed, Z. and Singh, H. (2019). Text extraction and clustering for multimedia: A review on techniques and challenges. In *2019 International Conference on Digitization (ICD)*, pages 38–43. IEEE.

Benfenati, D., Montanaro, M., Rinaldi, A. M., Russo, C., and Tommasino, C. (2023). Using focused crawlers with obfuscation techniques in the audio retrieval domain. In *International Conference on Management of Digital*, pages 3–17. Springer.

Bergman, M. K. (2001). White paper: the deep web: surfacing hidden value. *Journal of electronic publishing*, 7(1).

Bhatt, D., Vyas, D. A., and Pandya, S. (2015). Focused web crawler. *algorithms*, 5:18.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bosch, A., Zisserman, A., and Munoz, X. (2007). Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 401–408.

Chakrabarti, S., Van den Berg, M., and Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. *Computer networks*, 31(11-16):1623–1640.

Cheng, H., Liu, S., Sun, W., and Sun, Q. (2023). A neural topic modeling study integrating sbert and data augmentation. *Applied Sciences*, 13(7).

Clinchant, S., Ah-Pine, J., and Csurka, G. (2011). Semantic combination of textual and visual information in multimedia retrieval. In *Proceedings of the 1st ACM international conference on multimedia retrieval*, pages 1–8.

Danilak, M. (2017). Langdetect 1.0. 7. *Python Package Index*.

Farag, M. M., Lee, S., and Fox, E. A. (2018). Focused crawler for events. *International Journal on Digital Libraries*, 19:3–19.

Fatima, N., Faheem, M., and Dar, M. Z. N. (2023). Optimized focused crawling for web page classification. In *2023 International Conference on Energy, Power, Environment, Control, and Computing (ICEPECC)*, pages 1–6.

Fernàndez-Cañellas, D., Marco Rimmek, J., Espadaler, J., Garolera, B., Barja, A., Codina, M., Sastre, M., Giro-i Nieto, X., Riveiro, J. C., and Bou-Balust, E. (2020). Enhancing online knowledge graph population with semantic knowledge. In *The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part I 19*, pages 183–200. Springer.

Fu, T., Abbasi, A., and Chen, H. (2010). A focused crawler for dark web forums. *Journal of the American Society for Information Science and Technology*, 61(6):1213–1231.

Hajba, G. L. (2018). Website scraping with python. *Berkeley: Apress*.

Hassan, T., Cruz, C., and Bertaux, A. (2017). Ontology-based approach for unsupervised and adaptive focused crawling. In *Proceedings of The International Workshop on Semantic Big Data*, pages 1–6.

Hinz, T., Heinrich, S., and Wermter, S. (2020). Semantic object accuracy for generative text-to-image synthesis. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1552–1565.

K, N. T., S, C., G, B., Dharani, C., and Karishma, M. S. (2023). Comparative analysis of various web crawler algorithms.

Kittler, J., Hatef, M., Duin, R. P., and Matas, J. (1998). On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence*, 20(3):226–239.

Kumar, N. and Aggarwal, D. (2023). Learning-based focused web crawler. *IETE Journal of Research*, 69(4):2037–2045.

Kunder, M. d. (2018). The size of the world wide web (the internet). *Pobrano z: http://www. world widewebsize. com/(19.01. 2017)*.

Landauer, T. K., Foltz, P. W., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284.

Liu, J., Li, X., Zhang, Q., and Zhong, G. (2022). A novel focused crawler combining web space evolution and domain ontology. *Knowledge-based systems*, 243:108495.

Lowe, G. (2004). Sift-the scale invariant feature transform. *Int. J*, 2(91-110):2.

Mary, J. D. P. N. R., Balasubramanian, S., and Raj, R. S. P. (2022). An enhanced focused web crawler for biomedical topics using attention enhanced siamese long short term memory networks. *Brazilian Archives of Biology and Technology*, 64:e21210163.

Mohandes, M., Deriche, M., and Aliyu, S. O. (2018). Classifiers combination techniques: A comprehensive review. *IEEE Access*, 6:19626–19639.

Pant, G. and Srinivasan, P. (2005). Learning to crawl: Comparing classification schemes. *ACM Transactions on Information Systems (TOIS)*, 23(4):430–462.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Ridnik, T., Ben-Baruch, E., Noy, A., and Zelnik-Manor, L. (2021). Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*.

Rinaldi, A. M. (2014). Using multimedia ontologies for automatic image annotation and classification. In *2014 IEEE International Congress on Big Data*, pages 242–249. IEEE.

Rinaldi, A. M. and Russo, C. (2021). Using a multimedia semantic graph for web document visualization and summarization. *Multimedia Tools and Applications*, 80(3):3885–3925.

Rinaldi, A. M., Russo, C., and Tommasino, C. (2021a). A semantic approach for document classification using deep neural networks and multimedia knowledge graph. *Expert Systems with Applications*, 169:114320.

Rinaldi, A. M., Russo, C., and Tommasino, C. (2021b). Visual query posing in multimedia web document retrieval. In *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*, pages 415–420. IEEE.

Rinaldi, A. M., Russo, C., and Tommasino, C. (2021c). Web document categorization using knowledge graph and semantic textual topic detection. In *Computational Science and Its Applications–ICCSA 2021: 21st International Conference, Cagliari, Italy, September 13–16, 2021, Proceedings, Part III 21*, pages 40–51. Springer.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695.

Russo, C., Madani, K., and Rinaldi, A. M. (2020). An unsupervised approach for knowledge construction applied to personal robots. *IEEE Transactions on Cognitive and Developmental Systems*, 13(1):6–15.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. (2022). Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494.

Shrivastava, G. K., Pateriya, R. K., and Kaushik, P. (2023). An efficient focused crawler using lstm-cnn based deep learning. *International Journal of System Assurance Engineering and Management*, 14(1):391–407.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sood, G. (2016). Parsed DMOZ data.

Tchakounte, F., Ngnintedem, J. C. T., Damakoa, I., Ahmadou, F., and Fotso, F. A. K. (2022). Crawlshing: A focused crawler for fetching phishing contents based on graph isomorphism. *Journal of King Saud University-Computer and Information Sciences*, 34(10):8888–8898.

Wu, H. and Hou, D. (2023). A focused event crawler with temporal intent. *Applied Sciences*, 13(7).

Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., and He, X. (2018). Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324.

Yager, R. R. and Kacprzyk, J. (2012). *The ordered weighted averaging operators: theory and applications*. Springer Science & Business Media.

Yan, W. and Pan, L. (2018). Designing focused crawler based on improved genetic algorithm. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pages 319–323. IEEE.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. (2017). Stackgan: Text to photorealistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915.

Antibiotic Resistance Gene Identification from Metagenomic Data Using Ensemble of Finetuned Large Language Models

Syama K[a] and J. Angel Arul Jothi[b]

Department of Computer Science, Birla Institute of Technology and Science Pilani Dubai Campus, Dubai, U.A.E.
p20190011@dubai.bits-pilani.ac.in, angeljothi@dubai.bits-pilani.ac.in

Keywords: Antibiotic Resistance Gene, Ensemble Learning, Transformer, LLM.

Abstract: Antibiotic resistance is a potential challenge to global health. It limits the effect of antibiotics on humans. Antibiotic resistant genes (ARG) are primarily associated with acquired resistance, where bacteria gain resistance through horizontal gene transfer or mutation. Hence, the identification of ARGs is essential for the treatment of infections and understanding the resistance mechanism. Though there are several methods for ARG identification, the majority of them are based on sequence alignment and hence fail to provide accurate results when the ARGs diverge from those in the reference ARG databases. Additionally, a significant fraction of proteins still need to be accounted for in public repositories. This work introduces a multi-task ensemble model called ARG-LLM of multiple large language models (LLMs) for ARG identification and antibiotic category prediction. We finetuned three pre-trained protein language LLMs, ProtBert, ProtAlbert, and Evolutionary Scale Modelling (ESM), with the ARG prediction data. The predictions of the finetuned models are combined using a majority vote ensembling approach to identify the ARG sequences. Then, another ProtBert model is fine-tuned for the antibiotic category prediction task. Experiments are conducted to establish the superiority of the proposed ARG-LLM using the PLM-ARGDB dataset. Results demonstrate that ARG-LLM outperforms other state-of-the-art methods with the best Recall of 96.2%, F1-score of 94.4%, and MCC of 90%.

1 INTRODUCTION

Antibiotics are one of the significant discoveries of the 20th century, saving millions of lives from infectious diseases. However, their widespread use and misuse make pathogens increasingly resistant to antibiotics. The World Health Organization (WHO) has listed antibiotic resistance among the top 10 threats to global health. Furthermore, according to WHO, antibiotic resistance directly caused 1.27 million deaths worldwide in 2019, and if no action is taken, this number is predicted to increase to 10 million by 2050 (Murray et al., 2022; Lázár and Kishony, 2019). Additionally, antibiotic resistance is spread between pathogens by transferring antibiotic resistant genes (ARG) through food, water, animals, and humans. Therefore, identifying ARG in pathogens is significant in stopping their spread, understanding the resistance mechanism, and developing the targeted treatment or control measures. Global efforts such as the Global Antimicrobial Resistance Surveillance System and the Global Antibiotic Research and Development Partnership have been initiated for this. The primary focus of these consortium efforts is to develop an efficient tool for identifying antibiotic resistance (Mendelson M, 2015). Culture-based Antibiotic Susceptibility Tests (AST) are the standard practice in clinical microbiology that determine the effectiveness of antibiotics against specific bacteria. However, it takes weeks to get the results and does not apply to the unculturable bacteria (Pham and Kim, 2012).

The emergence of high-throughput DNA sequencing techniques in metagenomics helped the development of various tools to profile the DNA of pathogens and increased the amount of DNA and protein sequences in public databases. For example, UniProt (Consortium, 2015) is the largest collection of protein sequences available after merging it with proteins translated from multiple metagenomic sequencing projects. This, in turn, encouraged researchers to enhance the understanding of the functional diversity of microbial communities significantly. This knowledge helped identify ARGs in different pathogens present in livestock manure, compost, wastewater treatment plants, soil, water, and the human microbiome (Mao et al., 2015; Pehrsson et al., 2016). How-

[a] https://orcid.org/0009-0000-6297-4407
[b] https://orcid.org/0000-0002-1773-8779

K., S. and Jothi, J.
Antibiotic Resistance Gene Identification from Metagenomic Data Using Ensemble of Finetuned Large Language Models.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 102-112
ISBN: 978-989-758-716-0; ISSN: 2184-3228

ever, the main challenge faced by researchers is a notable portion of proteins remains unannotated.

ARG identification methods are categorized into sequence-based alignment or assembly and machine learning (ML)-based. For alignment-based methods (McArthur et al., 2013), the query ARG sequence is compared against the existing ARG sequences in the database using alignment tools such as BLAST (Altschul et al., 1990), DIAMOND (Buchfink et al., 2015), and BWA (Li and Durbin, 2009). Although these methods are widely used in ARG identification, they also have disadvantages. For example, the sequence-based methods may miss novel genes that are not present in the reference genome database (Chowdhury et al., 2019), and the accurate results are highly dependent on the value of the critical hyperparameter, such as the similarity threshold (Li et al., 2021). Alternatively, multiple ML methods have been developed for ARG identification tasks (Gibson et al., 2015; Arango-Argoty et al., 2018a). ML-based methods depend on the features representing the characteristics of ARGs (Ruppé et al., 2019) and learn the statistical patterns of ARGs. So, ML methods are able to identify novel genes (Li et al., 2018). However, the ML methods are trained using the genetic features extracted from the ARG sequences of the particular organism of interest. This limits their capacity to a more generalized applicability. Deep learning (DL) methods are especially powerful due to their inherent capability to learn features, avoiding separate feature extraction. In both ML and DL methods, researchers always try to improve and optimize classification models to achieve better accuracy. Ensemble learning is a widely used technique to enhance classification accuracy (Miah et al., 2024). It aggregates two or more base classifiers to improve the predictive performance of the combined classifier, and it overcomes the weakness of a single weak base classifier.

Presently, to uncover the properties of the novel ARGs, the ideas embedded in natural language processing (NLP) are adopted into protein sequence processing. Protein sequences are considered as sentences in protein language, and then NLP techniques are used to extract the features in the protein sequences. In particular, transformer-based large language models (LLM) (Devlin et al., 2018) have achieved state-of-the-art (SOTA) performance for several NLP and protein language tasks (Bepler and Berger, 2021). Few LLM-based ARG identification models have been developed (Wu et al., 2023) for ARG identification. These models have been widely used as feature extractors, demonstrating significant improvements in various tasks. However, finetuning the pre-trained model further improves the model's predictive power. Finetuning involves training a pre-trained model further on a specific task or dataset to enhance its performance for that task. Since the model is already pre-trained on a large dataset, finetuning requires significantly less time and computational resources. Hence, an ARG prediction tool that harnesses the power of LLM-based models is in high demand.

In this work, a multi-task ensemble model, ARG-LLM, is used to leverage the prediction of ARG and then further identify what antibiotic family it is resistant to. It harnesses the capabilities of three publicly available pre-trained transformer-based LLMs such as ProtBert (Elnaggar et al., 2021), ProtAlbert (Elnaggar et al., 2021), and Evolutionary Scale Modelling (ESM) (Rao et al., 2021). In the first task, the three LLMs are finetuned with the ARG prediction dataset. The prediction output of each of the language models is passed through a majority-voting ensemble method. In the second task, the ProtBert model is finetuned with the Antibiotic category prediction dataset, and those sequences predicted as ARGs in the first task are further passed through the fine-tuned model for the prediction of antibiotic categories.

This paper is organized as follows. Section 2 reviews previous works done in ARG prediction and Antibiotic category prediction tasks. Details of the dataset used in this work are explained in Section 3. Section 4 presents the methodology. The experiments and the evaluation metrics are provided in Section 5. The results and discussion are presented in Section 6. Section 7 provides the conclusion and the future work.

2 RELATED WORKS

Antibiotic resistance is a serious global threat to human health that urgently requires practical action. Identifying antibiotic resistant genes is a crucial step in understanding the mechanism of antibiotic resistance. This section covers an overview of the related works introduced in the ARG identification field, emphasizing the works done using ML and DL methods.

The traditional computational methods developed for ARG identification are all sequence-based. Hence, they are designed to identify specific pathogens' ARGs. For instance, ResFinder (Kleinheinz et al., 2014) predicts specifically plasmid-borne ARGs and the tool developed in (Bradley et al., 2015) is dedicated to 12 types of antimicrobials. Similarly, another study (Davis et al., 2016) is limited to identifying ARGs encoding resistance to carbapenem, methicillin, and beta-lactam antibiotics. Most of these tools identify the query sequence's similarity with the sequences in the existing microbial resistance databases, using a ”best hit” approach to predict whether a sequence is an ARG. These methods require a cutoff threshold to identify the similarity between the sequences. This restricts those models from identifying novel ARGs (McArthur and Tsang, 2017). To overcome the disadvantages of the previous methods, many ML and DL-based methods have been introduced.

The work by Arango et al. (Arango-Argoty et al., 2018b) introduced DeepARG, a novel DL approach for predicting ARGs from metagenomic data. It contained two components: DeepARG-SS for classifying short reads and DeepARG-LS for annotating novel ARG genes. It used a Deep Neural Network (DNN) architecture for predicting ARGs from metagenomic data, and a bitscore-based dissimilarity index was used as the feature for the DL model. The DeepARG-SS model, trained on short sequence reads, achieved an overall precision of 0.97 and recall of 0.91 for the 30 antibiotic categories tested.

The HMD-ARG model in (Li et al., 2021) consisted of hierarchically connected three DL models that predict ARG properties by focusing on antibiotic resistance type, mechanism, and gene mobility. Convolutional Neural Network (CNN) models were used at each level. At the first level, the sequences were classified into ARG or not. The ARG sequences were classified in the second level based on the resistant antibiotic family, resistant mechanism, and gene mobility information. In the final level, if the predicted antibiotic family was beta-lactamase, the framework further predicted the subclass of beta-lactamase. The framework could identify ARGs without querying existing databases. The HMD-ARG model achieved an Accuracy of 0.948, Precision of 0.939, Recall of 0.951, and F1 of 0.938.

Another work named ARG-SHINE by (Wang et al., 2021) introduced a novel ARG prediction framework by integrating sequence homology and functional information with DL techniques. It used CNN for the classification. This framework proposed the method to combine sequence homology, functional information, and DL, and the integration improved antibiotic resistance prediction accuracy. The ARG-SHINE model achieved an Accuracy of 0.8557 and an F1 of 0.8595.

A recent work named PLM-ARG proposed by (Wu et al., 2023) introduced a novel method for ARG identification using a pre-trained protein language model, ESM-1b. It harnessed the power of ESM-1b to generate embedding for protein sequences and utilized the Extreme Gradient Boosting (XGBoost) ML model to classify the antibiotic category. The study provided insights into applying Artificial Intelligence (AI)-powered language models for ARG identification. The PLM-ARG model achieved an Accuracy of 0.912, Precision of 1, Recall of 0.825, F1 of 0.904, and Mathews Correlation Coefficient (MCC) of 0.838.

The literature review shows that the efficacy of transformer-based NLP models is less utilized in the ARG identification task. Researchers have identified that finetuning the transformer-based models gives an exceptional performance in NLP tasks (Devlin et al., 2018). However, finetuning the transformer-based models for ARG prediction with the ARG dataset has yet to be explored. Hence, in this work, we finetune the protein language models and use the finetuned model for classification. Additionally, we utilized the capacity of ensembling the prediction of the finetuned models to identify ARG sequences.

3 DATASET

We collected antibiotic resistance gene sequences from the published ARG database PLM-ARGDB (Wu et al., 2023). It contains 57158 gene sequences, 28579 of which are labeled as ARG and 28579 of which are labeled as non-ARGs. The sequences which are labeled as ARG are further labeled with their antibiotic category. The 26391 ARGs in the 28579 ARG sequences are labeled with 22 explicit resistance categories, and 2188 ARGs are tagged with a general category “multi-drug” or “antibiotic without defined classification.” PLM-ARGDB is constructed by extracting ARG sequences from six publicly available ARG databases, as 4790 from CARD (Jia et al., 2016), 859 from ResFinder (Zankari et al., 2012), 2044 from MEGARes (Lakin et al., 2017), 444 from AMRFinderPlus (Feldgarden et al., 2019), 9863 from ARGMiner, and 10579 from HMD-ARG-DB (Li et al., 2021). The non-ARG sequences are taken from the UniProt database.

4 PROPOSED METHODOLOGY

In this work, we introduce a novel multi-task ensemble framework, ARG-LLM, which automatically identifies the ARGs and the categories of antibiotics to which the pathogen is resistant. Figure 1 presents the overall methodology of this work. ARG-LLM performs two tasks: one is the ARG prediction task, and the other is the Antibiotic category prediction task. ARG-LLM starts with preprocessing the dataset and preparing the data for subsequent finetuning and prediction. The ARG prediction task

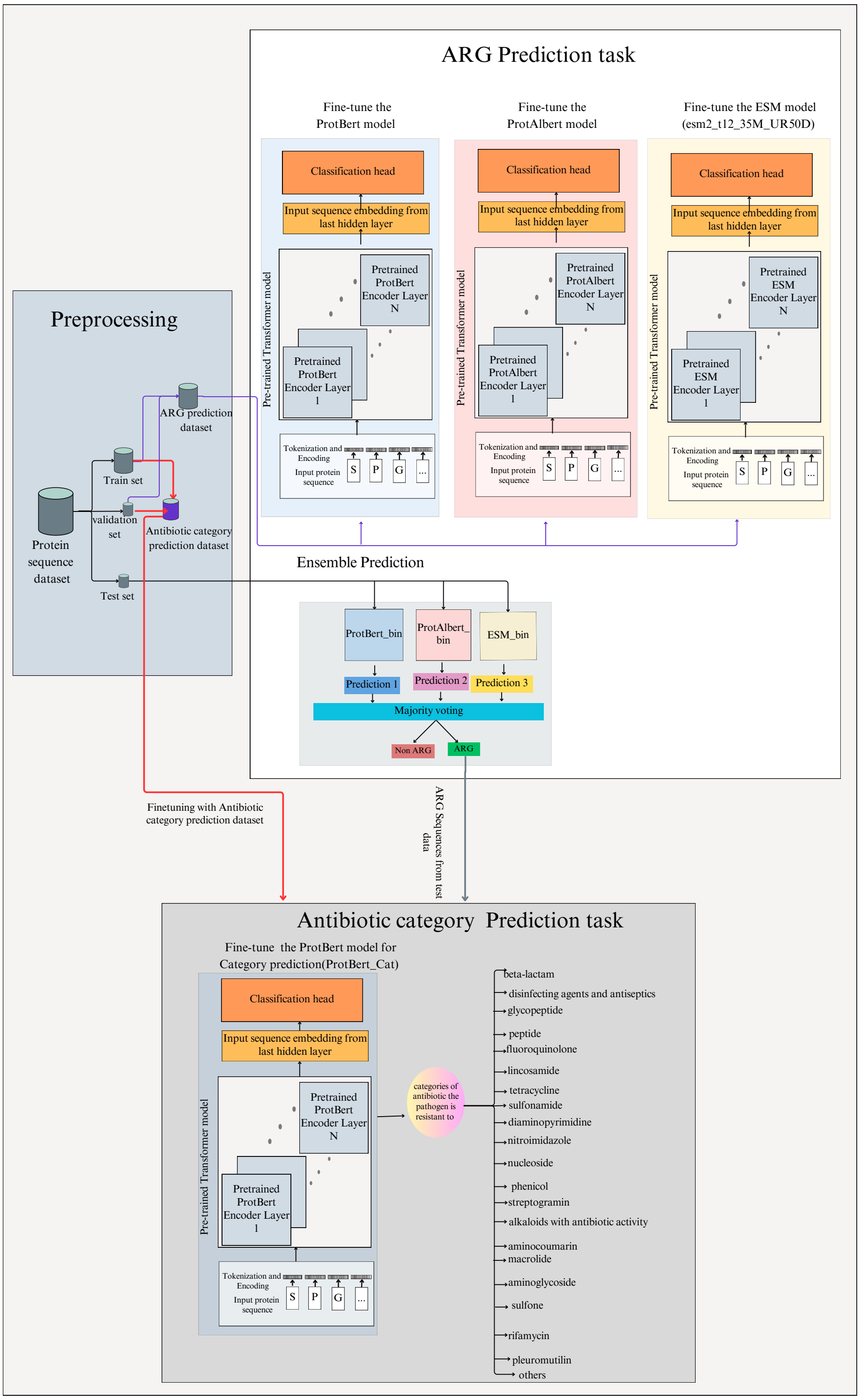

ARG Prediction task
Fine-tune the ProtBert model
Fine-tune the ProtAlbert model
Fine-tune the ESM model (esm2_t12_35M_UR50D)
Classification head
Input sequence embedding from last hidden layer
Pretrained ProtBert Encoder Layer N
Pretrained ProtBert Encoder Layer 1
Pretrained ProtAlbert Encoder Layer N
Pretrained ProtAlbert Encoder Layer 1
Pretrained ESM Encoder Layer N
Pretrained ESM Encoder Layer 1
Pre-trained Transformer model
Tokenization and Encoding
Input protein sequence
S
P
G
...
Preprocessing
Protein sequence dataset
Train set
validation set
Test set
ARG prediction dataset
Antibiotic category prediction dataset
Ensemble Prediction
ProtBert_bin
ProtAlbert_ bin
ESM_bin
Prediction 1
Prediction 2
Prediction 3
Majority voting
Non ARG
ARG
Finetuning with Antibiotic category prediction dataset
ARG Sequences from test data
Antibiotic category Prediction task
Fine-tune the ProtBert model for Category prediction(ProtBert_Cat)
categories of antibiotic the pathogen is resistant to
beta-lactam
disinfecting agents and antiseptics
glycopeptide
peptide
fluoroquinolone
lincosamide
tetracycline
sulfonamide
diaminopyrimidine
nitroimidazole
nucleoside
phenicol
streptogramin
alkaloids with antibiotic activity
aminocoumarin
macrolide
aminoglycoside
sulfone
rifamycin
pleuromutilin
others

Figure 1: Overview of the proposed methodology.

finetunes the pre-trained base LLM models, such as ProtBert, ProtAlbert, and ESM. Then, these finetuned models (ProtBert_bin, ProtAlbert_bin, ESM_bin) are used as base classifiers for the majority voting classifier to predict whether the given sequence is ARG or not. The Antibiotic category prediction task finetunes the ProtBert model and then predicts the categories of antibiotics for those sequences that are predicted as

ARG during the ARG prediction task. The following subsections explain each step in detail.

4.1 Preprocessing

In this step, the sequences are read from the database which is in the format of a FASTA file. The ARG sequences labeled "multi-drug" or "antibiotic without defined classification" are changed to the label "others". Thus, the sequences have two labels, where one is the ARG label and the other 21 are the antibiotic categories label. The ARG label is given as 0 or 1, where 0 represents non-ARG and 1 represents ARG. The antibiotic category labels are present for only those sequences with ARG equal to 1. The antibiotic category labels are transformed into a binary matrix format using sklearn MultiLabelBinarizer(). Then, separate train and validation sets are formed, one for the binary (ARG) prediction and the other for the multilabel (Antibiotic category) prediction. Hence, in this work, we refer to the ARG prediction dataset as the training and validation datasets used for ARG prediction. These datasets contain only the protein sequences and their ARG labels. Furthermore, these datasets are used to finetune the three base LLMs. Similarly, we refer to the Antibiotic category prediction dataset as the train and the validation datasets used for Antibiotic category prediction. This dataset contains the protein sequences and their Antibiotic category labels, which are used to finetune the ProtBert model for Antibiotic category prediction.

4.2 Architecture of ARG-LLM

The two tasks of ARG-LLM are explained in the following subsections.

4.2.1 ARG Prediction

ARG prediction task includes finetuning the base LLM models with ARG prediction dataset, and combine the predictions done by the finetuned model using ensemble prediction.

a) Finetuning the LLMs:
This task utilized three transformer-based LLMs. The transformer model was introduced in 2017 by Vaswani et al. (Vaswani et al., 2017). It is a neural network model that understands the context of the input sequence. Usually, the transformer has an encoder-decoder architecture. However, the pre-trained models used in this study are based on Encoder-only Transformer (EOT) architecture because they focus on generating embedding for the protein sequences. EOT understands the features and patterns in the input sequence and generates a representation for the input. The encoder is a stack of multiple layers. The encoder takes the input protein sequences composed of amino acids, passes them through a series of operations, and generates the abstract representation that encapsulates the learned information from the entire sequence.

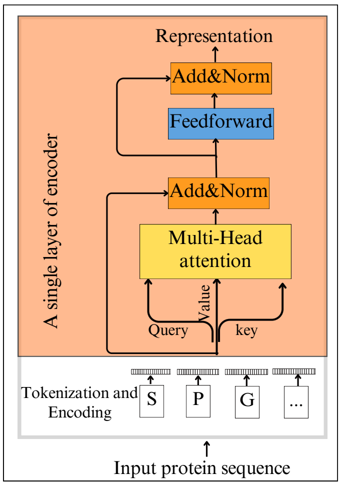


Figure 2: Architecture of a single layer of transformer encoder.

Figure 2 shows a single encoder layer in the transformer. It comprises of three modules: tokenization and encoding module, self-attention module, and feed-forward module. Tokenization aims to tokenize each amino acid (word) in the protein sequence (sentence). Then, the encoding step converts each token to a vector. In order to provide information about the position of a token in the sequence, positional encoding is then added to the vector of each token. Since transformers lack an inherent sense of sequence order, positional encoding is necessary to add information about the order of tokens in each sequence. All the pre-trained models used in this study use absolute positional encoding (Vaswani et al., 2017). Absolute positional encoding uses sine and cosine functions to generate a unique vector for each token's fixed position in the sequence. These vectors are added to the input representations of amino acids before being fed into the transformer layers. The positional encoding for each position *pos* pos is calculated as follows.

$$\begin{aligned} PE(pos,2i) &= sin(\frac{pos}{10000^{2i/d_{model}}}) \\ PE(pos,2i+1) &= cos(\frac{pos}{10000^{2i/d_{model}}}) \end{aligned} \tag{1}$$

where i is the dimension of the positional encoding, d_model is the dimensionality of the encoded input.

The self-attention module consists of self-attention and layer normalization. Self-attention uses a multi-head attention mechanism to relate each amino acid in the input protein sequence with other amino acids. The encoded vector of each amino acid (token) is then fed to three parameters: Query (Q), Key (K), and Value (V). Q is a vector representing the token for which the attention scores are calculated. K are vectors associated with each token in the sequence and are used to compare against the Q vector to compute a score. V are vectors the same as K but are used to calculate the final representation of the word after the attention mechanism is applied. In a multi-head attention mechanism with h heads, the Q, K, and V are linearly projected, and h versions of Q, K, and V are obtained as follows.

$$Q_i = XW_i^Q; \quad K_i = XW_i^K; \quad V_i = XW_i^V \qquad (2)$$

where $i = 1, 2, \cdots, h$,
W_i^Q,W_i^K, and W_i^V are the learned projection matrices for head i, and X is the input tokens matrix.

Each attention head i performs a scaled dot product attention as follows.

$$Attention(Q_i, K_i, V_i) = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})V_i \qquad (3)$$

where d_k is the dimension of the Key vectors.

After computing the attention from all the heads, the attention vectors are concatenated and transformed using a linear transformation as given below.

$$MultiHead(Q, K, V) = concat(head_1, head_2, \cdots, head_h)W^O \qquad (4)$$

where $head_i = Attention(Q_i, K_i, V_i)$, and W^O is the learned weight matrix for linear transformation.

By computing attention scores across multiple heads and combining the results, the transformer model can better understand the context and dependencies within the data. The output of the multi-head attention is added to the input using the residual connection, and the sum is passed to the layer normalization operation.

The output of the self-attention module is passed to the feed-forward module. The feed-forward module consists of a fully connected feed-forward network containing two linear transformations with a ReLU activation in between. Equation 5 shows the feed-forward network operation performed on input x.

$$FFN(x) = ReLU(W_1 x + b_1)W_2 + b_2 \qquad (5)$$

where W_1, W_2 are the learned weight matrices, and b_1, b_2 are biases.

The output of the feed-forward module is added to its input using a residual connection, followed by layer normalization. These operations are performed in each of the layers of the encoder. The transformer encoder can have N such layers. The output of the final encoder layer is the abstract representation of the input sequence with a rich contextual understanding.

After the success of transformers in many NLP tasks, Devlin et al. introduced a bidirectional Encoder Only transformer called Bidirectional Encoder Representations from Transformers (BERT) for text processing in 2018 (Devlin et al., 2018). BERT differs from traditional transformer models by using a bidirectional approach, meaning it considers the context from both the left and right sides of a sequence. BERT is pre-trained on a large corpus of text using two unsupervised tasks: Masked Language Modeling (MLM) (Taylor, 1953) and Next Sentence Prediction (NSP). BERT can be adapted to various NLP tasks by adding a simple output layer. The models used in this work, such as ProtBert, ProtAlbert, and ESM, are based on the BERT architecture.

ProtBert: It is a protein-specific variant of BERT[1] developed by training the pre-trained BERT model using 393 billion amino acid sequences from UniRef (Suzek et al., 2015) and BFD(Steinegger and Söding, 2018) databases. It is trained using MLM objective in a self-supervised manner. The number of layers of ProtBert was increased to 30 compared to BERT, which had 24 layers.

ProtAlbert: It is a protein-specific variant of A Lite BERT(ALBERT [2]) model developed by pretraining the Albert model using UniRef100 (Suzek et al., 2015) dataset. Albert models use parameter sharing across layers, which reduces the total number of parameters while maintaining a similar model depth, making it a Lite version of BERT.

ESM: It is a transformer-based model designed explicitly for protein sequence analysis and was developed by Meta AI (formerly Facebook AI Research). ESM is trained with UniRef50 (Suzek et al., 2015), a massive dataset of 250 million protein sequences encompassing 86 billion amino acids. The model utilizes unsupervised learning to learn representations that capture biological properties and evolutionary diversity from sequence data. It comes in different variants based on the number of parameters and layers. In this work, we used "esm2_t12_35M_UR50D", which refers to a specific variant or configuration of the ESM model.

To finetune the pre-trained LLMs for the ARG prediction task, the model is modified by adding a

[1] https://github.com/google-research/bert

[2] https://github.com/google-research/albert

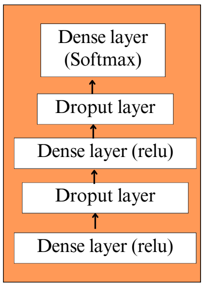


Figure 3: Architecture of the classification head.

classification head on top of the model architecture. Figure 3 presents the architecture of classification head. It includes two fully connected layers with a ReLU activation function. Each fully connected layer is followed by a dropout layer. Finally, a softmax activation function is used for the classification.

Then we train the entire model with the ARG prediction dataset. During training, only the weights of the last k layers of the pre-trained model and the newly added classification head are updated based on the loss calculated from the classification task. The loss function used here is the Binary Cross Entropy (BCE) loss. After finetuning, the finetuned models are called ProtBert_bin, ProtAlbert_bin, and ESM_bin respectively.

b) ARG Prediction using Ensemble of Finetuned LLMs:
The finetuned models are used for prediction with the test data. The predictions furnished by the models mentioned above are combined through a process known as majority voting (Dietterich, 2000). This entails tallying the occurrences of ARG and non-ARG labels. The final prediction is obtained depending on the votes achieved by each label. For a given protein sequence x, base classifier h_i, each h_i produces a predicted class label $h_i(x)$, then the majority voting method can be performed as follows.

$$\hat{y} = argmax_{c\in C}\sum_{i=1}^{N}\mathbb{1}(h_i(x) = c) \tag{6}$$

where C = {ARG, non-ARG}; set of possible class labels, N=3 is the number of base classifiers, $\mathbb{1}$ is the indicator function, which return 1 if the argument is true, *argmax* selects the class with the maximum vote and $\hat{y}$ is the final predicted class label.

4.2.2 Antibiotic Category Prediction

In this task, a ProtBert model with a classification head for predicting the antibiotic categories of the ARG sequences is finetuned with the Antibiotic category prediction dataset. The finetuned model is called ProtBert_cat. Then, ProtBert_cat is used to predict the antibiotic categories of those sequences which are predicted as ARG by the ensemble model.

5 EXPERIMENTAL SETUP AND EVALUATION METRICS

5.1 Experimental Setup

The proposed framework is written in Python 3, and the libraries used are Sklearn version 1.0.2 and Pytorch version 1.13. All the experiments are executed on an ml.g5.xlarge instance type in Amazon SageMaker, equipped with an NVIDIA A10G Tensor Core GPU and 24 GB dedicated memory. Table 1 presents the parameters used by each LLM.

5.2 Evaluation Metrics

The performance of the proposed model is evaluated using metrics like: F1-score (F1), accuracy, precision, recall and Matthews Correlation Coefficient (MCC). Let TP, TN, FP, and FN be the number of true positives, true negatives, false positives, and false negatives, respectively, then each of the metrics is calculated as follows. For the multilabel classification of category prediction the model performance was calculated based on micro-averages for each performance metric. Each of the metric is calculated as shown in equation 7.

$$\begin{aligned} Accuracy &= \frac{TP+TN}{TP+FP+TN+FN} \\ Recall &= \frac{TP}{TP+FN} \\ Precision &= \frac{TP}{TP+FP} \\ F1-score &= \frac{2\times Precision\times Recall}{Precision+Recall} \\ MCC &= \frac{TP\times TN - FP\times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \end{aligned} \tag{7}$$

6 RESULTS AND DISCUSSIONS

This section presents and discusses the results of the proposed ensemble framework obtained on the test dataset.

Table 1: The parameters and configurations used by each model.

Parameters	ProtBert	ProtAlbert	ESM
Number of Layers	30	12	12
Embedding Size	1024	4096	4096
Number of Parameters	420 M	224 M	35 M
Learning rate	0.0005	0.0005	0.0005
Optimizer	Adam	Adam	Adam
Batch size	1	1	1
1st dense layer size in the classification head	512	512	512
2nd dense layer size in the classification head	128	128	128
number of unfrozen layers	8	5	8
Loss function	BCE	BCE	BCE

Table 2: Comparison results of individual finetuned LLMs and ARG-LLM on ARG and Antibiotic category prediction (Best results are highlighted in bold).

	ARG Prediction					Category Prediction				
	Accuracy	Precision	Recall	F1	MCC	Accuracy	Precision	Recall	F1	MCC
ProtBert	0.9827	0.9689	0.9753	0.9721	0.9712	0.9168	0.9324	0.9289	0.9306	0.8754
ProtAlbert	0.9752	0.9638	0.9747	0.9692	0.9624	0.9175	**0.9461**	0.9293	0.9376	0.8854
ESM	0.9832	0.9723	0.9859	0.9791	0.9763	**0.9281**	0.9085	0.9612	0.9343	0.8967
ARG-LLM	**0.9931**	**1**	**0.9859**	**0.9929**	**0.9862**	0.9232	0.9261	**0.9616**	**0.9435**	**0.9001**

Table 3: Comparison with Pre-trained LLM models as embedding generator and XGBoost as classifier for ARG and Antibiotic category prediction (Best results are highlighted in bold).

	ARG Prediction					Category Prediction				
	Accuracy	Precision	Recall	F1	MCC	Accuracy	Precision	Recall	F1	MCC
ProtBert	0.9562	0.9678	0.9587	0.9632	0.9215	0.9108	0.9075	0.9151	0.9113	0.8941
ProtAlbert	0.9487	0.9758	0.9475	0.9614	0.9245	0.9012	0.9161	0.9327	0.9243	0.8995
ESM	0.9923	0.9954	0.9852	0.9902	0.9758	0.9174	0.9167	0.9296	0.9231	0.789
ARG-LLM	**0.9931**	**1**	**0.9859**	**0.9929**	**0.9862**	**0.9232**	**0.9261**	**0.9616**	**0.9435**	**0.9001**

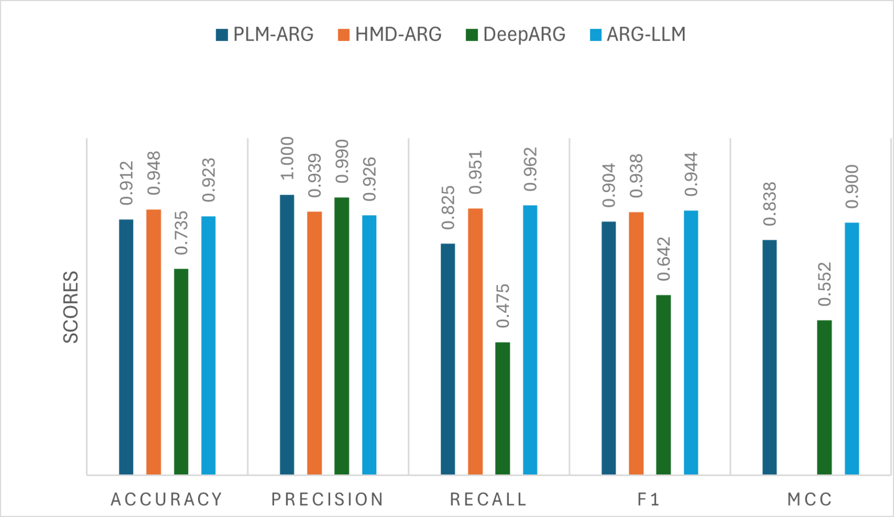


Figure 4: Comparison of the performance of ARG-LLM with state-of-the-art approaches on Antibiotic category prediction.

6.1 Comparison with Individual Finetuned LLMs

Experiments are conducted using the individual finetuned models ProtBert_bin, ProtAlbert_bin, and ESM_bin separately for ARG prediction and the ProtBert_cat model for category prediction. The results achieved by each individual finetuned model are compared with those of ARG-LLM. Table 2 presents the comparison results. The results show that the ARG-LLM invariably delivered competitive results across multiple metrics, highlighting its ability to predict ARG and its categories. The proposed model achieved the best accuracy of 0.9931, Precision of 1, Recall of 0.9859, F1 of 0.9929, and MCC of 0.9862 for the ARG prediction task. For the Antibiotic category prediction task, the ESM model achieves the best accuracy with a value of 0.9281, and the ProtAlbert model achieves the best Precision of 0.9461. ARG-LLM achieves the best Recall, F1, and MCC with values 0.9616, 0.9435, and 0.9001, respectively. Additionally, the ESM model's performance is significant out of the three transformer models, as it consistently shows strong prediction capability.

6.2 Comparison with Pre-Trained LLMs as Embedding Generators

In this experiment, the pre-trained ProtBert, ProtAlbert, and ESM models are used to generate embeddings for the sequences in the dataset. Then, the embeddings are provided as input for a subsequently trained XGBoost model for ARG prediction. A trained multilabel XGBoost classifier is used to predict the antibiotic categories. The comparison results are presented in Table 3. From the table 3, it is evident that the ARG-LLM achieves the best performance on the ARG prediction task with an Accuracy of 0.9931, Precision of 1, Recall of 0.9859, F1 of 0.9929, and MCC of 0.9862. Similarly, ARG-LLM outperforms the antibiotic category prediction task with best Accuracy of 0.9232, Precision of 0.9262, Recall of 0.9616, F1 of 0.9435, and MCC of 0.9001.

6.3 Comparison with SOTA Methods

Figure 4 compares ARG-LLM results with SOTA methods like DeepArg (Arango-Argoty et al., 2018b), HMD-ARG (Li et al., 2021), and PLM-ARG (Wu et al., 2023) for Antibiotic category prediction. The referenced studies have not provided the results of ARG prediction, so we are unable to give a comparison of ARG prediction in this section. Also, the referenced research HMD-ARG did not present the MCC value in their work paper; thus, we are unable to include it in our comparison. From the available results, it can be observed that the highest accuracy of 0.948 is achieved by the HMD-ARG model, and the PLM-ARG model achieves the highest precision of 1. However, ARG-LLM achieves the best Recall, F1, and MCC of 0.962, 0.944, and 0.900, respectively. Additionally, ARG-LLM achieves the 2nd highest accuracy of 0.923. Overall, if F1 is taken as a metric, ARG-LLM outperforms other SOTA methods.

Overall, this work aimed to predict ARG and antibiotic resistance categories using an ensemble of finetuned transformer-based LLMs. The experimental results reveal promising performance gains achieved by the ARG-LLM framework. The results from Table 3 show that finetuning the pre-trained LLMs improves their performance in classifying the ARG sequences into their antibiotic categories. Finetuning helps the model to adapt to the specific characteristics of a new, smaller dataset relevant to the target task. Similarly, from Table 2, it is clear that ensembling the three LLMs led to a significant improvement in performance.

7 CONCLUSION

We propose a multi-task ensemble model of finetuned LLMs to leverage the prediction of ARG and then further identify what antibiotic family it is resistant to. The experimental results confirm the reliability of the proposed model in identifying ARGs. The comparison results show that finetuning a pretrained model with a task-specific dataset improves the model's performance. Additionally, ensemble prediction with the fine-tuned LLMs further enhanced the performance of the proposed model. The outcomes of this experimentation have powerful implications for researchers and practitioners engaged in ARG identification tasks. The proposed model can be a powerful tool to alleviate the global threat of antibiotic resistance. In the future, the ARG structural information can be incorporated with the sequence features to improve the performance of the model.

REFERENCES

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410.

Arango-Argoty, G., Garner, E., Pruden, A., Heath, L. S., Vikesland, P., and Zhang, L. (2018a). Deeparg: a deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome*, 6:1–15.

Arango-Argoty, G., Garner, E., Pruden, A., Heath, L. S., Vikesland, P., and Zhang, L. (2018b). Deeparg: a deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome*, 6:1–15.

Bepler, T. and Berger, B. (2021). Learning the protein language: Evolution, structure, and function. *Cell systems*, 12(6):654–669.

Bradley, P., Gordon, N. C., Walker, T. M., Dunn, L., Heys, S., Huang, B., Earle, S., Pankhurst, L. J., Anson, L., De Cesare, M., et al. (2015). Rapid antibiotic-resistance predictions from genome sequence data for staphylococcus aureus and mycobacterium tuberculosis. *Nature communications*, 6(1):10063.

Buchfink, B., Xie, C., and Huson, D. H. (2015). Fast and sensitive protein alignment using diamond. *Nature methods*, 12(1):59–60.

Chowdhury, A. S., Call, D. R., and Broschat, S. L. (2019). Antimicrobial resistance prediction for gram-negative bacteria via game theory-based feature evaluation. *Scientific reports*, 9(1):14487.

Consortium, U. (2015). Uniprot: a hub for protein information. *Nucleic acids research*, 43(D1):D204–D212.

Davis, J. J., Boisvert, S., Brettin, T., Kenyon, R. W., Mao, C., Olson, R., Overbeek, R., Santerre, J., Shukla, M., Wattam, A. R., et al. (2016). Antimicrobial resistance prediction in patric and rast. *Scientific reports*, 6(1):27930.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. (2021). Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing.

Feldgarden, M., Brover, V., Haft, D. H., Prasad, A. B., Slotta, D. J., Tolstoy, I., Tyson, G. H., Zhao, S., Hsu, C.-H., McDermott, P. F., et al. (2019). Validating the amrfinder tool and resistance gene database by using antimicrobial resistance genotype-phenotype correlations in a collection of isolates. *Antimicrobial agents and chemotherapy*, 63(11):10–1128.

Gibson, M. K., Forsberg, K. J., and Dantas, G. (2015). Improved annotation of antibiotic resistance determinants reveals microbial resistomes cluster by ecology. *The ISME journal*, 9(1):207–216.

Jia, B., Raphenya, A. R., Alcock, B., Waglechner, N., Guo, P., Tsang, K. K., Lago, B. A., Dave, B. M., Pereira, S., Sharma, A. N., et al. (2016). Card 2017: expansion and model-centric curation of the comprehensive antibiotic resistance database. *Nucleic acids research*, page gkw1004.

Kleinheinz, K. A., Joensen, K. G., and Larsen, M. V. (2014). Applying the resfinder and virulencefinder web-services for easy identification of acquired antibiotic resistance and e. coli virulence genes in bacteriophage and prophage nucleotide sequences. *Bacteriophage*, 4(2):e27943.

Lakin, S. M., Dean, C., Noyes, N. R., Dettenwanger, A., Ross, A. S., Doster, E., Rovira, P., Abdo, Z., Jones, K. L., Ruiz, J., et al. (2017). Megares: an antimicrobial resistance database for high throughput sequencing. *Nucleic acids research*, 45(D1):D574–D580.

Lázár, V. and Kishony, R. (2019). Transient antibiotic resistance calls for attention. *Nature microbiology*, 4(10):1606–1607.

Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with burrows–wheeler transform. *bioinformatics*, 25(14):1754–1760.

Li, Y., Wang, S., Umarov, R., Xie, B., Fan, M., Li, L., and Gao, X. (2018). Deepre: sequence-based enzyme ec number prediction by deep learning. *Bioinformatics*, 34(5):760–769.

Li, Y., Xu, Z., Han, W., Cao, H., Umarov, R., Yan, A., Fan, M., Chen, H., Duarte, C. M., Li, L., et al. (2021). Hmd-arg: hierarchical multi-task deep learning for annotating antibiotic resistance genes. *Microbiome*, 9:1–12.

Mao, D., Yu, S., Rysz, M., Luo, Y., Yang, F., Li, F., Hou, J., Mu, Q., and Alvarez, P. (2015). Prevalence and proliferation of antibiotic resistance genes in two municipal wastewater treatment plants. *Water research*, 85:458–466.

McArthur, A. G. and Tsang, K. K. (2017). Antimicrobial resistance surveillance in the genomic age. *Annals of the New York Academy of Sciences*, 1388(1):78–91.

McArthur, A. G., Waglechner, N., Nizam, F., Yan, A., Azad, M. A., Baylay, A. J., Bhullar, K., Canova, M. J., De Pascale, G., Ejim, L., et al. (2013). The comprehensive antibiotic resistance database. *Antimicrobial agents and chemotherapy*, 57(7):3348–3357.

Mendelson M, M. M. (2015). The world health organization global action plan for antimicrobial resistance. *S Afr Med J*, 105(5):325.

Miah, M. S. U., Kabir, M. M., Sarwar, T. B., Safran, M., Alfarhood, S., and Mridha, M. (2024). A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and llm. *Scientific Reports*, 14(1):9603.

Murray, C. J., Ikuta, K. S., Sharara, F., Swetschinski, L., Aguilar, G. R., Gray, A., Han, C., Bisignano, C., Rao, P., Wool, E., et al. (2022). Global burden of bacterial antimicrobial resistance in 2019: a systematic analysis. *The lancet*, 399(10325):629–655.

Pehrsson, E. C., Tsukayama, P., Patel, S., Mejía-Bautista, M., Sosa-Soto, G., Navarrete, K. M., Calderon, M., Cabrera, L., Hoyos-Arango, W., Bertoli, M. T., et al. (2016). Interconnected microbiomes and resistomes in low-income human habitats. *Nature*, 533(7602):212–216.

Pham, V. H. and Kim, J. (2012). Cultivation of unculturable soil bacteria. *Trends in biotechnology*, 30(9):475–484.

Rao, R. M., Liu, J., Verkuil, R., Meier, J., Canny, J. F., Abbeel, P., Sercu, T., and Rives, A. (2021). Transformer protein language models are unsupervised structure learners. *bioRxiv*.

Ruppé, E., Ghozlane, A., Tap, J., Pons, N., Alvarez, A.-S., Maziers, N., Cuesta, T., Hernando-Amado, S., Clares, I., Martínez, J. L., et al. (2019). Prediction of the intestinal resistome by a three-dimensional structure-based method. *Nature microbiology*, 4(1):112–123.

Steinegger, M. and Söding, J. (2018). Clustering huge protein sequence sets in linear time. *Nature communications*, 9(1):2542.

Suzek, B. E., Wang, Y., Huang, H., McGarvey, P. B., Wu, C. H., and Consortium, U. (2015). Uniref clusters: a comprehensive and scalable alternative for improving sequence similarity searches. *Bioinformatics*, 31(6):926–932.

Taylor, W. L. (1953). “cloze procedure”: A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, Z., Li, S., You, R., Zhu, S., Zhou, X. J., and Sun, F. (2021). Arg-shine: improve antibiotic resistance class prediction by integrating sequence homology, functional information and deep convolutional neural network. *NAR Genomics and Bioinformatics*, 3(3):lqab066.

Wu, J., Ouyang, J., Qin, H., Zhou, J., Roberts, R., Siam, R., Wang, L., Tong, W., Liu, Z., and Shi, T. (2023). Plm-arg: antibiotic resistance gene identification using a pretrained protein language model. *Bioinformatics*, 39(11):btad690.

Zankari, E., Hasman, H., Cosentino, S., Vestergaard, M., Rasmussen, S., Lund, O., Aarestrup, F. M., and Larsen, M. V. (2012). Identification of acquired antimicrobial resistance genes. *Journal of antimicrobial chemotherapy*, 67(11):2640–2644.

Approaches for Extending Recommendation Models for Food Choices in Meals

Nguyen Thi Hong Nhung[1,2], Dao Khoa Nguyen[1,2], Tiet Gia Hong[1,2,*] and Thi My Hang Vu[1,2]
[1]*Faculty of Information Technology, University of Science, Ho Chi Minh City, Vietnam*
[2]*Vietnam National University, Ho Chi Minh City, Vietnam*
{20120093, 20120147}@student.hcmus.edu.vn, {tghong, vtmhang}@fit.hcmus.edu.vn

Keywords: Food Recommender System, Neighbor-Based Recommendation, Latent Factor-Based Recommendation.

Abstract: In this paper, we propose food recommender systems based on users' historical food choices. Their advantage lies in providing personalized food suggestions for each user considering each meal. These systems are developed using two popular recommendation principles: neighbor-based and latent factor-based. In the neighbor-based model, the system aggregates the food choices of neighboring users to recommend food choices for the active user during the considered meal. In contrast, the latent factor-based model constructs and optimizes an objective function to learn positive representations of users, foods, and meals. In this new space, predicting users' food choices during meals becomes straightforward. Experimental results have demonstrated the effectiveness of the proposed models in specific cases. However, in a global statistical comparison, the latent factor-based model has proven to be more effective than the neighbor-based model.

1 INTRODUCTION

Recommender systems are increasingly playing an important role on digital platforms. On YouTube and Netflix, they help suggest videos that match users' past viewing experiences (Amatriain and Basilico, 2015; Hong and Kim, 2016). Users on social networks are assisted by recommender systems in finding suitable friends (Ahmadian et al., 2020). On Amazon, thanks to recommender systems, users can quickly and accurately find desired items (Smith and Linden, 2017). Moreover, researchers are expanding traditional recommender systems to provide recommendations for groups of users (Nam, 2021a). As a result, recommender systems can fully meet users' needs, from individual preferences to group preferences.

In this study, we focus on a specific domain of recommender systems, which is food recommendation. Many previous food recommender systems have aimed to provide the most optimal recommendations by suggesting foods that users are predicted to like after trying them (Twomey et al., 2020; Jia et al., 2022; Hamdollahi et al., 2023; Bondevik et al., 2023). Such recommender systems are trained using preference data, which consists of ratings given by users after trying the foods. The rating scale is typically diverse, ranging from "dislike very much" to "like very much". Therefore, it is difficult for users to provide ratings that accurately reflect their feelings about foods (Shen et al., 2019; Vy et al., 2024). Collecting a large and accurate number of ratings for food recommender systems requires significant cost and time. Hence, our study aims to propose a more neutral recommendation solution by suggesting foods that users are likely to choose. For these systems, the underlying training data is much easier to collect, as it consists of users' food choice history.

Within the scope of this study, the distinguishing feature is considering meal information in the food recommendation process. Meal information directly influences users' food choices; for instance, users might choose a pastry for breakfast but not for lunch. Therefore, taking into account the user-food-meal correlation is more suitable for food recommendation systems compared to traditional models such as neighbor-based (Aggarwal, 2016) and latent factor-based recommendations (Nam, 2021b), which only address the user-product correlation during the recommendation process.

[*] Corresponding Author

Nhung, N., Nguyen, D., Hong, T. and Vu, T.
Approaches for Extending Recommendation Models for Food Choices in Meals.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 113-121
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Specifically, our contributions are as follows:

- We extend two typical user-product recommendation models, namely neighbor-based and latent factor-based models, to achieve user-food-meal recommendation models.
- We conduct experiments to conclude the suitability of the neighbor-based and latent factor-based models for the user-food-meal recommendation problem.

The structure of the paper is as follows. In section 2, we analyze some limitations of previous studies on food recommendation. In section 3, we propose approaches to address these limitations. In section 4, experiments are conducted to evaluate the proposed approaches. Finally, we present the conclusions and future works.

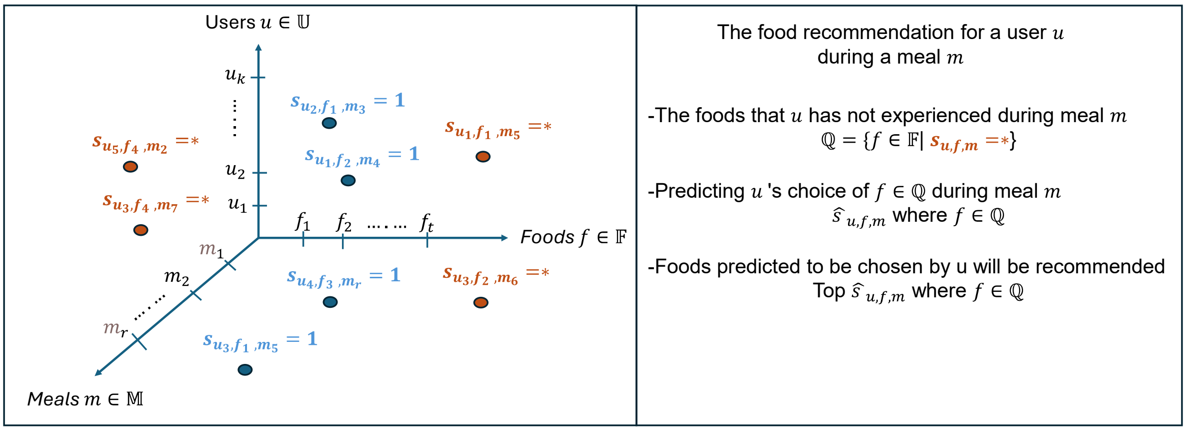
2 RELATED WORKS

The core of food recommender systems is predicting a user's preference for a food, and then recommending the foods predicted to be the most liked. To achieve this, previous studies (Twomey et al., 2020; Jia et al., 2022) have utilized the user's past food preferences and the descriptions of the foods to estimate how much the user would like a particular food. (Hamdollahi et al., 2023) also incorporate user descriptions and food images to predict food preferences.

One approach defines a similarity measure between the user vector and the food vector, recommending the food most similar to the user. To design this similarity measure, some studies use TF-IDF and cosine measures (Chhipa et al., 2022; Padmavathi et al., 2023), while others use Positive Pointwise Mutual Information (PPMI) (Teng et al., 2012; Zhang et al., 2022). Another approach (Mokdara et al., 2018) applies matrix factorization to learn features for representing both foods and users. This feature space facilitates the estimation of the compatibility between users and foods. Researchers improve the quality of this feature learning process by incorporating user tags (Ge et al., 2015). Another approach to matching users and foods is to use health rules combined with users' past preferences in certain contexts (Agapito et al, 2018; Vairale and Shukla, 2021).

It can be seen that previous studies have relied on users' past food preferences, typically indicated by a rating score ranging from 1 to 5, collected after users have experienced the foods. Due to this nature, the number of ratings collected is often very low, and the accuracy of these ratings is frequently not high (Vy et al., 2024). Evidence of this is apparent on platforms like Amazon, where users may leave highly positive textual reviews about an item but assign a low rating score, and vice versa (Shen et al., 2019). This discrepancy arises because users may not fully grasp the correlation between their preferences and the numerical rating scale, leading to ratings that do not accurately depict their true experience with the foods.

Furthermore, a variety of additional information is utilized to enhance the accuracy of predicting users' food preferences. This includes food descriptions, nutritional principles, health considerations, and more (Gao et al, 2019; Zhang et al., 2022; Oskouei and Hashemzadeh, 2023). However, it is not always feasible to comprehensively collect all such information. Moreover, the use of excessive additional information can also reduce the flexibility of the system.



The food recommendation for a user u during a meal m

-The foods that u has not experienced during meal m
$\mathbb{Q} = \{f \in \mathbb{F} | s_{u,f,m} =*\}$

-Predicting u 's choice of $f \in \mathbb{Q}$ during meal m
$\hat{s}_{u,f,m}$ where $f \in \mathbb{Q}$

-Foods predicted to be chosen by u will be recommended
Top $\hat{s}_{u,f,m}$ where $f \in \mathbb{Q}$

Figure 1: The user-food-meal recommendation problem.

Given the limitations identified above, this paper proposes food recommendation models that rely solely on the easiest-to-collect information: users' food choice history. To better reflect real-world scenarios, users' food choices will be detailed for each meal. Detailed descriptions of the user-food-meal recommendation problem are provided in Subsection 3.1.

Collaborative filtering is one of the effective models for achieving good recommendations. The term "collaborative" means utilizing community data to provide recommendations of items to users. Its two typical models are neighbor-based (Aggarwal, 2016) and latent factor-based (Nam, 2021b). As mentioned earlier, our recommendation model not only involves users and foods but also meals. Therefore, our motivation is to extend these two user-product collaborative filtering models to user-food-meal recommendation models. Details of this extension will be presented in Subsections 3.2 and 3.3

3 OUR PROPOSED APPROACHES

3.1 User-Food-Meal Recommendation Problem

Fig. 1 illustrates the user-food-meal recommendation problem addressed in this paper. Specifically, data on users' food choices during meals is collected. If a user $u \in \mathbb{U} = \{u_1, u_2, \dots, u_k\}$ chooses a food $f \in \mathbb{F} = \{f_1, f_2, \dots, f_t\}$ during a meal $m \in \mathbb{M} = \{m_1, m_2, \dots, m_r\}$, the corresponding value is 1, denoted by $s_{u,f,m} = 1$. For an active user u seeking food recommendations during a meal m, the choices of u for foods f not yet experienced in meal m need to be predicted ($s_{u,f,m} =*$). Foods predicted to be chosen by the active user will be recommended. Table 1 presents the symbols used to describe the proposed approaches in Subsections 3.2 and 3.3.

Table 1: The symbols.

Symbol	Description
$u \in \mathbb{U} = \{u_1, u_2, \dots, u_k\}$	User
$f \in \mathbb{F} = \{f_1, f_2, \dots, f_t\}$	Food
$m \in \mathbb{M} = \{m_1, m_2, \dots, m_r\}$	Meal
$s_{u,f,m} = 1$	User u has chosen food f during meal m
$s_{u,f,m} =*$	User u has not chosen food f during meal m
$\hat{s}_{u,f,m}$	Predicting user u 's choice of food f during meal m
$sim(u^{(m)}, u'^{(m')})$	The similarity between user u in meal m and user u' in meal m'
k	The number of selected neighbors in neighbor-based models
$\mathbb{N}_{u^m}$	Top k $u'^{(m')}$ similar to $u^{(m)}$
$\mathbb{H}_f$	Set of $u'^{(m')}$ who have chosen f in the past
z	The number of latent factors in latent-factor-based models
$a_{u,1}, a_{u,2}, \dots, a_{u,z}$	Representations of user $u \in \mathbb{U}$ under z latent factors
$b_{f,1}, b_{f,2}, \dots, b_{f,z}$	Representation of food $f \in \mathbb{F}$ under z latent factors
$c_{m,1}, c_{m,2}, \dots, c_{m,z}$	Representations of meal $m \in \mathbb{M}$ under z latent factors
λ	Tikhonov regularization weight
$\varphi_{u,1}, \varphi_{u,2}, \dots, \varphi_{u,z}$	Learning rates of user $u \in \mathbb{U}$ under z latent factors
$\varphi_{f,1}, \varphi_{f,2}, \dots, \varphi_{f,z}$	Learning rates of food $f \in \mathbb{F}$ under z latent factors
$\varphi_{m,1}, \varphi_{m,2}, \dots, \varphi_{m,z}$	Learning rates of meal $m \in \mathbb{M}$ under z latent factors

3.2 Neighbor-Based Model for User-Food-Meal Recommendation (NUFM)

The principle of the neighbor-based model is to recommend products that users similar to the active user have liked in the past (Aggarwal, 2016; Vy et al, 2024). In this section, we refine this principle to address the problem of recommending foods to users during meals, namely NUFM.

Specifically, for the offline phase, we implement the calculation of the similarity in food choices between each pair of users $u \in \mathbb{U} = \{u_1, u_2, \dots, u_k\}$ considering each pair of meals $m \in \mathbb{M} = \{m_1, m_2, \dots, m_r\}$. With collected data on food choices during meals, a Jaccard similarity (Bag et al., 2019) is suitable for this case. Accordingly, the more common food choices user u in meal m ($u^{(m)}$) and user u' in meal m' ($u'^{(m')}$), the higher their similarity

($sim(u^{(m)}, u'^{(m')}$). Specifically, the formula is implemented as follows:

$$sim\left(u^{(m)}, u'^{(m')}\right) = \frac{\left|\{f \mid s_{u,f,m} = 1 \wedge s_{u',f,m'} = 1\}\right|}{\left|\{f \mid s_{u,f,m} = 1 \vee s_{u',f,m'} = 1\}\right|} \quad (1)$$

In the online phase, the prediction of user u's choice of food f during meal m is as follows:

- Based on the similarity scores computed during the offline phase, the set of top k $u'^{(m')}$ similar to $u^{(m)}$: $\mathbb{N}_{u^{(m)}}^{\square}$.
- Get the set of $u'^{(m')}$ who have chosen f: $\mathbb{H}_f^{\square}$
- Predicting user u's choice of food f during meal m ($\hat{s}_{u,f,m}$) by computing the sum of similarities between $u'^{(m')}$ ($u'^{(m')} \in \mathbb{N}_{u^{(m)}}^{\square} \cap \mathbb{H}_f^{\square}$) and $u^{(m)}$, as follows:

$$\hat{s}_{u,f,m} = \sum_{u'^{(m')} \in \mathbb{N}_{u^{(m)}}^{\square} \cap \mathbb{H}_f^{\square}}^{\square} sim(u'^{(m')}, u^{(m)}) \quad (2)$$

If $\hat{s}_{u,f,m}$ is higher, it indicates that users similar to u often choose food f for meal m.

The drawback of the above approach is the high computation time required for calculating similarities in the offline phase, especially as the number of users and meals grows. Consequently, we propose parallel computation using Hadoop for the similarity calculation described above. Specifically, on Hadoop, the users' food choice data will be partitioned into smaller fragments corresponding to each food $f \in \mathbb{F} = \{f_1, f_2, \dots, f_t\}$, as follows:

$$\begin{gathered} f_1; {u_1}^{(m_1)}, {u_2}^{(m_3)}, {u_3}^{(m_4)} \\ f_2; {u_1}^{(m_2)}, {u_1}^{(m_3)}, {u_2}^{(m_4)} \\ f_3; {u_2}^{(m_1)}, {u_3}^{(m_1)}, {u_2}^{(m_2)} \\ \dots\dots \end{gathered} \quad (3)$$

where the right side represents $u^{(m)}$ chosen the left food $f \in \mathbb{F} = \{f_1, f_2, \dots, f_t\}$.

In each partitioned fragment, parallel computations are executed using mapping functions. Specifically, the mapping function generates (key; value) elements where the keys represent pairs of users who have both selected the food (denoted as $\_q$), pairs where only one user has selected the food (denoted as $\_p$), and the values are set to 1. For example, with the fragment corresponding to food f_1 ($f_1; {u_1}^{(m_1)}, {u_2}^{(m_3)}, {u_3}^{(m_4)}$) , (key; value) elements after the mapping function will be as follows:

$$\begin{gathered} ({u_1}^{(m_1)}\_{u_2}^{(m_3)}\_q; 1) \\ ({u_1}^{(m_1)}\_{u_3}^{(m_4)}\_q; 1) \\ ({u_2}^{(m_3)}\_{u_3}^{(m_4)}\_q; 1) \\ ({u_1}^{(m_1)}\_{u_2}^{(m_4)}\_p; 1) \\ ({u_1}^{(m_1)}\_{u_2}^{(m_5)}\_p; 1) \\ ({u_1}^{(m_1)}\_{u_2}^{(m_1)}\_p; 1) \\ \dots\dots \end{gathered} \quad (4)$$

After all mapping functions are completed, a reducing function is executed to compute the sum of values with the same key. For example, to compute $sim(u^{(m)}, u'^{(m')})$ as in Eq. (1), the sum of values with the same key $u^{(m)}\_u'^{(m')}\_q$ serves as the numerator, while the sum of values with the same key $u^{(m)}\_u'^{(m')}\_p$ serves as the denominator.

3.3 Latent-Factor-Based Model for User-Food-Meal Recommendation (LUFM)

The latent factor model aims to find the compatibility between users and products in a latent factor space to decide whether to recommend products to users (Shen et al., 2019; Nam, 2021a). Accordingly, given that the entities involved in our problem are users, foods, and meals, our model, namely LURM, needs to learn their representations under z latent factors, denoted as $a_{u,1}$, $a_{u,2}, \dots, a_{u,z}$ for each user $u \in \mathbb{U}$, $b_{f,1}, b_{f,2}, \dots, b_{f,z}$ for each food $f \in \mathbb{F}$, and $c_{m,1}, c_{m,2}, \dots, c_{m,z}$ for each meal $m \in \mathbb{M}$. At this point, user u's choice of food f during meal m ($\hat{s}_{u,f,m}$) will depend on the alignment of three latent-factor-based representations, as follows:

$$\hat{s}_{u,f,m} = \sum_{j=1}^{z} \left(a_{u,j}.b_{f,j} + a_{u,j}.c_{m,j} + c_{m,j}.b_{f,j}\right) \quad (5)$$

Fig. 2 illustrates the process in LUFM.

In the LUFM, the latent-factor-based representations for users, foods, and meals are optimized to minimize the distance between actual and predicted values, as follows:

$$\begin{gathered} \min_{\square} \frac{1}{2} \sum_{s_{u,f,m}} \left(\hat{s}_{u,f,m} - s_{u,f,m}\right)^2 \square \\ \Leftrightarrow \\ \min_{\square} \frac{1}{2} \sum_{s_{u,f,m}} \left(\sum_{j=1}^{j=z} \left(\begin{matrix} a_{u,j}.b_{f,j} + a_{u,j}.c_{m,j} \\ +c_{m,j}.b_{f,j} \end{matrix} \right) - s_{u,f,m} \right)^2 \end{gathered} \quad (6)$$

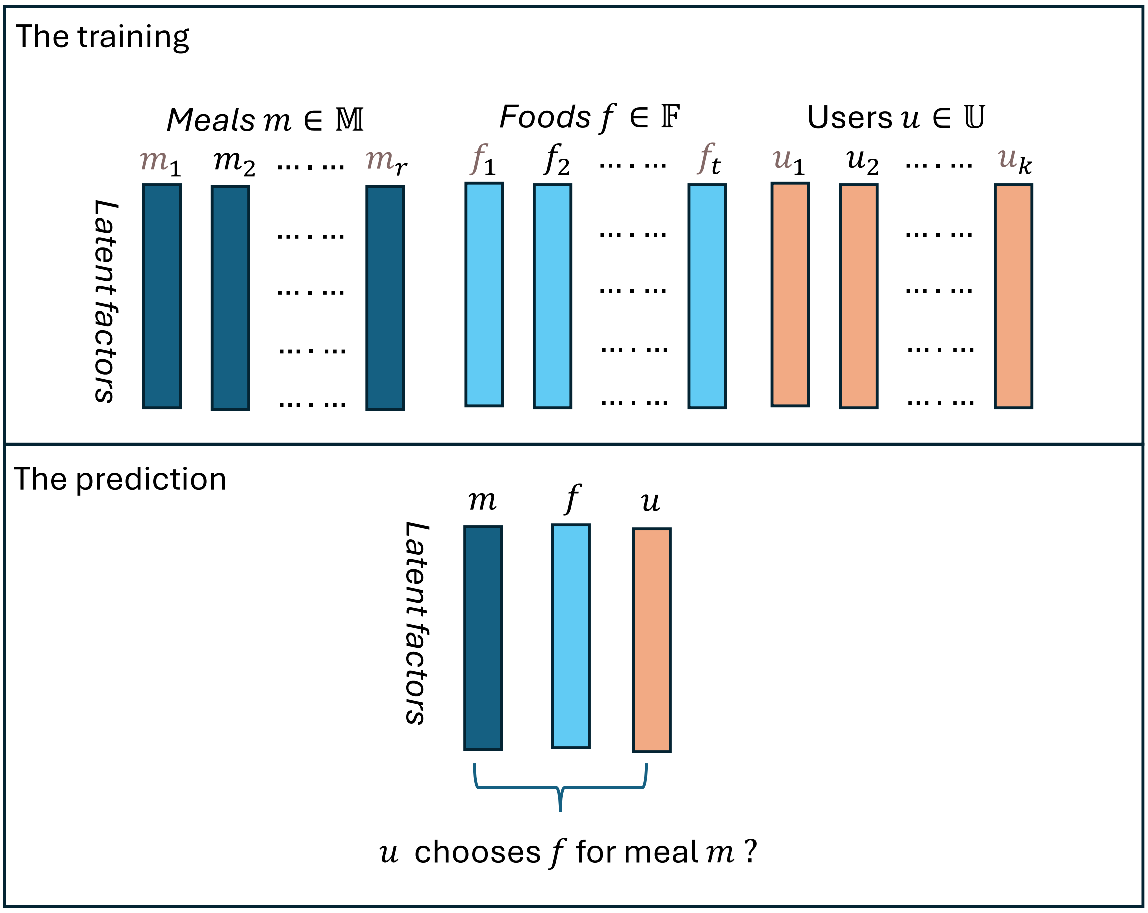


Figure 2: Our proposed approach, LUFM.

To enhance the semantic meaning of the latent-factor-based presentations, we enforce a constraint that their components are always positive. This constraint creates a meaningful part-based representation (Chen et al., 2021; Salahian et al., 2023). Additionally, to prevent overfitting, we add a Tikhonov regularization term (Nam, 2021a; Vy et al., 2024) to the objective function with a weight λ. Finally, the objective function will be rewritten as follows:

$$\min \frac{1}{2}\sum_{s_{u,f,m}} \left(\widehat{s}_{u,f,m} - s_{u,f,m}\right)^2 + \frac{\lambda}{2}\left(\sum_{u\in\mathbb{U}}\sum_{j=1}^{j=z} a_{u,j}^2 + \sum_{f\in\mathbb{F}}\sum_{j=1}^{j=z} b_{f,j}^2 + \sum_{m\in\mathbb{M}}\sum_{j=1}^{j=z} c_{m,j}^2\right) \quad (7)$$

Subject to positive parameters:

$$a_{u,j} \geq 0, b_{f,j} \geq 0, c_{m,j} \geq 0$$

$$\forall j = 1 \dots z,\ \ \forall u \in \mathbb{U},\ \ \forall f \in \mathbb{F},\ \ \forall m \in \mathbb{M}$$

To optimize Eq. (7), this paper employs Stochastic Gradient Descent (SGD). Specifically, SGD first sets up the objective function at a data point $s_{u,f,m}$, denoted by $V(u,f,m)$. Subsequently, partial derivatives of $V(u,f,m)$ concerning each parameter will be computed as follows:

$$V(u,f,m) = \frac{1}{2}\left(\widehat{s}_{u,f,m} - s_{u,f,m}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{j=z}\left(a_{u,j}^2 + b_{f,j}^2 + c_{m,j}^2\right)$$

$\Leftrightarrow$

$$V(u,f,m) = \frac{1}{2}\left(\sum_{j=1}^{j=z}\left(a_{u,j}.b_{f,j} + a_{u,j}.c_{m,j} + c_{m,j}.b_{f,j}\right) - s_{u,f,m}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{j=z}\left(a_{u,j}^2 + b_{f,j}^2 + c_{m,j}^2\right) \quad (8)$$

$$\forall j = 1 \dots z\colon \frac{\partial V(u,f,m)}{\partial a_{u,j}} = \left(b_{f,j} + c_{m,j}\right)\left(\hat{s}_{u,f,m} - s_{u,f,m}\right) + \lambda . a_{u,j} \tag{9}$$

$$j = 1 \dots z\colon \frac{\partial V(u,f,m)}{\partial b_{f,j}} = \left(a_{u,j} + c_{m,j}\right)\left(\hat{s}_{u,f,m} - s_{u,f,m}\right) + \lambda . b_{f,j} \tag{10}$$

$$j = 1 \dots z \; \frac{\partial V(u,f,m)}{\partial c_{m,j}} = \left(a_{u,j} + b_{f,j}\right)\left(\hat{s}_{u,f,m} - s_{u,f,m}\right) + \lambda . c_{m,j} \tag{11}$$

These partial derivatives will be used to update the corresponding parameters with learning rates $\varphi_{u,j}$, $\varphi_{f,j}$, $\varphi_{m,j}$ $j = 1 \dots z$, as follows:

$$\begin{aligned} &\forall j = 1 \dots z\colon a_{u,j} \leftarrow a_{u,j} - \varphi_{u,j} . \frac{\partial V(u,f,m)}{\partial a_{u,j}} \\ &\Leftrightarrow \\ &a_{u,j} \leftarrow a_{u,j} + \varphi_{u,j} . \left(b_{f,j} + c_{m,j}\right) . s_{u,f,m} \\ &\qquad - \varphi_{u,j} . \left(\left(b_{f,j} + c_{m,j}\right) . \hat{s}_{u,f,m} + \lambda . a_{u,j}\right) \end{aligned} \tag{12}$$

$$\begin{aligned} &\forall j = 1 \dots z\colon b_{f,j} \leftarrow b_{f,j} - \varphi_{f,j} . \frac{\partial V(u,f,m)}{\partial b_{f,j}} \\ &\Leftrightarrow \\ &b_{f,j} \leftarrow b_{f,j} + \varphi_{f,j} . \left(a_{u,j} + c_{m,j}\right) . s_{u,f,m} \\ &\qquad - \varphi_{f,j} . \left(\left(a_{u,j} + c_{m,j}\right) . \hat{s}_{u,f,m} + \lambda . b_{f,j}\right) \end{aligned} \tag{13}$$

$$\begin{aligned} &\forall j = 1 \dots z\colon c_{m,j} \leftarrow c_{m,j} - \varphi_{m,j} . \frac{\partial V(u,f,m)}{\partial c_{m,j}} \\ &\Leftrightarrow \\ &c_{m,j} \leftarrow c_{m,j} + \varphi_{m,j} . \left(a_{u,j} + b_{f,j}\right) . s_{u,f,m} \\ &\qquad - \varphi_{m,j} . \left(\left(a_{u,j} + b_{f,j}\right) . \hat{s}_{u,f,m} + \lambda . c_{m,j}\right) \end{aligned} \tag{14}$$

To ensure all parameters remain positive, the learning rates $\varphi_{u,j}$, $\varphi_{f,j}$, $\varphi_{m,j}$ $j = 1 \dots z$ must be set to eliminate negative components from Eqs. (12-14), as in (Luo et al., 2014), as follows:

$$\forall j = 1 \dots z\colon \varphi_{u,j} = \frac{a_{u,j}}{\left(b_{f,j} + c_{m,j}\right) . \hat{s}_{u,f,m} + \lambda . a_{u,j}} \tag{15}$$

$$\forall j = 1 \dots z\colon \varphi_{f,j} = \frac{b_{f,j}}{\left(a_{u,j} + c_{m,j}\right) . \hat{s}_{u,f,m} + \lambda . b_{f,j}} \tag{16}$$

$$\forall j = 1 \dots z\colon \varphi_{m,j} = \frac{c_{m,j}}{\left(a_{u,j} + b_{f,j}\right) . \hat{s}_{u,f,m} + \lambda . c_{m,j}} \tag{17}$$

Based on Eqs. (15-17), the update process Eqs. (12-14) can be rewritten as follows:

$$\forall j = 1 \dots z\colon a_{u,j} \leftarrow \varphi_{u,j} . \left(b_{f,j} + c_{m,j}\right) . s_{u,f,m} \tag{18}$$

$$\forall j = 1 \dots z\colon b_{f,j} \leftarrow \varphi_{f,j} . \left(a_{u,j} + c_{m,j}\right) . s_{u,f,m} \tag{19}$$

$$\forall j = 1 \dots z\colon c_{m,j} \leftarrow \varphi_{m,j} . \left(a_{u,j} + b_{f,j}\right) . s_{u,f,m} \tag{20}$$

Algorithm 1 presents a detailed description of LFUM

Algorithm 1: The LUFM training and prediction.

```
The training
Initialize $a_{u,j} \geq 0, b_{f,j} \geq 0, c_{m,j} \geq 0$
    $\forall j = 1 \dots z,\ \forall u \in \mathbb{U},\ \forall f \in \mathbb{F},\ \forall m \in \mathbb{M}$
While (Not satisfying the convergence criterion):
  Randomly shuffle ($u \in \mathbb{U}, f \in \mathbb{F}, m \in \mathbb{M}$)
  For each pair $(u, f, m)$:
    $\forall j = 1 \dots z$: Compute $\varphi_{u,j}$, $\varphi_{f,j}$, $\varphi_{m,j}$ based on Eqs. (15-17), respectively.
    $\forall j = 1 \dots z$: Update the latent representations of $u, f, m$ based on based on Eqs. (18-20), respectively
The prediction
    $\hat{s}_{u,f,m} = \sum_{j=1}^{z} \left(a_{u,j} . b_{f,j} + a_{u,j} . c_{m,j} + c_{m,j} . b_{f,j}\right)$
```

4 EXPERIMENT

4.1 Experiment Setup

In this section, we compare our approaches with a recent approach designed for the user-food-meal recommendation problem, as follows:

- NUFM: The neighbor-based model proposed in subsection 3.2 uses the Jaccard similarity between each pair of users considering each pair of meals.
- LUFM: The latent factor model proposed in section 3.3 learns positive latent factors representing users, foods, and meals.
- PPMI: The model for the Positive Pointwise Mutual Information between meals and foods is proposed by (Zhang et al., 2022).

For a fair comparison between approaches NUFM and LUFM, we set the number of neighbors in NUFM equal to the number of latent factors in LUFM. The regularization weight is set to 0.01. The convergence condition in LUFM is set to 500 updates.

4.2 Dataset

The experimental dataset was gathered from MyFitnessPal (MFP), a health and body management application. It details the specific food items chosen by each user for their daily meals. This dataset are presented in Table 2. 80% of the dataset is allocated for training, and the remaining 20% is used for testing to evaluate the system recommendations.

Table 2: Experimental dataset, MyFitnessPal https://www.kaggle.com/datasets/zvikinozadze/myfitnesspal-dataset.

Number of meals	Number of users	Number of foods	Number of food choices
6	9873	953296	5411275

4.3 Measurement

The F1-score is used to evaluate the accuracy of the recommendation results. It is calculated based on precision and recall as follows:

$$F1 - score = \frac{2. Precision. Recall}{Precision + recall} \quad (21)$$

To calculate precision and recall, the recommendation set ($\mathbb{T}_u$) and the correct set ($\mathbb{C}_u$) must be formed. The recommendation set consists of the top foods with the highest predicted values, while the correct set consists of the foods that users have chosen in the test set. Precision is the ratio of correct recommendations to the total number of recommended foods. Recall is the ratio of correct recommendations to the total number of correct foods, as follows:

$$precision = \frac{\sum_{u=1}^{m} |\mathbb{T}_u \cap \mathbb{C}_u|}{\sum_{u=1}^{m} |\mathbb{T}_u|}$$
$$recall = \frac{\sum_{u=1}^{m} |\mathbb{T}_u \cap \mathbb{C}_u|}{\sum_{u=1}^{m} |\mathbb{C}_u|} \quad (22)$$

4.4 Experiment Result and Discussion

Fig. 3 shows the comparison results between NUFM and LUFM. It can be seen that the neighbor-based model (NUFM) performs better than the latent factor-based model (LUFM) when the number of neighbors (which is also the number of latent factors) is set to a small value. This is because, with a small number of latent factors, the latent vectors are insufficient to fully represent the characteristics of users, foods, and meals. However, the performance of the latent factor-based model improves significantly as the number of latent factors increases. Evidence of this is that the recommendation performance of LUFM not only improves but also gradually surpasses that of NUFM. In practice, the number of neighbors or latent factors is determined by the computational power of the device. When computational capacity is limited and high accuracy is not required, these numbers are usually kept small, and vice versa.

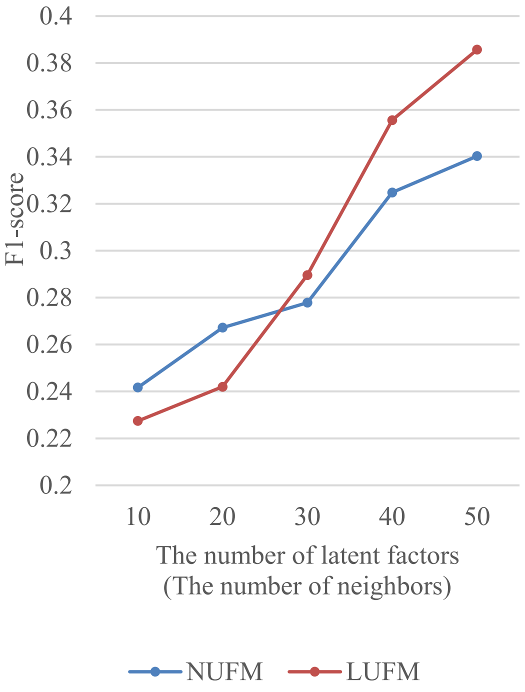


Figure 3: F1-score with a recommendation set size of 15.

Next, we fixed LUFM and NUFM at 45 latent factors and neighbors. As shown in Fig. 4, LUFM and NUFM consistently provide better recommendation results than PPMI across all sizes of the recommendation set. Specifically, when the size of the recommendation set is 10, the F1-score of LUFM and NUFM increases by 0.139 and 0.074 over PPMI, respectively.

Finally, to achieve more convincing conclusions, we conducted statistical t-test comparisons. The input sample for these comparisons consists of the F1-score results measured at the individual user level, instead of a single F-score result at the system level as shown in the previous experiments. The results in Table 3 indicate that LUFM provides the best statistical outcome compared to NUFM and PPMI, as all p-values are less than 0.05. Additionally, for LUFM, in

Table 4, we also performed a statistical comparison between it and a version of it that excludes positive constraints during training. In this comparison, the lack of positive constraints reduced the F1-score compared to when positive constraints were applied. This demonstrates that the positive constraints and the optimization method with these constraints are reasonable and suitable for the problem.

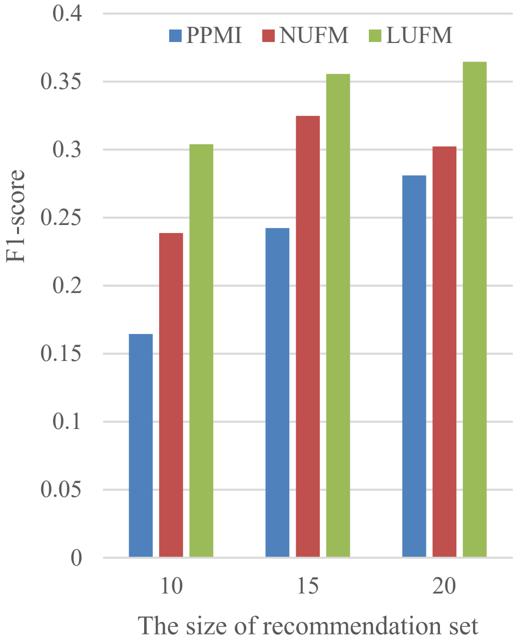


Figure 4: F1-score at 45 latent factors (neighbors).

Table 3: The t-test comparison between NUFM, LUFM, and PPMI.

Approach	NUFM >> PPMI	LUFM >> PPMI	LUFM >> NUFM
Sample mean	0.296 >> 0.270	0.337 >> 0.270	0.337 >> 0.296
p-value	0.0049	0.0001	0.0068

Table 4: The t-test comparison between LUFM with positive constraints and LUFM without positive constraints.

Approach	LUFM with positive constraints >> LUFM without positive constraints
Sample mean	0.337 >> 0.308
p-value	0.0072

5 CONCLUSION

In this paper, we extend two typical recommendation models, namely the neighbor-based model and the latent-factor-based model, to address the user-food-meal recommendation problem. Specifically, for the neighbor-based model, a similarity measure between pairs of users for each pair of meals is proposed using the Jaccard principle, while the positive latent-factor-based model for the user-food-meal recommendations is also implemented. Experiments have shown that the neighbor-based model performs better than the latent-factor-based model when the number of neighbors, which is also the number of latent factors, is set to a low value. As this number increases, the latent-factor-based model yields better results. However, overall, the latent factor-based model provides statistically better results than the neighbor-based model.

Our research focuses solely on the most basic data, which is users' food choice history. However, food choices also depend on various other factors such as nutrition, health, and so forth. Accurate recommendations based on food choice data are a crucial foundation for integrating additional factors in building a comprehensive method in the future.

ACKNOWLEDGEMENTS

This research is funded by University of Science, VNU-HCM under grant number CNTT 2024-05.

REFERENCES

Agapito, G., Simeoni, M., Calabrese, B., Caré, I., Lamprinoudi, T., Guzzi, P. H., ... & Cannataro, M. (2018). DIETOS: A dietary recommender system for chronic diseases monitoring and management. Computer methods and programs in biomedicine, 153, 93-104.

Aggarwal, C. C. (2016). Neighborhood-based collaborative filtering. Recommender Systems: The Textbook, 29-70.

Ahmadian, S., Joorabloo, N., Jalili, M., Ren, Y., Meghdadi, M., & Afsharchi, M. (2020). A social recommender system based on reliable implicit relationships. Knowledge-Based Systems, 192, 105371.

Amatriain, X., & Basilico, J. (2015). Recommender systems in industry: A netflix case study. In Recommender systems handbook (pp. 385-419). Boston, MA: Springer US.

Bag, S., Kumar, S. K., & Tiwari, M. K. (2019). An efficient recommendation generation using relevant Jaccard similarity. Information Sciences, 483, 53-64.

Bondevik, J. N., Bennin, K. E., Babur, Ö., & Ersch, C. (2023). A systematic review on food recommender systems. Expert Systems with Applications, 122166.

Chen, Z., Jin, S., Liu, R., & Zhang, J. (2021). A deep non-negative matrix factorization model for big data representation learning. Frontiers in Neurorobotics, 15, 701194.

Chhipa, S., Berwal, V., Hirapure, T., & Banerjee, S. (2022). Recipe recommendation system using TF-IDF. In ITM web of conferences (Vol. 44, p. 02006). EDP Sciences.

Gao, X., Feng, F., He, X., Huang, H., Guan, X., Feng, C., ... & Chua, T. S. (2019). Hierarchical attention network for visually-aware food recommendation. IEEE Transactions on Multimedia, 22(6), 1647-1659.

Ge, M., Elahi, M., Fernaández-Tobías, I., Ricci, F., & Massimo, D. (2015, May). Using tags and latent factors in a food recommender system. In Proceedings of the 5th international conference on digital health 2015 (pp. 105-112).

Hamdollahi Oskouei, S., & Hashemzadeh, M. (2023). FoodRecNet: a comprehensively personalized food recommender system using deep neural networks. Knowledge and Information Systems, 65(9), 3753-3775.

Hong, S. E., & Kim, H. J. (2016, July). A comparative study of video recommender systems in big data era. In 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN) (pp. 125-127). IEEE.

Jia, N., Chen, J., & Wang, R. (2022). An attention-based convolutional neural network for recipe recommendation. Expert Systems with Applications, 201, 116979.

Luo, X., Zhou, M., Xia, Y., & Zhu, Q. (2014). An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. IEEE Transactions on Industrial Informatics, 10(2), 1273-1284.

Mokdara, T., Pusawiro, P., & Harnsomburana, J. (2018, July). Personalized food recommendation using deep neural network. In 2018 Seventh ICT international student project conference (ICT-ISPC) (pp. 1-4). IEEE.

Nam, L. N. H. (2021a). Latent factor recommendation models for integrating explicit and implicit preferences in a multi-step decision-making process. Expert Systems with Applications, 174.

Nam, L. N. H. (2021b). Towards comprehensive profile aggregation methods for group recommendation based on the latent factor model. Expert Systems with Applications, 185.

Padmavathi, A., & Sarker, D. (2023, July). RecipeMate: A Food Media Recommendation System Based on Regional Raw Ingredients. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT) (pp. 1-6). IEEE.

Salahian, N., Tab, F. A., Seyedi, S. A., & Chavoshinejad, J. (2023). Deep autoencoder-like NMF with contrastive regularization and feature relationship preservation. Expert Systems with Applications, 214, 119051.

Shen, R. P., Zhang, H. R., Yu, H., & Min, F. (2019). Sentiment based matrix factorization with reliability for recommendation. Expert Systems with Applications, 135, 249-258.

Smith, B., & Linden, G. (2017). Two decades of recommender systems at Amazon. com. Ieee internet computing, 21(3), 12-18.

Teng, C. Y., Lin, Y. R., & Adamic, L. A. (2012, June). Recipe recommendation using ingredient networks. In Proceedings of the 4th annual ACM web science conference (pp. 298-307).

Twomey, N., Fain, M., Ponikar, A., & Sarraf, N. (2020, September). Towards multi-language recipe personalisation and recommendation. In Proceedings of the 14th ACM conference on recommender systems (pp. 708-713).

Vairale, V. S., & Shukla, S. (2021). Recommendation of food items for thyroid patients using content-based knn method. In Data Science and Security: Proceedings of IDSCS 2020 (pp. 71-77). Springer Singapore.

Vy, H. T. H., Pham-Nguyen, C., & Nam, L. N. H. (2024). Integrating textual reviews into neighbor-based recommender systems. Expert Systems with Applications, 249, 123648.

Zhang, J., Li, M., Liu, W., Lauria, S., & Liu, X. (2022). Many-objective optimization meets recommendation systems: A food recommendation scenario. Neurocomputing, 503, 109-117.

Personalization of Dataset Retrieval Results Using a Data Valuation Method

Malick Ebiele[1][a], Malika Bendechache[2][b], Eamonn Clinton[3] and Rob Brennan[1][c]

[1]*ADAPT, School of Computer Science, University College Dublin, Belfield, Dublin, Ireland*

[2]*School of Computer Science, University of Galway, Galway, Ireland*

[3]*Tailte Éireann, Phoenix Park, Dublin, Ireland*

malick.ebiele@adaptcentre.ie, malika.bendechache@universityofgalway.ie, Eamonn.Clinton@tailte.ie, rob.brennan@ucd.ie

Keywords: Data Valuation, Data Value, Personalized Data Value, Dataset Retrieval, Information Retrieval, Quantitative Data Valuation.

Abstract: In this paper, we propose a data valuation method that is used for Dataset Retrieval (DR) results re-ranking. Dataset retrieval is a specialization of Information Retrieval (IR) where instead of retrieving relevant documents, the information retrieval system returns a list of relevant datasets. To the best of our knowledge, data valuation has not yet been applied to dataset retrieval. By leveraging metadata and users' preferences, we estimate the personal value of each dataset to facilitate dataset ranking and filtering. With two real users (stakeholders) and four simulated users (users' preferences generated using a uniform weight distribution), we studied the user satisfaction rate. We define users' satisfaction rate as the probability that users find the datasets they seek in the top $k = \{5, 10\}$ of the retrieval results. Previous studies of fairness in rankings (position bias) have shown that the probability or the exposure rate of a document drops exponentially from the top 1 to the top 10, from 100% to about 20%. Therefore, we calculated the Jaccard_score@5 and Jaccard_score@10 between our approach and other re-ranking options. It was found that there is a 42.24% and a 56.52% chance on average that users will find the dataset they are seeking in the top 5 and top 10, respectively. The lowest chance is 0% for the top 5 and 33.33% for the top 10; while the highest chance is 100% in both cases. The dataset used in our experiments is a real-world dataset and the result of a query sent to a National mapping agency data catalog. In the future, we are planning to extend the experiments performed in this paper to publicly available data catalogs.

1 INTRODUCTION

Given rapidly rising data volumes, knowing which data to keep and which to discard has become an essential task. Data valuation has emerged as a promising approach to tackle this problem (Even and Shankaranarayanan (2005)). The primary focus of data valuation research is the development of methodologies for determining the value of data (Khokhlov and Reznik; Laney; Qiu et al.; Turczyk et al.; Wang et al.; Wang et al. (2020; 2017; 2017; 2007; 2021; 2020)).

Data valuation methods have been applied to data management, machine learning, system security, and energy (Khokhlov and Reznik; Turczyk et al.; Wang et al.; Wang et al. (2020; 2007; 2021; 2020)). There have been no previous attempts to apply data valuation to dataset retrieval. Dataset retrieval is a specialization of information retrieval where instead of retrieving relevant documents the Information Retrieval system returns a list of relevant datasets (Kunze and Auer (2013)). Dataset retrieval systems will return relevant datasets according to a given query. The retrieved datasets are sorted alphabetically by name or using another metadata like creation date or a ranking algorithm incorporated in the dataset retrieval technique. However, they do not consider the user's preferences in terms of metadata. Some dataset retrieval software allows users to sort the results by each metadata separately like creation date, usage, and last update or filtering the results using boolean operations. However, none of them allow users to sort the results by a combination of those metadata (see Equation 5 below). In this paper, we propose a metadata-based

[a] https://orcid.org/0000-0001-5019-6839

[b] https://orcid.org/0000-0003-0069-1860

[c] https://orcid.org/0000-0001-8236-362X

Ebiele, M., Bendechache, M., Clinton, E. and Brennan, R.
Personalization of Dataset Retrieval Results Using a Data Valuation Method.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 122-134
ISBN: 978-989-758-716-0; ISSN: 2184-3228

data valuation method that will allow users to sort dataset retrieval results using a combination of metadata.

Position bias is the study of the relationship between the ranking or position of a retrieved document and the exposure it receives (Agarwal et al.; Craswell et al.; Jaenich et al.; Wang et al. (2019; 2008; 2024; 2018)). In other words, position bias is the study of the probability of a document being consulted by a user according to its position among the retrieved documents. Previous studies have shown that the probability or the exposure rate of a document drops exponentially from the top 1 to the top 10 and then more logarithmically from the top 11 to the top 100 (Jaenich et al. (2024)). Jaenich et al. (2024) also showed that the number of possible orderings of documents for rankings of size $k = 1 \ldots 100$ grows exponentially. Using a group of 6 job seekers as an example, Singh and Joachims (2018) illustrated how a small difference in relevance (used to order retrieved documents or items) can lead to a large difference in exposure (an opportunity) for the group of females. They showed that a 0.03 difference in average relevance (between the top 3 who are all male and the bottom 3 who are all female) can result in a 0.32 difference in average exposure. The difference in average exposure (between the top 3 and the bottom 3) is 10 times the difference in average relevance.

The above studies show that putting the most relevant information on top or providing a fair ranking is crucial. Many fair ranking techniques have been designed to attempt to solve the fairness problem in rankings (Singh and Joachims; Zehlike et al.; Zehlike et al. (2018; 2022a; 2022b)). To the best of our knowledge, none of the existing ranking techniques integrate the user's preferences in the ranking algorithm or use them as a post-retrieval step to re-rank the retrieved information. Here, we present a metadata-based data valuation technique that takes in the retrieved datasets' metadata and the user's preferences and outputs a re-ranking of the retrieved datasets. It is worth noting that because data value is a relative measure if a dataset d_1 is more valuable than d_2 in the whole set of datasets D, then d_1 will always be ranked higher than d_2 considering D or any subset of D containing both datasets.

Many of the existing data valuation approaches are subjective. This is due to the subject-dependent nature of some dimensions (e.g. Utility dimension) that characterise data value (Attard and Brennan (2018)) or the subject-dependent weighting techniques (in the case of weighted averaging or summing) (Deng et al.; Odu (2023; 2019)). Subjective metrics of data value dimensions (metadata are proxy for data value dimensions, therefore usage metadata and usage dimension mean the same thing) or weighting techniques can only be defined by individual users or experts based on their personal views, experiences, and backgrounds. These are opposed to objective metrics that can be determined precisely based on a detailed analysis of the data or extracted from the data infrastructure (Bodendorf et al. (2022)). This makes it challenging to develop a fully objective data valuation model because of the difficulty to objectively measure some dimensions and also experts can be expensive. We believe that instead of generalizing subjective metrics and weighting techniques, it would be better to attempt to develop personalized data valuation models. The difference between subjective data value and personalized data value is that the former assumes that subjective metrics and weights can be applied to every user. Meanwhile, personalized data value will request the subjective metrics and the weights from each user representing their preferences to calculate a personal data value.

Choosing a suitable weighting technique is an additional challenge for weighted approaches to data valuation. For instance, usage-over-time is one of the first data valuation methods and developed a weighting technique based on recency (Chen (2005)). The recency-based weighting technique is objective. The only subjective decision is the choice of assigning higher or lower weights to the more recent Usage metadata. Chen (2005) assigned higher weights to the more recency Usage metadata; which is logical for their use case. In our case, the desired weighting technique should be subjective, performant (have low complexity for calculation), and straightforward for the users to interact with.

The research question is: To what extent can metadata-based data valuation methods improve the results of dataset retrieval systems in terms of users' satisfaction?

To answer this research question, we designed and implemented a metadata-based data valuation method and applied it to a dataset retrieval use case for a National Mapping Agency. The goal is to improve the users' satisfaction by putting on top the datasets they consider more valuable. This is done by taking into account the customers' dataset preferences to re-rank the retrieved datasets.

The contributions of this paper are as follows:

- The first application of a metadata-based data valuation method to dataset retrieval.
- Proposed a personalized and interactive data valuation method. Extant methods are mainly subjective approaches.

The remainder of this paper is structured as follows. Section 2 gives a description of the use case. Section 3 describes the related work. Our proposed metadata-based data valuation method is explained in Section 4. Section 5 explains our experimental design. In Section 6, the experimental results are shown and discussed. Finally, the conclusion and future work are presented in Section 7.

2 USE CASE DESCRIPTION AND BACKGROUND

2.1 Project Description

This data valuation project is part of an ongoing collaboration between researchers from University College Dublin (UCD) and Tailte Éireann (TE). Tailte Éireann (TE) is Ireland's state agency for property registrations, property valuation and national mapping services. It was established on 1 March 2023 from a merger of the Property Registration Authority (PRA), the Valuation Office (VO) and Ordnance Survey Ireland (OSI). The end goal of this collaboration is to design and implement a data valuation method for TE's datasets from the customer's perspective. They would like to apply a metadata-based data valuation to re-rank the results of a query sent to their dataset retrieval platform. The data valuation method should take into account the customers' preferences in terms of metadata. At this stage, the goal is to design and implement a proof of concept.

2.2 Current Dataset Retrieval Process

Figure 1 below displays the current dataset retrieval process (in Blue, some examples here[1][2][3]) and our proposed personalized dataset retrieval process (in Green). In the current process, the user sends a query to the data catalog. The query is then processed and used to extract the relevant datasets from the data catalog. The retrieved datasets are finally formatted in a user-friendly way and sent to the user. In our proposed approach, simultaneously or after the query is sent, the user can specify their preferences in terms of the retrieved datasets' metadata. The user preferences go through a validity test (to test if all of the weights provided are not zeros). The retrieved datasets' metadata and the user preferences are then used to compute the value of each retrieved dataset. The calculated data value is finally used to re-rank the retrieval datasets before formatting them in a user-friendly way and sending them to the user. If no preferences are provided or if they are invalid, then the retrieved datasets are presented alphabetically.

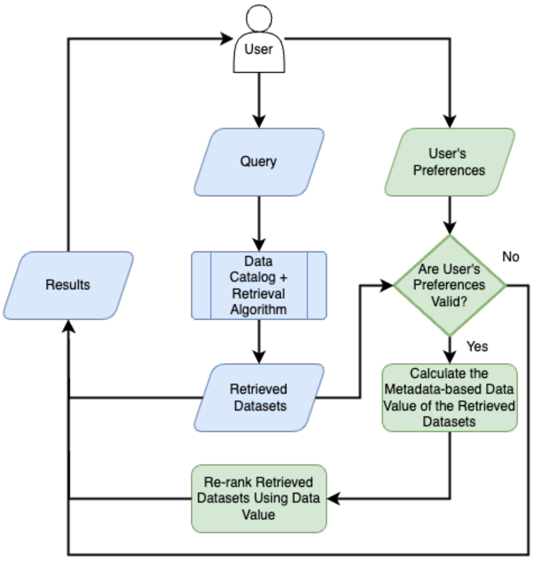

User
Query
User's Preferences
Results
Data Catalog + Retrieval Algorithm
Are User's Preferences Valid?
No
Yes
Retrieved Datasets
Calculate the Metadata-based Data Value of the Retrieved Datasets
Re-rank Retrieved Datasets Using Data Value

Figure 1: Personalized datasets retrieval using a metadata-based data valuation. *In Blue is the current dataset retrieval process. In Green are the additional steps we proposed to personalize dataset retrieval results.*

2.3 Information and Dataset Retrieval Performance Metrics

The Jaccard index also known as the Jaccard score has been chosen to evaluate the users' satisfaction. The Jaccard score measures the similarity between at least two finite sets and is defined as the size their intersection divided by the size of their union (see Equation 1 below). The truncated Jaccard score at k (Jaccard_score@k), which only focuses on the top k elements, is preferred for our use case. As shown in Section 1, only the top k (with $k \leq 10$) are most likely to be consulted. Therefore, focusing mainly on the top k elements makes sense.

However, Jaccard score does not take into account the positions. So, the Normalized Discounted Cumulative Gain (NDCG) has also been calculated. NDCG is widely used and involves a discount function over the rank while many other measures uniformly weight all positions (see Equation 2 below). It measures the matching degree between our ranking and other rankings.

NDCG and Jaccard_score@k range between 0 and 1, with 1 being the optimal performance. We used the scikit-learn implementation of NDCG with the de-

[1] https://data.gov

[2] https://www.kaggle.com/datasets

[3] https://datasetsearch.research.google.com

fault parameters[4] and a self implementation of Jaccard_score@k in Python.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (1)$$

$$\begin{aligned} \mathrm{NDCG}_D(f,S_n) &= \frac{\mathrm{DCG}_D(f,S_n)}{\mathrm{IDCG}_D(S_n)}, \\ \mathrm{DCG}_D(f,S_n) &= \sum_{r=1}^{n} \mathrm{G}(y^{f}_{(r)})D(r), \qquad (2) \\ \mathrm{IDCG}_D(S_n) &= \max_{f'} \sum_{r=1}^{n} \mathrm{G}(y^{f'}_{(r)})D(r), \end{aligned}$$

with $D(r) = \frac{1}{\log_b(1+r)}$ (inverse logarithm decay with base b) the discount function, S_n is a dataset, f is a ranking function, f' is the best ranking function on S_n, and G is the Gain. $\mathrm{DCG}_D(f,S_n)$ is the Discounted Cumulative Gain (DCG) of f on S_n with discount D and $\mathrm{IDCG}_D(S_n)$ is the Ideal DCG.

3 RELATED WORK

This section describes the current state-of-the-art information and dataset retrieval approaches and their limitations. Then, it highlights the challenges related to weighted average approaches because the approach proposed in this paper falls into that category.

3.1 Information and Dataset Retrieval

Tamine and Goeuriot (2021) define Information retrieval (IR) as a system that deals with the representation, storage, organization and access to information items. It has two main processes: Indexing (which consists of building computable representations of content items using metadata) and Retrieval (which consists of optimally matching queries to relevant documents) (Tamine and Goeuriot (2021)). IR models have evolved since the 1960s from Boolean to Neural Networks (Lavrenko and Croft; Liu; Maron and Kuhns; Miutra and Craswell; Robertson et al.; Salton et al.; Salton and McGill; Tamine and Goeuriot (2001; 2009; 1960; 2018; 1980; 1983; 1986; 2021)).

Hambarde and Proença (2023) argue that IR systems have two stages: retrieval and ranking. The retrieval stage consists of four main techniques: Conventional IR, Sparse IR, Dense IR, and Hybrid IR techniques. The latter is any combination of the former three. The ranking stage consists of two main approaches: Learning To Rank and Deep Learning Based Ranking approaches. For more details on this categorization of IR techniques, please refer to Hambarde and Proença (2023).

Liu et al. (2020) argue that the IR research community has long agreed that major improvement of search performance can only be achieved by taking account of the users and their contexts, rather than through developing new retrieval algorithms that have reached a plateau. Three main approaches have been employed to personalize IR results: Query expansion, Result re-ranking, and Hybrid personalization techniques (Liu et al. (2020)). Query expansion collects additional information about user interest from heterogeneous sources, represents them by some terms, and automatically adds these terms to the initial query for a refined search (Bai et al.; Belkin et al.; Biancalana et al.; Bilenko et al.; Bouadjenek et al.; Budzik and Hammond; Buscher et al.; Cai and de Rijke; Chen and Ford; Chirita et al.; Jayarathna et al.; Kelly et al.; Kraft et al. (2007; 2005; 2008; 2008; 2013; 1999; 2009; 2016; 1998; 2007; 2013; 2005; 2005)). Result re-ranking techniques reorder search results for users according to document relevance (Gauch et al.; Liu et al.; Liu and Hoeber; Tanudjaja and Mui; Wang et al. (2003; 2002; 2011; 2002; 2013)). Hybrid techniques combine query expansion and result re-ranking; they outperform either one individually but are under-explored (Ferragina and Gulli; Lv et al.; Pitkow et al.; Pretschner and Gauch; Shen et al. (2005; 2006; 2002; 1999; 2005)).

Most re-ranking systems are not interactive. They have some sort of pre-settled weighting criteria for re-ranking, giving heavier weight to those documents that match user interests and push them to top ranks (Liu et al.; Tanudjaja and Mui (2002; 2002)). The ones that are interactive present the top k documents to the users for feedback and then refine ranking based on the feedback (Gauch et al.; Liu and Hoeber; Wang et al. (2003; 2011; 2013)).

Thus it can be seen that interactive IR result re-ranking based on users' preferences is under-explored. The approach proposed in this paper is an interactive dataset retrieval technique based on users' preferences in terms of the retrieved datasets' metadata.

3.2 Weighted Average Data Valuation Methods

There were also previous attempts to calculate the data value using weighted averaging of metadata describing data value dimensions (Chen; Ma and Zhang; Qiu et al. (2005; 2019; 2017)). For instance, measur-

[4] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ndcg_score.html

ing usage-over-time is one of the first data valuation methods and it estimates data value with the weighted averaging approach of Chen (Chen (2005)). It consists of splitting the usage data into a series of time slots, assigning a weight to each time slot, and then computing the data value using the weighted average. The weights are the normalized recency weights. The more recent time slots are assigned the higher weights (Chen (2005)). Ma and Zhang (2019) extended the usage-over-time model by adding the age and size dimensions. Their Multi-Factors Data Valuation Method (*MDV*) is a trade-off between dynamic and static data value. The dynamic data value is the usage-over-time model of Chen. The static data value is the weighted average of the normalized age and size. The weights of the age and size dimensions are assigned subjectively by experts.

Qiu et al. (2017) used the Analytic Hierarchy Process (AHP) which is a different weighting approach. AHP requires a subjective rating of the input dimensions in pairs. These pairwise comparisons are then arranged in a matrix (the Judgement matrix, see Appendix 7), from which a final weighting of the dimensions will be calculated. AHP is technically straightforward to implement and more importantly allows to assess the transitivity consistency of the pairwise comparisons matrix by assigning a consistency score to it. However, experts are still needed for the pairwise rating of the input dimensions. Qiu et al. (2017) use the measure of 6 dimensions in their model. Those dimensions are: the size of the data (S), the access interval (T), the data read and write frequency (F), the number of visits (C), the contents of the file (D), and the potential value of the data (V). For more details on the dimensions used, please refer to Qiu et al. (2017).

The challenge of applying weighted approaches is the weighting technique. In our case, the desired weighting technique must be straightforward for the users to interact with and fast to compute as it is supposed to be integrated into a live system for interactive IR re-ranking. The weighting approach used in this paper is detailed in Section 4.1.

To the best of our knowledge, the application of a metadata-based data valuation approach to dataset retrieval proposed in this study is unique. Also, none of the studies described above validated the outputs of their data valuation approaches. Our approach is validated using preferences from two stakeholders and four simulated users.

4 PROPOSED DATA VALUATION METHOD

Our method has two main steps: first dimension metadata weight determination and then data value calculation. These are described below.

4.1 Weight Determination

Analytic Hierarchy Process (AHP) was our first choice because of its sound mathematical basis (Saaty (1987)). However, it was challenging to apply, as instead of assigning a weight to each metadata or dimension, a pairwise comparison of the dimensions is needed (Saaty (1987)). E.g. usage is twice as important as creation date, usage is 5 times more important than the number of spatial objects, or usage is twice less important than currency. This exercise was difficult for the stakeholders who participated in the experiments. They confessed being more comfortable with a rating-like weighting approach e.g. 1 to 5 websites or products rating mechanism. Also, AHP assumes that preferences are transitive and has a transitivity consistency test. Saaty (1987) advise to discard the current weights deduced from the pairwise comparisons if the consistency ratio is greater than 0.1. Previous studies showed that preferences are not always transitive (Alós-Ferrer et al.; Alós-Ferrer and Garagnani; Fishburn; Gendin (2023; 2021; 1991; 1996)). Alós-Ferrer et al. (2023) shows using two preference datasets that no matter the initial assumptions, even when the preferences are supposed to be transitive, a maximum of 27.45% of individual preferences are non-transitive. We believe that assuming that all preferences are transitive implies ignoring some individual preferences. Therefore, we used a slider from 0 to 10 (with a step of 1) as the weights determination technique; the presence of a zero rating allows the individual to discard a particular metadata as not relevant to the use case or at that time. This approach is straightforward and inclusive because it was tested during the interviews with the stakeholders. The only constraint in our weighting approach is that at least one of the provided weights should be non-zero.

4.2 Data Value Calculation

This is split into the following steps: Data preprocessing and Data value calculation.

4.2.1 Data Preprocessing

As the collected metadata values have different scales, they must be normalized. The weights also must be normalized. For the Number of spatial objects metadata (see Table 1 below for the description), the values are divided by the maximum value. For the Usage, because it is a time series data (collected monthly from January 2017 to January 2023). It is normalized by dividing each value by the maximum value of each month. Then the current Usage value is the 6-month Exponential Moving Average (EMA). EMA is widely used in finance to capture stock and bond price trends while reducing noises like sudden sharp moves. It was first introduced by Roberts (1959) (see Equation 3). The 6-month Exponential Moving Average was calculated using the Pandas implementation with default parameters[5]. As to the creation date, we applied the probabilistic approach of calculating data currency with a decline rate of 20%. This approach was proposed by Heinrich and Klier (2011) and the data currency $Q_{Curr.}(\omega, A)$ formula is shown in the Equation 4 below. ω is a value in the Attribute A. The motivation is that the currency of information does not solely depend on its age but also on whether the information is likely to change over time or not. For instance, a satellite image of a mountain range might still be relevant even if the image is 30 years old. On the other hand, a 10-year-old satellite image of road networks might be outdated.

$$\mathrm{EMA}_t(U,n) = \frac{\sum_{i=0}^{n}(1-\alpha)^i U_{t-i}}{\sum_{i=0}^{n}(1-\alpha)^i}, \tag{3}$$

t is the current time, n the number of past periods, U the time-series of usage metadata, U_t the usage metadata at time t, α ($0 < \alpha \leq 1$) is the smoothing factor, and $\mathrm{EMA}_t(U,n)$ the EMA of usage metadata at time t considering n previous periods.

$$Q_{Curr.}(\omega, A) := exp(-decline(A) \cdot age(\omega, A)) \tag{4}$$

For the weights, the weight of each metadata has been divided by the sum of the weights of all three metadata per stakeholder.

4.2.2 Actual Data Value Calculation

The data value is then the weighted average of the metadata values using the Equation 5 below.

$$V(d_i) = w_{\mathrm{U}} \times \mathrm{U}_i + w_{\mathrm{Q}} \times \mathrm{Q}_i + w_{\mathrm{O}} \times \mathrm{O}_i, \tag{5}$$

where $w_{\{\mathrm{U,\,Q,\,O}\}}$ in [0,1] are the weights and $V(d_i)$ in [0,1] the data value. U, Q, and O stand for Usage, Currency (derived from the Creation date; see Equation 4), and Number of Spatial Objects, respectively.

Table 1: Description of the metadata used in this paper.

Metadata / Data Value Dimension	Description
Usage	Access counts. It measures how many times a given dataset has been accessed.
Creation date	Date the first version has been made available for the users or the last date it has been updated.
Number of spatial objects	The number of geometric data (e.g. points, lines, polygons, paths) in the dataset. It is a domain-relevant measure of data volume and information content.

5 EXPERIMENTAL DESIGN

Figure 2 below shows the flowchart of our experimental design. The experiments consist of re-ranking dataset retrieval results using a metadata-based data valuation technique. It has four main steps: Metadata extraction, User preferences request, Data value calculation, and Re-ranking of the retrieved datasets.

5.1 Metadata Extraction

This consists of extracting metadata from the data catalog system. For this use case, only three metadata types have been extracted from 15 datasets: creation date, number of spatial objects, and usage. The 15 selected datasets are the results of a query sent to the data catalog system; they are ordered alphabetically by default.

5.2 User Preferences Request

For this use case, the user preferences have been requested during interviews with three stakeholders. The stakeholders included in this study are managers within the mapping agency with data management responsibilities for at least 3 years each.

The main goal of each interview (15-20 minutes) was to get the stakeholders to assign weights to each metadata field. A slider from 0 to 10 (with a step of 1) is used to assign the weight to each metadata.

Table 3 shows the weights provided by each stakeholder. Stakeholder 2 (SH2) provided an invalid set of weights (all of the weights are zero) because all of the metadata selected for this case study was irrelevant to them. Therefore, the retrieved datasets will be alphabetically presented to Stakeholder 2.

5.3 Personal Data Value Calculation

The personal data value is calculated for each dataset using the valid weights provided by stakeholders SH1

[5] https : / / pandas.pydata.org / docs / reference / api / pandas.DataFrame.ewm.html

and SH3 and four randomly generated users' preferences (using a uniform weight distribution) and Equation 5. The datasets are then ranked by data value. The resulting personalized rankings are then compared to the default alphabetic order, MDV and AHP-based re-rankings. They were also compared to the univariate rankings based on each metadata independently (Usage, Number of Spatial Objects, and Currency; the current IR/DR data catalog re-ranking options).

6 EXPERIMENTAL RESULTS

6.1 Comparison with Other Data Valuation Approaches

In this Section, we compare our approach with other data valuation approaches: Chen (2005)'s usage-over-time, Ma and Zhang (2019)'s MDV, and Qiu et al. (2017)'s AHP-based data valuation techniques.

6.1.1 Our Approach vs Usage-over-Time Model

To computer the usage-over-time data value (see Equation 6), we used a valuation period (vp) of 6 months, a lifestage length s of 1 month (usually in terms of usage metadata granularity, here on monthly basis), $N_t = 6$ (N_t is the number of lifestages per valuation period), and $x = 2$ (x is a regularizer of the slope of the weight distribution together with N_t). Chen (2005) suggest that significantly flat (too large x or N_t) or steep (too small x or N_t) weight distributions should be avoided. Chen (2005) also advised that a valid valuation period for long-lived information should be at least a few months on a quarterly or semi-annual basis.

We chose $x = 2$ because, for the examples shown by Chen (2005) with $N_t = 5$, the weight distribution is too flat for $x = 1.2$, too steep for $x = 3$, and in between for $x = 2$.

We have 13 valuation periods with a length of 6 months for the first 12 periods and 1 month for the last period. Therefore, for the last valuation period, UT is equal to the collected usage data.

$$V_t(d) = \sum_{i=1}^{N_t} (w(i) \times f(U_i(d))),\ 0 \leq f(U_i(d)) \leq 1,$$
$$w(i) = \frac{(\frac{1}{x})^i}{\sum_{j=1}^{N_t} (\frac{1}{x})^j},\ \sum_{i=1}^{N_t} w(i) = 1,\ x \geq 1,$$
$$vp = [t - (N_t \times s), t], N_t = \frac{vp}{s}. \tag{6}$$

Figure 3 below shows the Usage metadata trends of the retrieved datasets (Figure 3a), the usage-over-time (Figure 3b), and 6-month Exponential Moving Average (EMA-6, Figure 3c). We can see that both usage-over-time (as per Chen) and our proposed 6-month EMA capture the main usage trends with reduced noise (steep highs and lows). The main difference is that the 6-month EMA reduces the effects of the noise on the present values while the usage-over-time removes them completely. EMA is preferred because it captures every movement while usage-over-time fails for the same valuation period.

To make the graphs below and in the remainder of this paper easy to read, Table 2 has been generated. It maps each dataset to a unique ID. The dataset names have been sorted alphabetically and an ID starting from 1 has been assigned to them.

Table 2: Dataset IDs and Names Mapping.

IDs	Datasets
1	ig/basemap_premium
2	itm/6inch_cassini
3	itm/basemap_premium
4	itm/basemap_public
5	itm/digitalglobe
6	itm/historic_25inch
7	itm/historic_6inch_cl
8	itm/national_high_resolution_imagery
9	itm/ortho
10	itm/ortho_2005
11	wm/basemap_eire
12	wm/basemap_ms_public
13	wm/basemap_premium
14	wm/basemap_public
15	wm/digitalglobe

6.1.2 Our Approach vs MDV and AHP Data Valuation Approaches

MDV (Ma and Zhang (2019)) is a natural extension of the usage-over-time model by adding the Age (Valuation Date minus Creation Date) and the Size metadata to the Usage metadata. MDV is calculated using the Equation 7 below. The weights of the age (W_{age}) and the size (W_{size}), and the trade-off coefficient k are set to $W_{age} = W_{size} = 0.5$ and $k = 0.2$; the same values as the example presented by (Ma and Zhang (2019)). The Age and Size metadata are normalized using the MinMax scaler (Scikit-learn implementation with default parameters[6]) then the resulting value is subtracted from 1 because Ma and Zhang (2019) assume that more recent and smaller sized datasets are

[6] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

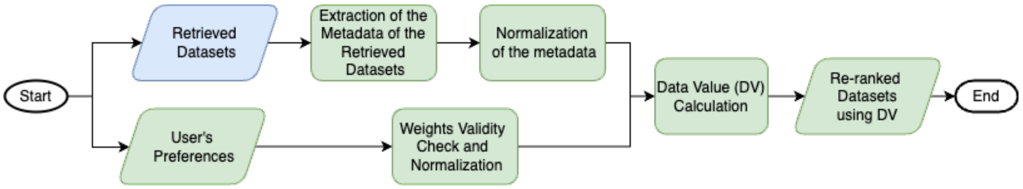
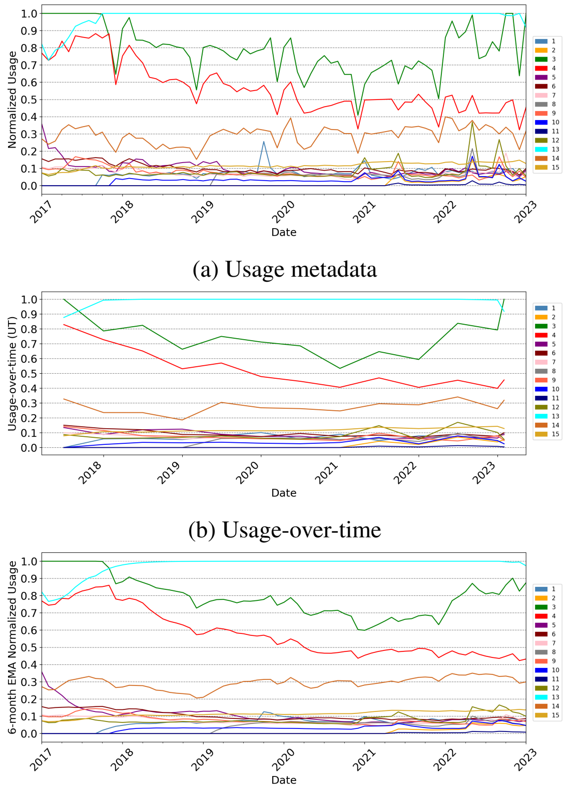

Start
Retrieved Datasets
Extraction of the Metadata of the Retrieved Datasets
Normalization of the metadata
User's Preferences
Weights Validity Check and Normalization
Data Value (DV) Calculation
Re-ranked Datasets using DV
End

Figure 2: Experimental design for personalized metadata-based data valuation.



(a) Usage metadata

(b) Usage-over-time

(c) 6-month Exponential Moving Average (EMA)

Figure 3: Comparison of usage-over-time with a 6-month EMA at capturing the usage metadata trends.

considered more valuable.

$$
\begin{aligned}
V &= kV_s + (1-k)V_d,\\
V_s &= w_{size} \times f(S(d)) + w_{age} \times f(A(d)),\\
&\quad 0 \le f(S(d)) \le 1, 0 \le f(A(d)) \le 1,\\
V_d &= V_t(d) \text{ (see Equation 6).}
\end{aligned}
\tag{7}
$$

As we couldn't collect pairwise comparisons of the metadata from the stakeholders (see Section 4.1), we will use the weights they provided (see Table 3) to produce proxy pairwise comparisons. The provided weights are summed per metadata type and then the inverse of the sum per metadata is multiplied by the maximum of the sum (see Table 4 and Equation 8). The obtained pairwise comparison vector is used to fill out the AHP Judgement matrix using its reciprocity and transitivity properties (see Appendix 7).

$$
\begin{aligned}
V_{\text{AHP}} &= \left[\frac{w''_{\text{Q}}}{w''_{\text{Q}}}, \frac{w''_{\text{Q}}}{w''_{\text{U}}}, \frac{w''_{\text{Q}}}{w''_{\text{O}}}\right] = \left[1, \frac{w''_{\text{Q}}}{w''_{\text{U}}}, \frac{w''_{\text{Q}}}{w''_{\text{O}}}\right],\\
&\text{Because } w''_{\text{Q}} = \text{Max}(w''_{\text{Q}}, w''_{\text{U}}, w''_{\text{O}}).\\
\text{With } w''_{\text{U}} &= \sum_{i=1}^{m} w'_{\text{U}_i} \neq 0, w''_{\text{Q}} = \sum_{i=1}^{m} w'_{\text{Q}_i} \neq 0,\\
w''_{\text{O}} &= \sum_{i=1}^{m} w'_{\text{O}_i} \neq 0,
\end{aligned}
\tag{8}
$$

V_{AHP} is the first row vector of the judgement matrix P because the diagonal elements of P are equal to 1. From V_{AHP}, we can deduce the first column vector of P using its reciprocity property. Then, fill out the rest of the matrix P using its transitivity property[7]. For more details see Appendix 7.

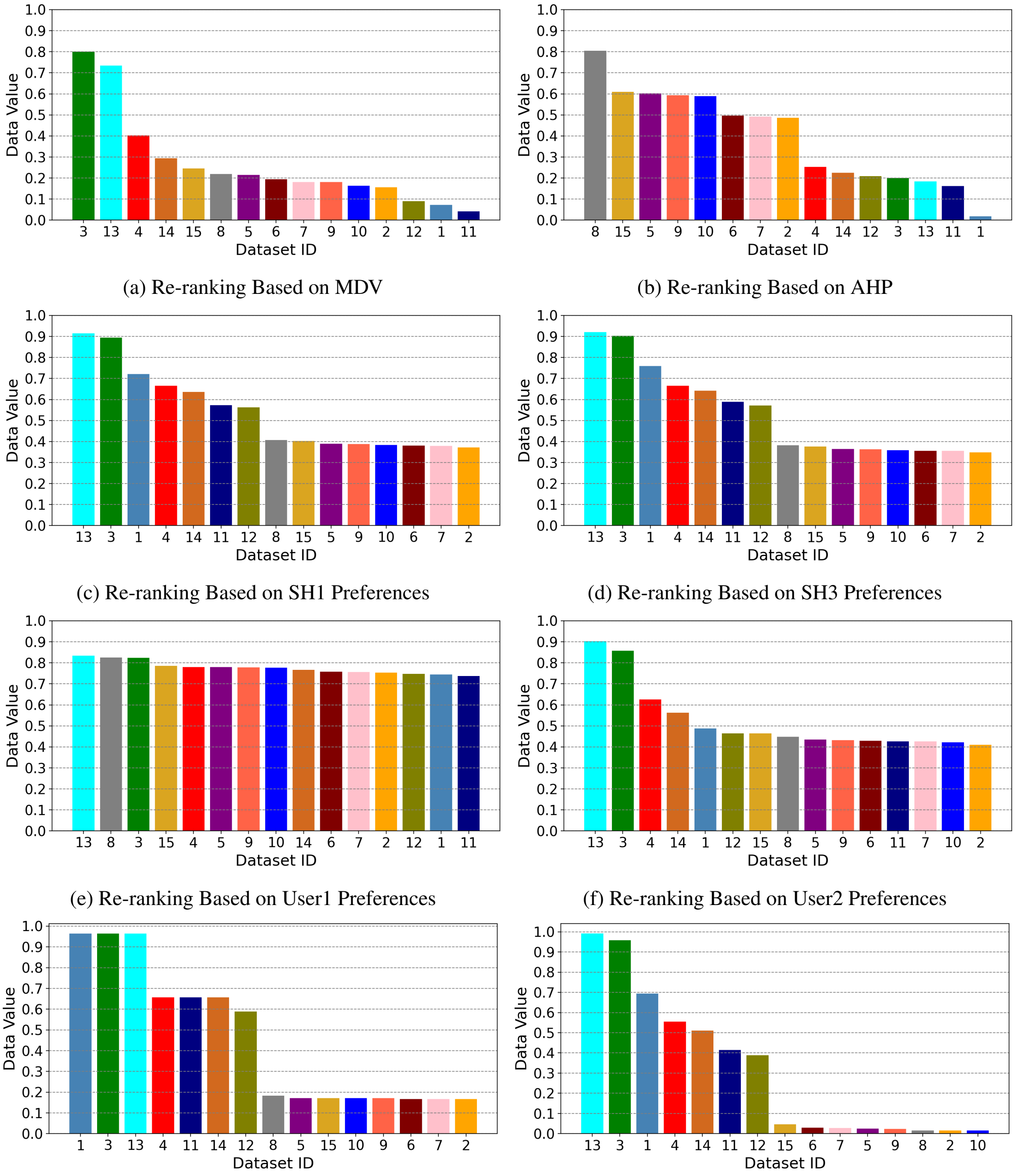
Figure 4 below shows the order in which the retrieved datasets are presented to the users based on MDV, AHP, and ours (ties are broken using alphabetic order). Figures 4a and 4b display the order in which the retrieved datasets are shown to all the users. Figures 4c-4h show the order in which the results are presented to each user according to their preferences. One can see that the order is different from one user to another and from each user to MDV and AHP-based rankings.

It can also be seen that the data value varies according to the weights assigned to each metadata. Therefore, we are going to measure the users' satisfaction rate in Section 6.2 below.

6.2 Users' Satisfaction Evaluation

We define a user satisfaction rate as the probability that users find the datasets they seek in the top $k = \{5, 10\}$ of the retrieval results. Therefore, we calculated the Jaccard_score@5 and Jaccard_score@10 between our approach and other re-ranking options. We also computed NDCG which measures the degree

[7] It works fine considering V_{AHP} as the first column vector of P instead of its first row vector. One just needs to apply the reciprocity property of P then its transitivity property.



(a) Re-ranking Based on MDV

(b) Re-ranking Based on AHP

(c) Re-ranking Based on SH1 Preferences

(d) Re-ranking Based on SH3 Preferences

(e) Re-ranking Based on User1 Preferences

(f) Re-ranking Based on User2 Preferences

(g) Re-ranking Based on User3 Preferences

(h) Re-ranking Based on User4 Preferences

Figure 4: Retrieved Datasets' Re-ranking Based on MDV, AHP, and Ours.

to which the results re-ranking using users' preferences match the other re-rankings.

Table 5 presents the evaluation results regarding NDCG, Jaccard_score@5, and Jaccard_score@10 per user. The highest and the lowest scores per user and metric are highlighted in **bold** and red. There is a 42.24% and a 56.52% chance on average that users will find the dataset they are seeking in the top 5 and top 10, respectively. The lowest chance is 0% for the top 5 and 33.33% for the top 10; while the highest chance is 100% in both cases. On average, the different re-rankings match the users' preferred ordering 81.81% of the time.

It can also be seen in Table 5 that the degree to

Table 3: Dataset value dimension (metadata field) weights provided by stakeholders. SH2 provided an invalid set of weights.

Stakeholders (SH) / Users	Currency	#Spatial Objects	Usage
SH1	10	8	5
SH3	9	9	4
User1	9	0	1
User2	7	1	7
User3	2	8	0
User4	0	4	2

Table 4: From stakeholders' provided weights to AHP weights.

Steps	Currency / Age	#Spatial Objects (Proxy for Size)	Usage
SH1	10	8	5
SH3	9	9	4
The Sum of the provided weights	19	17	9
A pairwise comparison	1	19/17	19/9
AHP weights	0.4222	0.3778	0.2

Table 5: Evaluation Results.

Users	Data Value Dims/Methods	NDCG	Jaccard_score@5	Jaccard_score@10
SH1	#Objects	0.8035	**0.6667**	**1.0000**
	6month_EMA	**0.8958**	**0.6667**	0.6667
	AHP (Qiu et al.)	0.7487	0.0000	0.3333
	Alphabetic order	0.7506	0.4286	0.3333
	Currency	0.8482	0.0000	0.3333
	MDV (Ma and Zhang)	0.8445	**0.6667**	0.5385
	UT (Chen)	0.8384	**0.6667**	0.6667
SH3	#Objects	0.8035	**0.6667**	**1.0000**
	6month_EMA	**0.8958**	**0.6667**	0.6667
	AHP (Qiu et al.)	0.7487	0.0000	0.3333
	Alphabetic order	0.7506	0.4286	0.3333
	Currency	0.8482	0.0000	0.3333
	MDV (Ma and Zhang)	0.8445	**0.6667**	0.5385
	UT (Chen)	0.8384	**0.6667**	0.6667
User1	#Objects	0.7669	0.2500	0.5385
	6month_EMA	0.8418	**0.4286**	0.5385
	AHP (Qiu et al.)	0.8170	**0.4286**	0.6667
	Alphabetic order	0.8199	0.2500	0.5385
	Currency	0.7846	**0.4286**	0.5385
	MDV (Ma and Zhang)	**0.8540**	**0.4286**	**0.8182**
	UT (Chen)	0.8320	**0.4286**	0.5385
User2	#Objects	**0.8660**	**0.6667**	**0.8182**
	6month_EMA	0.8051	**0.6667**	0.6667
	AHP (Qiu et al.)	0.7857	0.0000	0.4286
	Alphabetic order	0.7692	0.4286	0.4286
	Currency	0.7215	0.0000	0.4286
	MDV (Ma and Zhang)	0.7524	**0.6667**	0.6667
	UT (Chen)	0.7493	**0.6667**	0.6667
User3	#Objects	**0.9977**	**1.0000**	**1.0000**
	6month_EMA	0.8225	0.4286	0.6667
	AHP (Qiu et al.)	0.8040	0.0000	0.3333
	Alphabetic order	0.7571	0.4286	0.3333
	Currency	0.8128	0.0000	0.3333
	MDV (Ma and Zhang)	0.8174	0.4286	0.5385
	UT (Chen)	0.8239	0.4286	0.6667
User4	#Objects	0.8245	**0.6667**	0.6667
	6month_EMA	**0.9062**	**0.6667**	**0.8182**
	AHP (Qiu et al.)	0.7434	0.0000	0.3333
	Alphabetic order	0.8174	0.4286	0.3333
	Currency	0.8891	0.0000	0.3333
	MDV (Ma and Zhang)	0.8623	**0.6667**	0.5385
	UT (Chen)	0.8556	**0.6667**	**0.8182**

which a given re-ranking technique matches a user's preferred ordering does not predict the probability of the user finding what they are seeking. For instance, for SH1, 6month_EMA got the highest NDCG score. However, 6month_EMA got the same Jaccard_score@5 as #Objects, MDV, and UT and a lower Jaccard_score@10 than #Objects.

7 CONCLUSION

This paper introduces a data valuation method that can be used to re-rank dataset retrieval results. It showed, using 12 datasets (the result of a query sent to a data catalog) and 6 users (including two stakeholders and 4 randomly generated using the uniform distribution of the weights), that there is only a 42.24% and a 56.52% chance on average that users will find the dataset they are seeking in the top 5 and top 10, respectively. Users should find the information they are seeking in the top 10 because, as shown by Jaenich et al. (2024), the probability of a document being consulted drops exponentially from the top 1 (100%) to the top 10 (about 20%). In other words, if a document is not in the top 10, its chances of being consulted are less than 20%. It is important to re-rank retrieval results according to users' interests because, in addition to the query sent to a data catalog, users also have preferences regarding the retrieved datasets' properties or metadata. In fact, Liu et al. (2020) argue that the IR scholars have agreed that major improvement in search performance can only be achieved by considering the users and their contexts; thus their preferences. This paper is a step in that direction by using the users' preferences to re-rank IR results.

In the future, we are planning to run a set of queries on public data catalogs (e.g. Kaggle datasets[8]) and collect the top k (k≤100) results sorted by relevance and study the distribution of users' satisfaction through simulation.

ACKNOWLEDGEMENTS

This research has received funding from the ADAPT Centre for Digital Content Technology, funded under the SFI Research Centres Programme (Grant 13/RC/2106_P2), co-funded by the European Regional Development Fund. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

Agarwal, A., Zaitsev, I., Wang, X., Li, C., Najork, M., and Joachims, T. (2019). Estimating Position Bias without Intrusive Interventions. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pages 474–482, New York, NY, USA. Association for Computing Machinery.

Alós-Ferrer, C., Fehr, E., and Garagnani, M. (2023). Identifying nontransitive preferences. Publisher: [object Object].

Alós-Ferrer, C. and Garagnani, M. (2021). Choice consistency and strength of preference. *Economics Letters*, 198:109672.

Attard, J. and Brennan, R. (2018). Challenges in Value-Driven Data Governance. In Panetto, H., Debruyne, C., Proper, H. A., Ardagna, C. A., Roman, D., and Meersman, R., editors, *On the Move to Meaningful Internet Systems. OTM 2018 Conferences*, volume 11230, pages 546–554. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Bai, J., Nie, J.-Y., Cao, G., and Bouchard, H. (2007). Using query contexts in information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 15–22, New York, NY, USA. Association for Computing Machinery.

Belkin, N. J., Cole, M., Gwizdka, J., Li, Y. L., Liu, J. J., Muresan, G., Roussinov, D., Smith, C. A., Taylor, A., and Yuan, X. J. (2005). Rutgers information interaction lab at TREC 2005: Trying HARD. In *NIST Special Publication*. Institute of Electrical and Electronics Engineers Inc. ISSN: 1048-776X.

Biancalana, C., Micarelli, A., and Squarcella, C. (2008). Nereau: a social approach to query expansion. In *Proceedings of the 10th ACM workshop on Web information and data management*, WIDM '08, pages 95–102, New York, NY, USA. Association for Computing Machinery.

Bilenko, M., White, R. W., Richardson, M., and Murray, G. C. (2008). Talking the talk vs. walking the walk: salience of information needs in querying vs. browsing. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 705–706, New York, NY, USA. Association for Computing Machinery.

Bodendorf, F., Dehmel, K., and Franke, J. (2022). *Scientific Approaches and Methodology to Determine the Value of Data as an Asset and Use Case in the Automotive Industry*.

Bouadjenek, M. R., Hacid, H., and Bouzeghoub, M. (2013). LAICOS: an open source platform for personalized social web search. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '13, pages 1446–1449, New York, NY, USA. Association for Computing Machinery.

Budzik, J. and Hammond, K. (1999). Watson: Anticipating and Contextualizing Information Needs.

Buscher, G., van Elst, L., and Dengel, A. (2009). Segment-level display time as implicit feedback: a comparison to eye tracking. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 67–74, New York, NY, USA. Association for Computing Machinery.

Cai, F. and de Rijke, M. (2016). Selectively Personalizing Query Auto-Completion. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, SIGIR '16, pages 993–996, New York, NY, USA. Association for Computing Machinery.

Chen, S. Y. and Ford, N. (1998). Modelling user navigation behaviours in a hyper-media-based learning system : An individual differences approach. *Knowledge Organization*.

Chen, Y. (2005). Information valuation for information lifecycle management. In *Second International Conference on Autonomic Computing (ICAC'05)*, pages 135–146. IEEE.

Chirita, P. A., Firan, C. S., and Nejdl, W. (2007). Personalized query expansion for the web. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 7–14, New York, NY, USA. Association for Computing Machinery.

Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 87–94, New York, NY, USA. Association for Computing Machinery.

Deng, Y., Li, X., Yu, X., Fu, Z., Chen, H., Xie, J., and Xie, D. (2023). Electricity Data Valuation Considering Attribute Weights. In *2023 3rd International Conference on Intelligent Power and Systems (ICIPS)*, pages 788–794.

Even, A. and Shankaranarayanan, G. (2005). Value-Driven Data Quality Assessment. In *ICIQ*.

[8] https://www.kaggle.com/datasets

Ferragina, P. and Gulli, A. (2005). A personalized search engine based on web-snippet hierarchical clustering. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, WWW '05, pages 801–810, New York, NY, USA. Association for Computing Machinery.

Fishburn, P. C. (1991). Nontransitive Preferences in Decision Theory. *Journal of Risk and Uncertainty*, 4(2):113–134. Publisher: Springer.

Gauch, S., Chaffee, J., and Pretschner, A. (2003). Ontology-based personalized search and browsing. *Web Intelli. and Agent Sys.*, 1(3-4):219–234.

Gendin, S. (1996). Why Preference is Not Transitive. *The Philosophical Quarterly (1950-)*, 46(185):482–488. Publisher: [Oxford University Press, University of St. Andrews, Scots Philosophical Association].

Hambarde, K. A. and Proença, H. (2023). Information Retrieval: Recent Advances and Beyond. *IEEE Access*, 11:76581–76604. Conference Name: IEEE Access.

Heinrich, B. and Klier, M. (2011). Assessing data currency—a probabilistic approach. *Journal of Information Science*, 37(1):86–100. Publisher: Sage Publications Sage UK: London, England.

Jaenich, T., McDonald, G., and Ounis, I. (2024). Query Exposure Prediction for Groups of Documents in Rankings. In Goharian, N., Tonellotto, N., He, Y., Lipani, A., McDonald, G., Macdonald, C., and Ounis, I., editors, *Advances in Information Retrieval*, pages 143–158, Cham. Springer Nature Switzerland.

Jayarathna, S., Patra, A., and Shipman, F. (2013). Mining user interest from search tasks and annotations. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, CIKM '13, pages 1849–1852, New York, NY, USA. Association for Computing Machinery.

Kelly, D., Dollu, V. D., and Fu, X. (2005). The loquacious user: a document-independent source of terms for query expansion. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 457–464, New York, NY, USA. Association for Computing Machinery.

Khokhlov, I. and Reznik, L. (2020). What is the value of data value in practical security applications. In *2020 IEEE Systems Security Symposium (SSS)*, pages 1–8. IEEE.

Kraft, R., Maghoul, F., and Chang, C. C. (2005). Y!Q: contextual search at the point of inspiration. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 816–823, New York, NY, USA. Association for Computing Machinery.

Kunze, S. R. and Auer, S. (2013). Dataset Retrieval. In *2013 IEEE Seventh International Conference on Semantic Computing*, pages 1–8.

Laney, D. B. (2017). *Infonomics: how to monetize, manage, and measure information as an asset for competitive advantage*. Routledge.

Lavrenko, V. and Croft, W. B. (2001). Relevance based language models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 120–127, New York, NY, USA. Association for Computing Machinery.

Liu, F., Yu, C., and Meng, W. (2002). Personalized web search by mapping user queries to categories. In *Proceedings of the eleventh international conference on Information and knowledge management*, CIKM '02, pages 558–565, New York, NY, USA. Association for Computing Machinery.

Liu, H. and Hoeber, O. (2011). A Luhn-Inspired Vector Reweighting Approach for Improving Personalized Web Search. pages 301–305. IEEE Computer Society.

Liu, J., Liu, C., and Belkin, N. J. (2020). Personalization in text information retrieval: A survey. *Journal of the Association for Information Science and Technology*, 71(3):349–369. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.24234.

Liu, T.-Y. (2009). Learning to Rank for Information Retrieval. *Found. Trends Inf. Retr.*, 3(3):225–331.

Lv, Y., Sun, L., Zhang, J., Nie, J.-Y., Chen, W., and Zhang, W. (2006). An iterative implicit feedback approach to personalized search. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 585–592, USA. Association for Computational Linguistics.

Ma, X. and Zhang, X. (2019). MDV: A Multi-Factors Data Valuation Method. In *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pages 48–53.

Maron, M. E. and Kuhns, J. L. (1960). On Relevance, Probabilistic Indexing and Information Retrieval. *J. ACM*, 7(3):216–244.

Miutra, B. and Craswell, N. (2018). An Introduction to Neural Information Retrieval. *Found. Trends Inf. Retr.*, 13(1):1–126.

Odu, G. O. (2019). Weighting methods for multi-criteria decision making technique. *Journal of Applied Sciences and Environmental Management*, 23(8):1449–1457.

Pitkow, J., Schütze, H., Cass, T., Cooley, R., Turnbull, D., Edmonds, A., Adar, E., and Breuel, T. (2002). Personalized search: A contextual computing approach may prove a breakthrough in personalized search efficiency. *Commun. ACM*, 45(9):50–55.

Pretschner, A. and Gauch, S. (1999). Ontology based personalized search. In *Proceedings 11th International Conference on Tools with Artificial Intelligence*, pages 391–398. ISSN: 1082-3409.

Qiu, S., Zhang, D., and Du, X. (2017). An Evaluation Method of Data Valuation Based on Analytic Hierarchy Process. In *2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing (ISPAN-FCST-ISCC)*, pages 524–528. ISSN: 2375-527X.

Roberts, S. W. (1959). Control Chart Tests Based on Geometric Moving Averages. *Technometrics*, 1(3):239–250.

Robertson, S. E., van Rijsbergen, C. J., and Porter, M. F. (1980). Probabilistic models of indexing and searching. In *Proceedings of the 3rd annual ACM conference on Research and development in information retrieval*, SIGIR '80, pages 35–56, GBR. Butterworth & Co.

Saaty, R. W. (1987). The analytic hierarchy process—what it is and how it is used. *Mathematical Modelling*, 9(3):161–176.

Salton, G., Fox, E. A., and Wu, H. (1983). Extended Boolean information retrieval. *Commun. ACM*, 26(11):1022–1036.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., USA.

Shen, X., Tan, B., and Zhai, C. (2005). Implicit user modeling for personalized search. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 824–831, New York, NY, USA. Association for Computing Machinery.

Singh, A. and Joachims, T. (2018). Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 2219–2228, New York, NY, USA. Association for Computing Machinery.

Tamine, L. and Goeuriot, L. (2021). Semantic Information Retrieval on Medical Texts: Research Challenges, Survey, and Open Issues. *ACM Comput. Surv.*, 54(7):146:1–146:38.

Tanudjaja, F. and Mui, L. (2002). Persona: A Contextualized and Personalized Web Search. pages 67–67. IEEE Computer Society.

Turczyk, L., Groepl, M., Liebau, N., and Steinmetz, R. (2007). A method for file valuation in information lifecycle management.

Wang, B., Guo, Q., Yang, T., Xu, L., and Sun, H. (2021). Data valuation for decision-making with uncertainty in energy transactions: A case of the two-settlement market system. *Applied Energy*, 288:116643.

Wang, H., He, X., Chang, M.-W., Song, Y., White, R. W., and Chu, W. (2013). Personalized ranking model adaptation for web search. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '13, pages 323–332, New York, NY, USA. Association for Computing Machinery.

Wang, T., Rausch, J., Zhang, C., Jia, R., and Song, D. (2020). A principled approach to data valuation for federated learning. *Federated Learning: Privacy and Incentive*, pages 153–167. Publisher: Springer.

Wang, X., Golbandi, N., Bendersky, M., Metzler, D., and Najork, M. (2018). Position Bias Estimation for Unbiased Learning to Rank in Personal Search. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 610–618, New York, NY, USA. Association for Computing Machinery.

Zehlike, M., Yang, K., and Stoyanovich, J. (2022a). Fairness in Ranking, Part I: Score-Based Ranking. *ACM Comput. Surv.*, 55(6):118:1–118:36.

Zehlike, M., Yang, K., and Stoyanovich, J. (2022b). Fairness in Ranking, Part II: Learning-to-Rank and Recommender Systems. *ACM Comput. Surv.*, 55(6):117:1–117:41.

APPENDIX

AHP Explained

AHP stands for Analytic Hierarchy Process and was first introduced by Saaty (1987). It is used to calculate the relative weights of the criteria in a multi-criteria decision setting. For instance, a multi-criteria decision consists of choosing the best dataset among multiple datasets considering their currency, size, and usage frequency, simultaneously.

AHP has 5 main components:

1. **Criteria.** Selection of the criteria to be considered in the decision making.
2. **Pairwise Comparisons of the Criteria.** This consists of comparing each criterion to all the other criteria. There are $\frac{n(n-1)}{2}$ comparisons needed for n criteria.
3. **Judgement Matrix** P
   - P is reciprocal: $P(i,j) = 1/P(j,i)$
   - The diagonal elements of P are equal to 1
   - Each element of P is a strictly positive real number:
     - $P(i,j) = 1$ means criteria i and j are equivalent
     - $P(i,j) < 1$ means criterion i is less important than criterion j
     - $P(i,j) > 1$ means criterion i is more important than criterion j
4. **Criteria Weights.** The weights are calculated using the judgement matrix P. The details of the calculation steps can be found in (Qiu et al.; Saaty (2017; 1987)).
5. **Consistency Ratio (CR).** CR should be less than or equal to 0.1 or 10%. It measures the transitive consistency.
   - Transitivity: if $a = 2b$ and $b = 3c$, then $a = 6c$
   - CR = 0 iff P is transitively consistent. Then $P(i,j) = P(i,k) \times P(k,j)$, for all i, j, and k.

With one row or column vector from the judgement matrix P (a vector of n elements with at least one element equal to 1), one can fill out the rest of the judgement matrix P using its reciprocity and transitivity properties. This is how we derived the AHP weights shown in Table 4.

A Systematic Literature Review on LLM-Based Information Retrieval: The Issue of Contents Classification

Diogo Cosme[1][a], António Galvão[2][b] and Fernando Brito E Abreu[1][c]
[1] *ISTAR-IUL, Instituto Universitário de Lisboa (Iscte-IUL), Av. das Forças Armadas, 40, 1649-026 Lisboa, Portugal*
[2] *CENSE, School of Science and Technology, NOVA University Lisbon, 2829-516 Caparica, Portugal*
dfmce@iscte-iul.pt, amg13172@campus.fct.unl.pt, fba@iscte-iul.pt

Keywords: Systematic Literature Review, Large Language Model, Information Retrieval, Contents Classification.

Abstract: This paper conducts a systematic literature review on applying Large Language Models (LLMs) in information retrieval, specifically focusing on content classification. The review explores how LLMs, particularly those based on transformer architectures, have addressed long-standing challenges in text classification by leveraging their advanced context understanding and generative capabilities. Despite the rapid advancements, the review identifies gaps in current research, such as the need for improved transparency, reduced computational costs, and the handling of model hallucinations. The paper concludes with recommendations for future research directions to optimize the use of LLMs in content classification, ensuring their effective deployment across various domains.

1 MOTIVATION

Generative AI (GenAI), particularly LLMs, which were designed for Natural Language Processing (NLP) tasks, has changed the paradigm of Information Retrieval (IR). An interesting list of IR topics and themes based on LLMs is presented in (Liu et al., 2024). Notably, automatic content classification has improved thanks to LLMs. Before their rise, achieving accurate and efficient content classification, mainly of textual content, was challenging. LLMs have successfully overcome these limitations.

Besides being trained on vast amounts of data, most LLMs follow the transformer architecture (Vaswani et al., 2017). According to NVIDIA, *"70 percent of arXiv papers on AI posted in the last two years mention transformers" (March 25, 2022).* These models effectively capture context and dependencies using self-attention mechanisms, excelling in NLP tasks, text generation, and context understanding. The key concepts of the transformer models are:

- **Model Architecture:** It can be encoder-only, designed to understand the meaning and context of each word in relation to others, making it suitable for classifying texts, answering questions, and other comprehension-based applications. It can also be decoder-only and used to generate a new sequence of words, making it suitable for various generative tasks such as text generation, language modeling, and conversational agents. Lastly, combining both is also possible, resulting in encoder-decoder models. The foundational models[1] that stand out in each architecture are, respectively: BERT (Devlin et al., 2019), GPT (Radford et al., 2018), and BART (Lewis et al., 2020).
- **Adapting a LLM:** There are two main ways to specialize a LLM for specific tasks. One method is fine-tuning the model, which consists of adjusting the model's weights based on the new data. The larger the model, the greater the computing resources required. A more resource-efficient alternative, though potentially less effective, is In-Context Learning (ICL). It involves giving the model examples of the task during inference[2] without additional training, allowing it to learn from these examples. It can receive zero examples (zero-shot), i.e., the hy-

[a] https://orcid.org/0009-0001-1245-286X
[b] https://orcid.org/0000-0002-6566-9114
[c] https://orcid.org/0000-0002-9086-4122

[1] A foundational model refers to a large, pre-trained model that serves as a starting point or base for various specialized tasks and applications. These models are typically trained on vast amounts of data and are designed to capture general patterns and features that can be fine-tuned for specific use cases.

[2] Inference in the context of LLMs refers to generating a response or prediction based on a given input.

Cosme, D., Galvão, A. and Brito E Abreu, F.
A Systematic Literature Review on LLM-Based Information Retrieval: The Issue of Contents Classification.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 135-146
ISBN: 978-989-758-716-0; ISSN: 2184-3228

pothesis that the model is already capable is tested, or it can receive some examples (few-shot).

Due to the immense potential and inherent complexities of LLMs, it is essential to evaluate or conduct literature reviews to support the field of LLM-based content classification, especially for textual content. By understanding the current landscape and methodologies, researchers can realize LLMs' full potential and ensure their applications are innovative and effective in various fields. To check if the characterization of that landscape (aka state of the art) was already performed, we searched for literature reviews on this topic in the SCOPUS database using this search string:
"literature review" AND ("information retrieval" OR "contents classification" OR "topics classification") AND (LLM OR "large language model" OR "foundational model" OR GPT)

We obtained ten hits, but only two corresponded to literature reviews (Mahadevkar et al., 2024; Yu et al., 2023). However, none of these were about LLM-based content classification. On (Yu et al., 2023), a literature review addressed the critical need for guidelines for incorporating LLMs and GenAI into healthcare and medical practice. In contrast, a systematic literature review on (Mahadevkar et al., 2024) identified potential research directions for information extraction from unstructured documents.

In summary, the importance of LLM-based content classification and the lack of previous literature reviews on this topic motivated us to write this paper. It is organized as follows: Section 2 describes the review methodology used to identify and conduct the study; Section 3 analyzes the studies obtained; and Section 4 provides a summary of the existing research and identifies the threats to this literature review.

2 METHODOLOGICAL APPROACH

A systematic literature review (SLR), in contrast to an unstructured review approach, reduces bias by following a strict and methodical sequence of stages for conducting literature searches (Wohlin, 2014; Kitchenham and Brereton, 2013). The ability of an SLR to methodically search, extract, analyze, and document findings in stages depends on carefully designed and evaluated review protocols. The technique for these efforts is described in this section.

2.1 Planning the Review

2.1.1 Research Questions

The following research questions were formulated:

- **RQ1:** What type of empirical studies have been conducted in LLM-based content classification?
- **RQ2:** How extensive is the research in this area?
- **RQ3:** What were the relevant contributions of the existing studies?
- **RQ4:** Can LLMs be used to assess the quality of studies?

2.1.2 Review Protocol

Based on the research conducted by (Stahlschmidt and Stephen, 2020), Scopus offers more extensive subject coverage than Web of Science and Dimensions, encompassing the majority of articles found in these two databases. As a result, we chose to use the Scopus database exclusively for our formal literature search.

2.1.3 Search String

Keywords were derived from the research questions and used to search the primary study source. The search string included the most important terms related to the research questions, including synonyms, related terms, and alternative spellings.

To carry out the intended research, the following search string was drawn up:
("Large Language Model" OR "Foundational Model") AND ("Contents Classification" OR "Topic Classification")

2.1.4 Inclusion Criteria

A careful review of the abstracts and overall structure of the studies was conducted to determine their relevance to our research. The decision to include a study in our selection was based on the fulfillment of the following inclusion criteria: be written in English; be a primary study; match at least one of the literature review objectives; be the most up-to-date and comprehensive version of the document.

2.1.5 Data Extraction

The *Elicit* AI Research Assistant was used to extract details from papers into an organized table. According to its website, it has been used by more than 2 million researchers. Besides, it is claimed that *Elicit* uses various strategies to reduce the rate of hallucinations such as *"process supervision, prompt engineering, ensembling multiple models, double-checking our results with custom models and internal evaluations, and more to reduce the rate of hallucinations"*. This indicates that it is a robust and trustworthy AI solution for summarizing, finding, and extracting details from scientific articles.

Elicit allows us to extract several details from scientific articles, but we have only selected these: research question; summary of introduction; dataset; limitations; research gaps; software used; algorithms; methodology; main findings; Study Objectives; study design; intervention effects; hypotheses tested; experimental techniques.

All the information extracted with *Elicit* is available online here (Cosme et al., 2024).

2.1.6 Quality Assessment

Despite the limited number of articles under review, the studies from the preceding phase were evaluated and analyzed to gauge their quality.

The quality assessment of the studies consists of 7 questions (see box with **Prompt 1** and box with **Prompt 2**), each to be answered with a score from an ordinal scale: 0—Strongly Disagree, 1—Disagree, 2—Neither Agree nor Disagree, 3—Agree, 4—Strongly Agree.

Since the main objective of our scientific research involves using LLMs, we decided to carry out a performance comparison test to evaluate the quality of articles between a manual assessment and an LLM-based one.

The information extracted from *Elicit* was then used as a basis for the manual and LLM-based quality assessment. For the LLM-based evaluation, we carried it out using prompting combined with the ICL Zero-shot technique, as this is the fastest and most cost-effective approach compared to fine-tuning and few-shot ICL techniques.

The prompt template used, which is outlined below (**Prompt 1**), is organized in the following manner: it begins with an introduction to the task, followed by the expected output that the LLM should produce, a JSON object where each key represents a question indicator, and the values are the assigned scores. Lastly, for every article, the term *"""ARTICLE"""* is substituted with the corresponding JSON object, in which each key signifies an *Elicit* field, and the values are the related information. An important note is that none of the available *Elicit* fields refer to related work, so it is impossible to answer Q2 the same way as the other questions.

> Prompt 1
>
> Your task is to assess the quality of a study article based on the information provided. You'll receive two JSON objects:
> 1 - A JSON object with question indicators as keys and the corresponding questions as values.
> 2 - Another JSON object containing information about the article, where keys represent specific parameters.
> Your goal is to assign to each question a score from 0 to 4 (0 - strongly disagree, 1 - disagree, 2 - neither agree nor disagree, 3 - agree, 4 - strongly agree).
> Please provide your evaluation in the following JSON format: {"Q1": <score>, "Q2": <score>, . . . }.
> Questions: {
> "Q1": "Were the study's goals and research questions clearly defined?"
> "Q3": "Was the research design clearly outlined?"
> "Q4": "Were the study limitations evaluated and identified?"
> "Q5": "Was the data used for validation described in sufficient detail and made available?"
> "Q6": "Were answers to the research questions provided?"
> "Q7": "Were negative or unexpected findings reported about the study?"
> }
> Article:
> """ARTICLE"""
> Please provide the requested JSON.

Microsoft Copilot was the LLM used. For Q2, the procedure was as follows: via the Copilot sidebar section in the Microsoft Edge browser, we can restrict the relevant information sources to the open page only, which in this case is a PDF opened in Microsoft Edge. We then provided **Prompt 2** (see the corresponding box).

> Prompt 2
>
> Your task is to assign a score from 0 to 4 (0 - strongly disagree, 1 - disagree, 2 - neither agree nor disagree, 3 - agree, 4 - strongly agree) to a question from a study quality assessment about this article. Besides the score, you must provide a detailed justification and identify the sections or pages (if possible both) that contribute to your answer.
> The question is: "Was previously published related work exposed and compared with the research results claimed in the study?"

2.2 Conducting the Review

2.2.1 Execute Search

Applying the specified search string resulted in the retrieval of nineteen scientific articles. Seven studies were rejected, and twelve articles were accepted.

One of the accepted studies, (Russo et al., 2023), is an overview of a challenge in which several teams presented their approach to classifying the content of messages as conspiratorial or non-conspiratorial and their conspiratorial type. So, articles of that challenge relevant to the research topic that did not appear in the search string results and fulfill the inclusion criteria have been added. This resulted in a total of thirteen accepted articles.

2.2.2 Apply Quality Assessment

Figure 1 shows the mean absolute score difference between the two methods (LLM and manual) for each question, highlighting the response variability. A lower difference indicates that the responses, while not identical, are relatively similar. Inversely, a higher difference indicates significant variability in responses. A red line is drawn at a mean absolute difference of 0.5 to help visualize the variability. We consider an average difference of 0.5 or less across the 13 studies to be a strong indicator of agreement between the methods. For example, for questions Q1 and Q6, the number of questions without agreement was 4 for each.

Nevertheless, analyzing the mean scores assigned to each question by method is also helpful in understanding the performance (Figure 2). Both graphs show that Q7 has the most significant disparity, with the highest mean absolute score difference between the two methods and the largest gap between the mean scores (|2.77 - 1.08| = 1.69). Given that Q7 relates to identifying negative or unexpected findings in the study, the higher scores assigned by the LLM-based method may indicate that LLMs have difficulty penalizing score assignments. Q4 shows a minimal difference in average scores, with |3.08 - 3.00| = 0.08, but a mean absolute score difference of 0.54. This discrepancy occurs because one study had opposite responses (4 vs 0), significantly affecting the mean absolute score difference.

This suggests that the most effective way to evaluate performance on this test is to examine the mean absolute difference in scores. For example, if Study X scored 2 and 4 on the same question using the LLM and Manual methods, respectively, and Study Y scored 4 and 2, the difference between the mean scores would be 0: *3 - 3 = 0*. However, the mean absolute difference would be 2: *(| 4 - 2 | + | 2 - 4 |) / 2 = 2*. In other words, focusing only on the difference between the average scores could misleadingly suggest that the LLMs gave the same answers as humans, when in fact they did not.

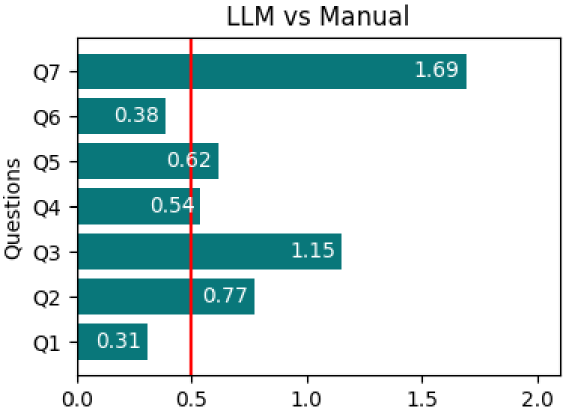


Figure 1: Mean Absolute Score Difference Between Methods Per Question.

The data obtained in the comparison between manual (***M***) and LLM (***L***) analysis is available online here (Cosme et al., 2024).

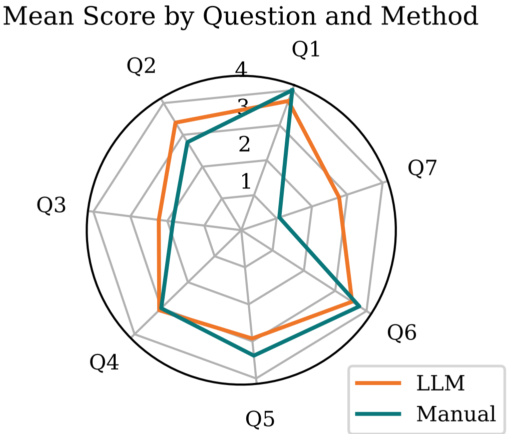


Figure 2: Radar Chart Displaying the Average Scores Given to the Studies by M and L.

Although the results indicate that using ICL zero-shot is not yet reliable, we conclude that assessing the quality of scientific articles with LLMs may be feasible. This could be achieved through more extensive research with a fine-tuned model or by using ICL few-shot examples.

Due to the few studies, this task did not remove any studies and was only useful for assessing their overall quality.

3 DOCUMENT THE REVIEW

3.1 Demographics

Figure 3 illustrates that all studies are collaborative efforts with multiple authors, with most having two authors. There are also two rare cases with many researchers (16). Regarding the authors' affiliation (Figure 5), the most common scenario involves one or two institutions. The relatively low number of institutions compared to the number of authors suggests a gap in inter-institutional collaboration that could improve research. This is further emphasized by the lack of international partnerships, with only one article involving cooperation between teams from Indonesia and Turkey. Regarding authors' affiliation countries, while no single country dominates, Europe emerges as the most active continent (Figure 4).

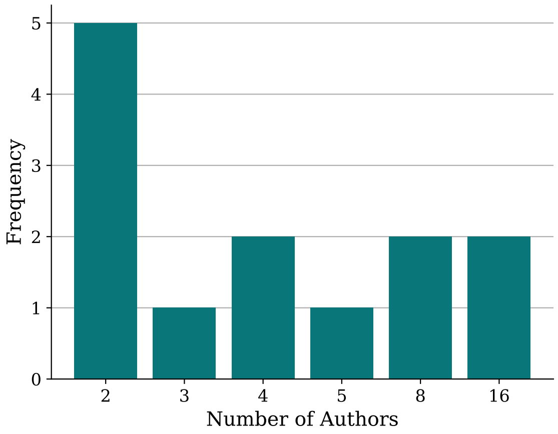


Figure 3: Publication Frequency by Authors Count.

Figure 6 clearly shows that most selected studies were published in workshops and journals. It should be remarked that three articles come from the same workshop (EVALITA 2023). This "high concentration" in a single workshop may indicate the topic is still niche, with limited venues for broader exposure. It can also be considered a sign that a community is emerging, with the possibility of broader interest in the future.

3.2 Analysis and Findings

A methodology was proposed in (Rodríguez-Cantelar et al., 2023) to address the problem of inconsistent responses in chatbots. It consists of hierarchical topic/subtopic detection using zero-shot learning (through GPT-4), and detecting inconsistent answers using clustering techniques. The datasets used in the study were the DailyDialog corpus (Li et al., 2017)

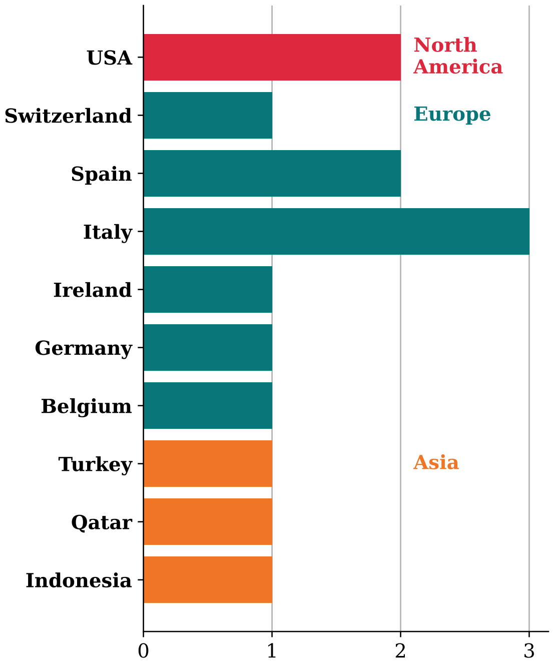


Figure 4: Publication Frequency by Author Affiliations' Country.

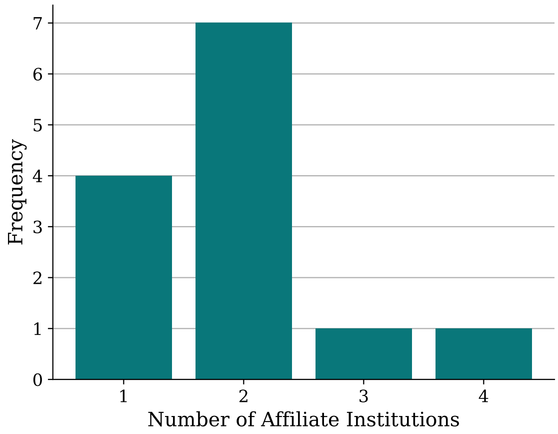


Figure 5: Publication Frequency Affiliates Count.

and data collected by the authors' Thaurus bot during the Alexa Prize Socialbot Challenge (SGC5). Using the *DailyDialog* dataset, the authors achieved a weighted F1 score of 0.34 for topic detection and 0.78 for subtopic detection. The SGC5 dataset obtained an accuracy of 81% and 62% for topic and subtopic detection, respectively. Notably, there is room for improvement in the *DailyDialog* topic detection, as the authors recorded a lower weighted F1 score, indicating a significant number of false positives or false negatives.

An overview of the EVALITA 2023 challenge "Automatic Conspiracy Theory Identification

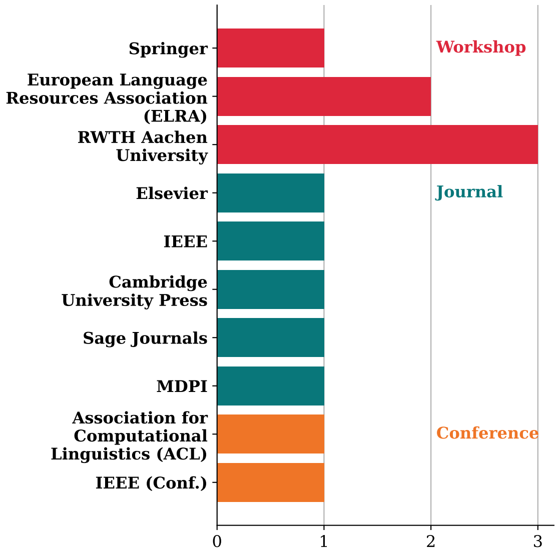


Figure 6: Publication Frequency by Publisher.

(ACTI)" is presented in (Russo et al., 2023). The challenge focuses on identifying whether an Italian message contains conspiratorial content (Subtask A) and, if so, classifying it into one of four possible conspiracy topics: "*Covid*", "*Qanon*", "*Flat Earth*", or "*Pro-Russia*" (Subtask B). A total of eight teams participated in Subtask A and seven teams in Subtask B. The provided dataset was the same for each team and each task. It used a collection of Italian comments scraped from 5 Telegram channels known for hosting conspiratorial content, collected between January 1, 2020, and June 30, 2020. The comments were manually annotated by two human annotators to identify conspiratorial content (as "*Not Relevant*", "*Non-Conspiratorial*" or "*Conspiratorial*") and categorize it into specific conspiracy theories. The authors calculated inter-annotator agreement rates using Cohen's Kappa coefficient to evaluate the consistency among annotators. They achieved high agreement levels: a Cohen's Kappa of 0.93 for Subtask A and 0.86 for Subtask B. For data integrity reasons, comments that didn't receive the same classification were excluded, and "Not Relevant" comments were also discarded to focus solely on relevant conspiratorial content. The final datasets consist of 2,301 comments labeled with a binary label for Subtask A and 1,110 comments labeled with a value from 0 to 3, representing the specific conspiracy topic. The articles in this challenge that are relevant to the subject of this paper are:

- The authors of (Cignoni and Bucci, 2023) compared the performance between two fine-tuned encoder-only transformer models (bert-base-italian-xxl-cased and XLM-RoBERTa (Conneau et al., 2020)) and a non fine-tuned decoder-only transformer model (LLaMA 7B (Touvron et al., 2023)). The BERT models achieved a higher test score than the LLaMa model in both subtasks. For Subtask A: 0.83, 0.82 and 0.80, respectively. For Subtask B: 0.83, 0.85 and 0.74, respectively. The article does not provide details regarding the study's limitations and how LLaMa was used.
- In (Hromei et al., 2023), the authors took a distinct approach. Initially, they introduced a model to address all tasks in the EVALITA 2023 challenge, not just the ACTI task. Consequently, their dataset was significantly larger than the one provided for the ACTI task, comprising 134,018 examples from various tasks. For each task, the authors compared the performance of two models. One is an encoder-decoder model named *extremIT5*, based on IT5, consisting of approximately 110 million parameters. It was fine-tuned by concatenating task names and input texts to generate text solving the target tasks. The other model is a decoder-only model named *extremITLLaMA*, based on LLaMa 7B. It was first trained on Italian translations of Alpaca instruction data using LoRA (Low-Rank Adaptation)[3](Hu et al., 2022), to enable the model to comprehend instructions in Italian. Then, it is further fine-tuned using LoRA on instructions reflecting the EVALITA tasks. In their final results, the authors achieved an F1 score of 0.82 for Subtask A using *extremIT5* and 0.86 with *extremITLLaMA*. For Subtask B, the F1 scores were 0.81 and 0.86, respectively. The biggest limitations of this study are the computational cost and inference speed of the larger *extremITLLaMA* model and the limited exploration of architectures and hyperparameters due to time constraints. In conclusion, the authors suggest that exploring zero-shot or few-shot learning could benefit sustainability, as it reduces the need for large amounts of annotated data.

For Subtask A, the approach in (Cignoni and Bucci, 2023) achieved the sixth rank, while the one in (Hromei et al., 2023) secured the second position. For Subtask B, their rankings were fourth and fifth. The winning team in both subtasks employed an approach that leveraged data augmentation through LLMs.

In (Trust and Minghim, 2023), query-focused submodular mutual information functions are proposed to select diverse and representative demonstration examples for ICL in prompting. In addition, an interactive tool is presented to explore the impact of

[3] LoRA fine-tuning significantly reduces the computational and storage costs of training large language models by only adjusting a subset of low-rank parameters.

hyperparameters on model performance in ICL. For evaluation purposes, the authors have applied their method to the following tasks: two sentiment classification tasks with Stanford Sentiment Treebank datasets (SST-2 and SST-5) (Socher et al., 2013), and a topic classification task with the AG News Classification Dataset (Zhang et al., 2015). Their methodology consists of the following two steps.

i. **Retrieval:** The goal here is to, based on the input test, select representative and diverse in-context demonstration examples from the training data. The input test and the training dataset undergo embedding via the sentence transformer (Reimers and Gurevych, 2019) to achieve this. Subsequently, specialized selection occurs by leveraging Submodular Mutual Information (SMI) functions to choose examples from the training data. The selected examples are then incorporated into a prompt template alongside an optional task directive or as stand-alone demonstrations.

ii. **Inference:** The prompt template and input test are fed into a pre-trained language model to deduce the corresponding label. They used three open-source pre-trained models: GPT-2 (Radford et al., 2019), OPT (Zhang et al., 2023), and BLOOM (Le Scao et al., 2022).

According to the authors, their approach can yield performance enhancements of up to 20% when compared to random selection or conventional prompting methods, and the size and type of the language model do not always guarantee better performance.

A transit-topic-aware language model that can classify open-ended text feedback into relevant transit-specific topics based on traditional transit Customer Relationship Management (CRM) feedback is proposed in (Leong et al., 2024). The primary dataset includes around 180,000 anonymous customer feedback comments, manually labeled, from the Washington Metropolitan Area Transit Authority (WMATA) CRM database, covering January 2017 to December 2022. Given 61 distinct labels, the authors used Latent Dirichlet Allocation (LDA) to group customer feedback into broader topics. Due to the limitation of LDA in detecting significantly less represented topics, these topics were excluded from the CRM dataset before applying LDA and grouped according to their original topic (2 niche groups). LDA failed to identify a primary topic for approximately 62,000 complaints. As a result, the final dataset included around 120,000 complaints categorized into 11 topics (9 LDA-detected topics and two niche topics). They evaluated the performance of five ML models (Random Forest, Linear SGD, SVM, Naive Bayes, and Logistic Regression) against the proposed MetRoBERTA LLM. MetRoBERTA is a fine-tuned version, with the CRM dataset, of the RoBERTa LLM open-sourced by Meta Research (Liu et al., 2019). MetRoBERTA outperformed the traditional ML models with a macro average F1-score of 0.80 and a weighted average F1-score of 0.90, compared to the best ML model with 0.76 and 0.88, respectively. A significant limitation of this study is the exclusion of approximately 60,000 initial complaints, accounting for over one-third of the entire dataset.

The paper (Borazio et al., 2024) introduces a novel framework that uses LLMs to identify and categorize emergent socio-political phenomena during health crises, with a focus on the COVID-19 pandemic, and to provide explicit support to analysts through the generation of actionable statements for each topic. For this aim, they used a dataset of 2,254 news articles manually categorized by ISS (Istituto Superiore di Sanità) experts into five topics: "*Covid Variants*," "*Nursing Homes Outbreaks*," "*Hospital Outbreaks*," "*School Outbreaks*," and "*Family/Friend Outbreaks*," collected from February 2020 to September 2022. Then, their system generates linguistic triples to capture fine-grained concepts, which analysts can refine to correlate themes. For the following step, they have employed a model based on BART (Lewis et al., 2020) and previously trained on the Multi-Genre Natural Language Inference corpus (Williams et al., 2018). The model uses zero-shot classification to associate news articles with the identified topics without fine-tuning. Preliminary results demonstrate accurate mapping of news articles to specific, detailed topics. The system achieved an accuracy of 67% when proposing a single class, which increased to 88% when considering the top two system suggestions. However, the authors acknowledge potential limitations, including hallucinations from integrating a decoder LLM (GPT-4) for prompting generation.

The benchmarking study LAraBench (Abdelali et al., 2024) addresses the gap in comparing LLMs against state-of-the-art (SOTA) models used already for Arabic natural language processing and speech processing tasks. 61 publicly available datasets were used to support 9 task groups: Word Segmentation, Syntax and Information Extraction; Machine Translation; Sentiment, Stylistic and Emotion Analysis; News Categorization; Demographic Attributes; Factuality, Disinformation and Harmful Content Detection; Semantics; Question Answering; Speech Processing. The models GPT-3.5-Turbo, GPT-4, BLOOMZ, and Jais-13b-chat were used for NLP tasks combined with zero and few-shot learning. Following the recommended format from Azure OpenAI Studio Chat playground and PromptSource (Bach et al., 2022), various prompts were explored, and the most reasonable one was selected. The study revealed that in specific multilabel tasks, like propaganda detection, the LLMs sometimes generated outputs that did not fit the predefined labels. Besides that, they mention that deploying LLMs seamlessly requires substantial effort in crafting precise prompts or post-processing to align outputs with reference labels. While GPT-4 has made significant strides by closing the gap with state-of-the-art models and outperforming them in high-level abstract tasks like news categorization, consistent SOTA performance in sequence tagging remains challenging. In addition, the authors registered an averaged macro-F1 improvement from 0.656 to 0.721 by using few-shot learning (10-shot) instead of zero-shot learning.

In (Peña et al., 2023), the potential of LLMs to enhance the classification of public affairs documents is studied. The researchers gathered raw data from the Spanish Parliament, spanning November 2019 to October 2022. They acquired approximately 450,000 records, with only around 92,500 of them labeled. They concentrated on the 30 most frequent topics out of 385 labels to mitigate the impact of significant class imbalances. As models, they have used four transformer models pre-trained from scratch in Spanish by the Barcelona Supercomputing Center in the context of the MarIA project (Gutiérrez-Fandiño et al., 2022): RoBERTa-base, RoBERTa-large, RoBERTalex, and GPT2-base. Their approach involves employing transformer models in conjunction with classifiers. They conducted experiments using four models combined with three classifiers (Neural Networks, Random Forests, and SVMs). The results demonstrate that utilizing an LLM backbone alongside SVM classifiers is an effective strategy for multi-label topic classification in public affairs, achieving accuracy exceeding 85%.

An improvement of the GPT-3 performance on a short text classification task, using data augmentation, is explored in (Balkus and Yan, 2023). The authors pretend to classify whether a question is related to data science by comparing two approaches: augmenting the GPT-3 Classification Endpoint by increasing the training set size and boosting the GPT-3 Completion Endpoint by optimizing the prompt using a genetic algorithm. Both methods are accessible via the GPT-3 API, each with advantages and drawbacks. The Completion Endpoint relies on a text prompt followed by ICL (zero-shot or few-shot), but its performance is notably influenced by the specific examples included. In contrast, the Classification Endpoint utilizes text embeddings and offers more consistent performance, although it necessitates a substantial number of examples (hundreds or thousands) to achieve optimal results. The dataset used in the study consists of 72 short text questions collected from the University of Massachusetts Dartmouth Big Data Club's Discord server. In Classification Endpoint Augmentation, GPT-3 was employed to generate new questions. Among the approaches, the embedding-based GPT-3 Classification Endpoint achieved the highest accuracy, approximately 76%, although this falls short of the estimated human accuracy of 85%. On the other hand, the GPT-3 Completion Endpoint, optimized using a genetic algorithm for in-context examples, exhibited strong validation accuracy but lower test accuracy, suggesting potential overfitting.

The study in (Nasution and Onan, 2024) presents a comparison on the quality of annotations generated by humans and LLMs for Turkish, Indonesian, and Minangkabau NLP tasks (Topic Classification, Tweet Sentiment Analysis, and Emotion Classification). In their study, the authors used three Turkish datasets, each designed for one of the NLP tasks. Additionally, they employed two Indonesian datasets: one customized for Tweet Sentiment Analysis and the other for Emotion Classification. Furthermore, they included two Minangkabau datasets translated from the Indonesian datasets. The study employed the following LLMs: ChatGPT-4, BERT (Devlin et al., 2019), BERTurk (a fine-tuned Turkish version of BERT), RoBERTa (Liu et al., 2019) (fine-tuned on specific datasets), and T5 (Mastropaolo et al., 2021). Human annotations consistently outperformed LLMs across various evaluation metrics, serving as the benchmark for annotation quality. While ChatGPT-4 and BERTurk demonstrated competitive performance, they still fell short of human annotations in certain aspects. The trade-off between precision and recall was observed among the LLMs, highlighting the need for better balance in these two measures.

The use of LLMs for moderating online discussions is investigated in (Gehweiler and Lobachev, 2024). The focus is on identifying user intent in various types of content and exploring content classification methods. As data sources, the authors have used various datasets, such as the One Million Posts Corpus dataset by the Austrian Research Institute for Artificial Intelligence (OFAI) of German comments made on the Austrian newspaper website's (Schabus et al., 2017). Another dataset used was the New York Times Comments collection with over two million comments on over 9,000 articles. The LLMs they used were obtained from the Detoxify python library. Their research highlights effective LLM approaches

Table 1: Articles summary information.

Article	**Method**	**Evaluation Metrics**	**Description**
(Rodríguez-Cantelar et al., 2023)	ICL	Weighted F1	Topic: 0.34; Subtopic: 0.78 (DailyDialog)
		Accuracy	Topic: 81%; Subtopic: 62% (SGC5)
(Cignoni and Bucci, 2023)	Fine-tuning	Macro-avg F1	Subtask A: 0.83, 0.82 and 0.80, respectively.
			Subtask B: 0.83, 0.85 and 0.74, respectively.
(Hromei et al., 2023)	Fine-tuning	F1	Subtask A: 0.82 (extremIT5); 0.86 (extremITLLaMA).
			Subtask B: 0.81 (extremIT5); 0.86 (extremITLLaMA)
(Trust and Minghim, 2023)	ICL	F1	Sentiment Classification: 88.35%.
			Topic Classification: 90.56%.
(Leong et al., 2024)	Fine-tuning	Macro-avg F1	0.80 compared to the best ML model with 0.76
		Weighted F1	0.90 compared to the best ML model with 0.88
(Borazio et al., 2024)	ICL	Accuracy	Single Class: 67%; Top two system suggestions: 90.56%.
(Abdelali et al., 2024)	ICL	Macro-avg F1	Few-shot (10-shot): 0.721; Zero-shot: 0.656.
(Peña et al., 2023)	Fine-tuning	Accuracy	Accuracies higher than 85%.
(Balkus and Yan, 2023)	ICL	Accuracy	LLM: 76%; Estimated Human: 85%.
(Nasution and Onan, 2024)	Fine-tuning; ICL	Avg F1	Human: 0.883; GPT-4: 0.865.
(Gehweiler and Lobachev, 2024)	Fine-tuning	F1	Identifying user intent: 0.755.
(Van Nooten et al., 2024)	Fine-tuning; ICL	F1 score	Zero-shot experiments lag behind fine-tuned models.

for discerning authors' intentions in online discussions and that fine-tuned AI models, based on extensive data, show promise in automating this detection.

The authors of (Van Nooten et al., 2024) report their results for classifying the Corporate Social Responsibility (CSR) Themes and Topics shared task, which encompasses cross-lingual multi-class and monolingual multi-label classification. The shared task involved two subtasks: cross-lingual, multi-class classification for recognizing CSR themes (using one dataset) and monolingual multi-label text classification of CSR topics related to Environment (ENV) and Labour and Human Rights (LAB) themes (using two datasets). For text classification, the LLMs used were GPT-3.5 and GPT-4 (both zero-shot and without fine-tuning), as well as fine-tuned versions of DistilBERT (Sanh et al., 2019), BERT (Devlin et al., 2019), RoBERTa, and RoBERTa-large (Liu et al., 2019). For the themes dataset, the authors used fine-tuned versions of Multi-Lingual DistilBERT, XLM-RoBERTa, and XLM-RoBERTa-large (Conneau et al., 2020). Their zero-shot experiments with GPT models show they still lag behind fine-tuned models in multi-label classification.

Table 1 shows the training methods used, the evaluation metrics, and the main results of this evaluation.

4 CONCLUSIONS

4.1 Recap of Research Questions

RQ1: What Type of Empirical Studies Have Been Conducted in LLM-Based Content Classification? Although the number of studies is limited, their analysis reveals a wide variety of methodologies, including different approaches (e.g., ICL vs. fine-tuning, prompting strategies) and model architectures (encoder-only, encoder-decoder, decoder-only), as well as research areas explored:

- Hierarchical topic/subtopic detection in inconsistent chatbot responses (Rodríguez-Cantelar et al., 2023)
- Socio-political phenomena during health crises (Borazio et al., 2024);

- Public affairs documents (Peña et al., 2023);
- Customer feedback (Leong et al., 2024);
- Corporate Social Responsibility themes and topics (Van Nooten et al., 2024);
- Conspiracy Content (Cignoni and Bucci, 2023; Hromei et al., 2023)
- Sentiment (Trust and Minghim, 2023; Nasution and Onan, 2024)
- Emotion (Nasution and Onan, 2024)
- Benchmarking of NLP and speech processing tasks (Arabic) (Abdelali et al., 2024)
- Short questions (Balkus and Yan, 2023)
- User intent in online discussions (Gehweiler and Lobachev, 2024)
- Comparison of generated annotations (Nasution and Onan, 2024)

RQ2: How Extensive Is the Research in this Area? Although there are currently only a few approaches to topic/content classification using LLMs, this field is emerging. We believe it will grow and improve significantly in the future.

RQ3: What Were the Relevant Contributions of the Existing Studies? Based on the available studies, fine-tuned LLMs outperform LLMs prompted with ICL techniques (Balkus and Yan, 2023; Van Nooten et al., 2024). When fine-tuning models, it is essential to carefully consider the choice between an encoder-only model, a decoder-only model, or an encoder-decoder model. Each architecture has distinct characteristics and implications for the model's behavior and performance. However, achieving optimal performance requires substantial computational resources and a dataset containing hundreds or thousands of examples. LLMs can be prompted using zero-shot or few-shot techniques as a more cost-effective alternative. A comparison between these two methods for a specific case was conducted in (Abdelali et al., 2024), revealing that few-shot outperformed zero-shot. Notably, the selection of few-shot examples plays a crucial role (Trust and Minghim, 2023), and there are limitations related to the reasoning abilities of LLMs. Researchers (Abdelali et al., 2024; Borazio et al., 2024) reported challenges arising from model hallucinations.

RQ4: Can LLMs Be Used to Assess the Quality of Studies? While the results suggest that using ICL zero-shot is not yet reliable, we conclude that evaluating the quality of scientific articles with LLMs may be feasible. This could be achieved either through more extensive research with a fine-tuned model or by using ICL few-shot examples.

4.2 Threats to Validity

The following types of validity issues were considered when interpreting the results from this review.

Construct Validity: A literature database of relevant books, conferences, and journals served as the source for the research found in the systematic review. Therefore, bias in selecting publications is a potential drawback of this strategy, especially considering that three of the thirteen articles were submitted to the same workshop. To address this, we used a research protocol that included the study objectives, research questions, search approach, and search terms. Inclusion and exclusion criteria for data extraction were established to reduce this bias further.

Our dataset only includes studies published in the last two years (2023 and 2024), making it challenging to identify trends due to the recent and limited sample size. Moreover, the studies on LLM-based content classification only used well-established taxonomies, such as news categorization and fake news topics. None of the studies used a taxonomy the model had not encountered during its training process.

Internal Validity: No studies were excluded during the quality assessment due to the low number of documents retrieved in the search, so there is no potential threat to internal validity. In other words, we did not exclude studies that could contribute significantly despite their lower quality.

External Validity: There may be other valid studies in digital libraries that we did not search. However, we attempted to mitigate this limitation using the most relevant literature repository. Additionally, studies not written in English were excluded, which may have omitted important papers that would otherwise have been included.

Conclusion Validity: There may be some bias during the data extraction phase. However, we have addressed this by defining a data extraction form to ensure consistent and accurate data collection to answer the research questions. While there is always a small chance of inaccuracies in the numbers, we mitigate this by publishing our final dataset, allowing for replication and further validation.

4.3 Future Work

The use of LLMs in information retrieval is promising, as shown by recent studies and their years of publication. Future research should optimize LLMs for different domains, focusing on domain-specific fine-tuning and possibly hybrid models to maintain broad knowledge while adapting to specialized domains.

Improving the interpretability of LLM-based classifiers is critical because they often operate as black boxes, limiting trust in sensitive areas such as healthcare and finance. Creating explainability frameworks within LLM architectures can increase transparency and trust by clarifying classification decisions.

Ethical considerations are also critical. Research should focus on mitigating biases in LLM training data and outputs to ensure fair content classification.

Efficiency, scalability, and dynamic adaptation of LLMs are growing challenges. Future studies should improve computational efficiency through model compression or streamlined architectures, and explore continuous or reinforcement learning to help keep LLMs up to date with evolving content such as social media and news.

Lastly, enhancing cross-domain transfer learning can improve LLM adaptability across different applications. By refining these techniques, LLMs could become more versatile and excel at content classification across various industries.

ACKNOWLEDGEMENTS

This work was partially funded by the Portuguese Foundation for Science and Technology (FCT), under ISTAR-Iscte project UIDB/04466/2020 and CENSE NOVA-FCT project UIDB/04085/2020.

REFERENCES

Abdelali, A., Mubarak, H., Chowdhury, S. A., Hasanain, M., Mousi, B., Boughorbel, S., Abdaljalil, S., Kheir, Y. E., Izham, D., Dalvi, F., Hawasly, M., Nazar, N., Elshahawy, Y., Ali, A., Durrani, N., Milic-Frayling, N., and Alam, F. (2024). LAraBench: Benchmarking Arabic AI with Large Language Models. In Y., G., M., P., and M., P., editors, *Proc. of the 18th EACL Conf.*, volume 1, pages 487–520. ACL.

Bach, S., Sanh, V., Yong, Z. X., Webson, A., Raffel, C., Nayak, N. V., Sharma, A., Kim, T., Bari, M. S., Fevry, T., Alyafeai, Z., Dey, M., Santilli, A., Sun, Z., Ben-david, S., Xu, C., Chhablani, G., Wang, H., Fries, J., Al-shaibani, M., Sharma, S., Thakker, U., Almubarak, K., Tang, X., Radev, D., Jiang, M. T.-j., and Rush, A. (2022). PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts. In Basile, V., Kozareva, Z., and Stajner, S., editors, *Proc. of the 60th Annual Meeting of the ACL: System Demonstrations*, pages 93–104. ACL.

Balkus, S. V. and Yan, D. (2023). Improving short text classification with augmented data using GPT-3. *Natural Language Engineering*.

Borazio, F., Croce, D., Gambosi, G., Basili, R., Margiotta, D., Scaiella, A., Del Manso, M., Petrone, D., Cannone, A., Urdiales, A. M., Sacco, C., Pezzotti, P., Riccardo, F., Mipatrini, D., Ferraro, F., and Pilati, S. (2024). Semi-Automatic Topic Discovery and Classification for Epidemic Intelligence via Large Language Models. In *Proc. of PoliticalNLP@LREC-COLING Workshop*, pages 68–84.

Cignoni, G. and Bucci, A. (2023). Cicognini at ACTI: Analysis of techniques for conspiracies individuation in Italian. In *CEUR Workshop Proceedings*, volume 3473.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics*, pages 8440–8451. ACL.

Cosme, D., Galvão, A., and Brito e Abreu, F. (2024). Supplementary Data for "A Systematic Literature Review on LLM-Based Information Retrieval: The Issue of Contents Classification". *Zenodo*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Computing Research Repository (CoRR)*.

Gehweiler, C. and Lobachev, O. (2024). Classification of intent in moderating online discussions: An empirical evaluation. *Decision Analytics Journal*, 10.

Gutiérrez-Fandiño, A., Armengol-Estapé, J., Pàmies, M., Llop-Palao, J., Silveira-Ocampo, J., Carrino, C. P., Armentano-Oller, C., Rodriguez-Penagos, C., Gonzalez-Agirre, A., and Villegas, M. (2022). MarIA: Spanish Language Models. *Procesamiento del Lenguaje Natural*, page 39–60.

Hromei, C. D., Croce, D., Basile, V., and Basili, R. (2023). ExtremITA at EVALITA 2023: Multi-Task Sustainable Scaling to Large Language Models at its Extreme. In *CEUR Workshop Proceedings*, volume 3473.

Hu, E., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2022). LoRA: Low-Rank Adaptation of Large Language Models. In *Proc. of ICLR Conf.*

Kitchenham, B. and Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information and Software Technology*, 55(12):2049–2075.

Le Scao, T., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., et al. (2022). BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *Computing Research Repository (CoRR)*.

Leong, M., Abdelhalim, A., Ha, J., Patterson, D., Pincus, G. L., Harris, A. B., Eichler, M., and Zhao, J. (2024). MetRoBERTa: Leveraging Traditional Customer Relationship Management Data to Develop a Transit-Topic-Aware Language Model. *Transportation Research Record*.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of the Annual Meeting of the Assoc. for Computational Linguistics*, pages 7871–7880. ACL.

Li, Y., Su, H., Shen, X., Li, W., Cao, Z., and Niu, S. (2017). DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset. *Computing Research Repository (CoRR)*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *Computing Research Repository (CoRR)*, abs/1907.11692.

Liu, Z., Zhou, Y., Zhu, Y., Lian, J., Li, C., Dou, Z., Lian, D., and Nie, J.-Y. (2024). Information Retrieval Meets Large Language Models. In *Proc. of the ACM Web Conf. (WWW Companion)*, pages 1586–1589.

Mahadevkar, S. V., Patil, S., Kotecha, K., Soong, L. W., and Choudhury, T. (2024). Exploring AI-driven approaches for unstructured document analysis and future horizons. *Journal of Big Data*, 11(1).

Mastropaolo, A., Scalabrino, S., Cooper, N., Nader Palacio, D., Poshyvanyk, D., Oliveto, R., and Bavota, G. (2021). Studying the usage of text-to-text transfer transformer to support code-related tasks. In *Proc. of ICSE Conf.*, pages 336–347.

Nasution, A. H. and Onan, A. (2024). ChatGPT Label: Comparing the Quality of Human-Generated and LLM-Generated Annotations in Low-Resource Language NLP Tasks. *IEEE Access*, 12:71876–71900.

Peña, A., Morales, A., Fierrez, J., Serna, I., Ortega-Garcia, J., Puente, I., Córdova, J., and Córdova, G. (2023). Leveraging Large Language Models for Topic Classification in the Domain of Public Affairs. *Lecture Notes in Computer Science*, 14193 LNCS:20–33.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using siamese BERT-networks. *Computing Research Repository (CoRR)*.

Rodríguez-Cantelar, M., Estecha-Garitagoitia, M., D'Haro, L. F., Matía, F., and Córdoba, R. (2023). Automatic Detection of Inconsistencies and Hierarchical Topic Classification for Open-Domain Chatbots. *Applied Sciences (Switzerland)*, 13(16).

Russo, G., Stoehr, N., and Ribeiro, M. H. (2023). ACTI at EVALITA 2023: Automatic Conspiracy Theory Identification Task Overview. In *CEUR Workshop Proc.*, volume 3473. CEUR-WS.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *Computing Research Repository (CoRR)*.

Schabus, D., Skowron, M., and Trapp, M. (2017). One Million Posts: A Data Set of German Online Discussions. In *Proc. of the 40th SIGIR Conf.*, page 1241–1244. ACM.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. of EMNLP*, pages 1631–1642.

Stahlschmidt, S. and Stephen, D. (2020). Comparison of Web of Science, Scopus and Dimensions databases. Technical report, KB forschungspoolprojekt, DZHW Hannover, Germany.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. *Computing Research Repository (CoRR)*.

Trust, P. and Minghim, R. (2023). Query-Focused Submodular Demonstration Selection for In-Context Learning in Large Language Models. In *Proc. of the 31st Irish AICS Conf.*

Van Nooten, J., Kosar, A., De Pauw, G., and Daelemans, W. (2024). Advancing CSR Theme and Topic Classification: LLMs and Training Enhancement Insights. In *Proc. of FinNLP-KDF-ECONLP@LREC-COLING*, pages 292–305.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 2017-December, pages 5999–6009.

Williams, A., Nangia, N., and Bowman, S. (2018). A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In Walker, M., Ji, H., and Stent, A., editors, *Proc. of the Conf. of the North American Chapter of the Assoc. for Computational Linguistics: Human Language Technologies*, volume 1, pages 1112–1122. ACL.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proc. of the 18th EASE Conf.* ACM.

Yu, P., Xu, H., Hu, X., and Deng, C. (2023). Leveraging Generative AI and Large Language Models: A Comprehensive Roadmap for Healthcare Integration. *Healthcare (Switzerland)*, 11(20).

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. (2023). Opt: Open pre-trained transformer language models, 2022. *Computing Research Repository (CoRR)*, 3:19–0.

Zhang, X., Zhao, J., and Lecun, Y. (2015). Character-level convolutional networks for text classification. *Computing Research Repository (CoRR)*.

Enhancing Answer Attribution for Faithful Text Generation with Large Language Models

Juraj Vladika[a], Luca Mülln and Florian Matthes[b]
Technical University of Munich, School of Computation, Information and Technology, Department of Computer Science, Germany
{*juraj.vladika, luca.muelln, matthes*}*@tum.de*

Keywords: Natural Language Processing, Large Language Models, Information Retrieval, Question Answering, Answer Attribution, Text Generation, Interpretability.

Abstract: The increasing popularity of Large Language Models (LLMs) in recent years has changed the way users interact with and pose questions to AI-based conversational systems. An essential aspect for increasing the trustworthiness of generated LLM answers is the ability to trace the individual claims from responses back to relevant sources that support them, the process known as *answer attribution*. While recent work has started exploring the task of answer attribution in LLMs, some challenges still remain. In this work, we first perform a case study analyzing the effectiveness of existing answer attribution methods, with a focus on subtasks of answer segmentation and evidence retrieval. Based on the observed shortcomings, we propose new methods for producing more independent and contextualized claims for better retrieval and attribution. The new methods are evaluated and shown to improve the performance of answer attribution components. We end with a discussion and outline of future directions for the task.

1 INTRODUCTION

As Large Language Models (LLMs) rise in popularity and increase their capabilities for various applications, the way users access and search for information is noticeably changing (Kaddour et al., 2023). The impressive ability of LLMs to produce human-sounding text has led to new applications but also raised concerns. They sometimes generate responses that sound convincing but lack accuracy or credible sources, so-called hallucinations (Ji et al., 2023). This poses challenges to their reliability, especially in critical applications like law or healthcare, as well as in day-to-day usage (Wang et al., 2024a).

Additionally, the opaque nature of these models complicates understanding their decision-making processes and interpretability of generated outputs (Singh et al., 2024). As these models continue to permeate various sectors, from education (Kasneci et al., 2023) to healthcare (Nori et al., 2023) — the need for verifiable and accountable information becomes increasingly crucial. If LLMs provide incorrect information or biased content, the inability to trace back the origin of such responses can lead to misinformation and potential harm or infringe on copyrighted material (Lewis, 2023).

A promising avenue for increasing the trustworthiness and transparency of LLM responses is the idea of *answer attribution*. It refers to the process of tracing back (”attributing”) the claims from the output to external evidence sources and showing them to users (Rashkin et al., 2023). Distinct from usual methods of hallucination mitigation, which focus on altering the model’s output, answer attribution is oriented towards end users. It aims to equip users with a list of potential sources that support the output of the LLM to increase its transparency and leaves quality assurance to the users. This process usually involves segmenting LLM answers into claims and linking them to relevant evidence. While many attribution systems have started emerging in recent years (Li et al., 2023), we observe they still suffer from drawbacks limiting their applicability. The retrieved sources for specific claims and their respective entailment can be inaccurate due to inadequate claim formulation (Liu et al., 2023; Min et al., 2023).

To address these research gaps, in this study, we provide incremental contributions to the answer attribution process by enhancing its components. We: (1) perform a case study of current answer attribution components from literature and detect their shortcomings;

[a] https://orcid.org/0000-0002-4941-9166
[b] https://orcid.org/0000-0002-6667-5452

Vladika, J., Mülln, L. and Matthes, F.
Enhancing Answer Attribution for Faithful Text Generation with Large Language Models.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 147-158
ISBN: 978-989-758-716-0; ISSN: 2184-3228

(2) propose improvements to the answer-segmentation and evidence-retrieval components; and (3) provide a numerical and qualitative analysis of improvements. We involve human annotation on subsets when possible and consider multiple competing approaches. Our research builds on top of recent LLM factuality and answer attribution works and outlines open challenges, leaving the door open for further advancements and refinement of the process.

2 RELATED WORK

A lot of ongoing NLP work is devoted to ensuring the trustworthiness of LLMs in their everyday use (Liu et al., 2024), including their reliability (Zhang et al., 2023a), safety (Wei et al., 2024), fairness (Li et al., 2023), efficiency (Afzal. et al., 2023), or explainability (Zhao et al., 2024a). An important aspect hindering the trust in LLMs are hallucinations – described as model outputs that are not factual, faithful to the provided source content, or overall nonsensical (Ji et al., 2023).

A recent survey by (Zhang et al., 2023b) divides hallucinations into input-conflicting, context-conflicting, and fact-conflicting. Our work focuses on fact-conflicting, which are hallucinations in which facts in output contradict the world knowledge. Detecting hallucinations is tied to the general problem of measuring the factuality of model output (Augenstein et al., 2023; Zhao et al., 2024b) and automated fact-checking of uncertain claims (Guo et al., 2022; Vladika and Matthes, 2023). The recently popular method FactScore evaluates factuality by assessing how many atomic claims from a model output are supported by an external knowledge source (Min et al., 2023). Hallucinations can be corrected in the LLM output by automatically rewriting those claims found to be contradicting a trusted source, as seen in recent CoVe (Dhuliawala et al., 2023) or Factcheck-Bench (Wang et al., 2024b).

A middle ground between pure factuality evaluation and fact correction is answer attribution. The primary purpose of answer attribution is to enable users to validate the claims made by the model, promoting the generation of text that closely aligns with the cited sources to enhance accuracy (Li et al., 2023). One task setting is evaluating whether the LLMs can cite the references for answers from their own memory (Bohnet et al., 2023). A more common setup involves retrieving the references either before the answer generation or after generating it (Malaviya et al., 2024). When attributing claims to scientific sources, the more recent and better-cited publications were found to be the most trustworthy evidence (Vladika and Matthes, 2024). Some approaches to the problem include fine-tuning smaller LMs on NLP datasets (Yue et al., 2023) or using human-in-the-loop methods (Kamalloo et al., 2023). Our work builds on top of (Malaviya et al., 2024) by utilizing their dataset but improves the individual components of the attribution pipeline.

3 FOUNDATIONS

We provide a precise description for the task of attribution in the context of LLMs for this work as follows: **Answer Attribution** is the task of providing a set of sources s that inform the output response r of a language model for a given query q. These sources must be relevant to the model's response and should contain information that substantiates the respective sections of the response. This definition provides a comprehensive overview of the task and encapsulates its constituent subtasks:

1. **Response Segmentation.** Segmenting the response r into individual claims c_i.
2. **Claim Relevance Determination.** Determining the relevance of each claim c_i for the need of attribution ("claim check-worthiness").
3. **Finding Relevant Evidence.** Retrieving a list of relevant evidence sources s_i for each claim c_i.
4. **Evidence-Claim Relation.** Determining whether the evidence sources from the list of sources s_i actually refer to the claim c_i.

In our work, we focus on analyzing and improving subtasks 1 and 4, and to a lesser extent, subtasks 2 and 3, leaving further improvements to future work. We take the recent dataset *ExpertQA* (Malaviya et al., 2024) as a starting point for our study. Moving away from short factoid questions, this dataset emulates how domain experts in various fields interact with LLMs. Thus, the questions posed to the model are longer and more complex, can contain hypothetical scenarios, and elicit long, descriptive responses. This makes it a realistic benchmark for modern human-LLM interaction.

We take the responses generated by GPT-4 ("gpt-4" in OpenAI API) from ExpertQA and perform attribution evaluation based on claims found in its responses. Two main setups for attribution are post-hoc retrieval (PHR), which first generates the response and then does retrieval to attribute the facts; and retrieve-then-read (RTR), which first retrieves the sources and then generates the response (i.e., RAG). In our work, we focus on the PHR system (Fig. 1, because it is closer to the definition of attribution. Still, the challenges in claim formulation and evidence retrieval apply to both settings, so our findings also hold for RTR.

Table 1: High-level comparison of the different answer segmentation systems.

Segmentation System	Number of c	Unique $\#c$	avg. len(c)	c / Sentence
spaCy_sentences	938	855	103.2	1.00
gpt35_factscore	3016	2684	61.4	3.2
segment5_propsegment	2676	2232	54.2	2.85

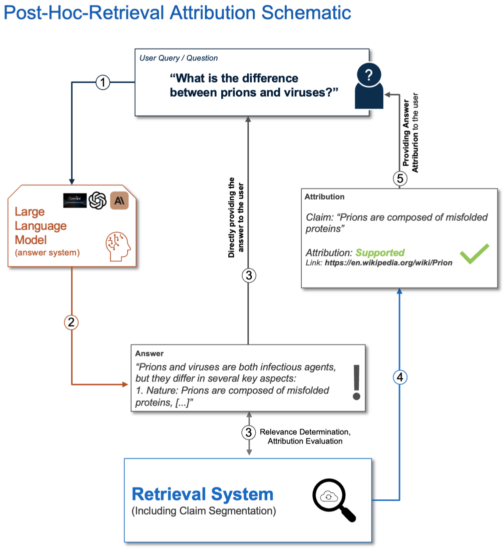

Post-Hoc-Retrieval Attribution Schematic
User Query / Question
"What is the difference between prions and viruses?"
1
Large Language Model (answer system)
Gemini
2
Directly providing the answer to the user
3
Providing Answer Attriburion to the user
5
Attribution
Claim: "Prions are composed of misfolded proteins"
Attribution: Supported
Link: https://en.wikipedia.org/wiki/Prion
Answer
"Prions and viruses are both infectious agents, but they differ in several key aspects:
1. Nature: Prions are composed of misfolded proteins, [...]"
4
3 Relevance Determination, Attribution Evaluation
Retrieval System
(Including Claim Segmentation)

Figure 1: The complete answer attribution process (in the Post-Hoc-Retrieval setup).

4 CASE STUDY OF EXISTING SOLUTIONS

This section provides a case study of recently popular approaches for different components of the answer attribution pipeline.

4.1 Answer Segmentation

As described above, the first step for attribution in PHR systems is to segment the provided LLM response into claims (atomic facts). We define a claim as ”a statement or a group of statements that can be attributed to a source”. The claim is either a word-by-word segment of the generated answer or semantically entailed by the answer. To validate the segmentation, we sample 20 random questions from the ExpertQA dataset. Three different segmentation systems are evaluated based on the number of atomic facts each claim contains and the number of claims they generate.

The first (**i**) and most intuitive way of segmenting an answer into claims is to use the syntactic structure of the answer, segmenting it into sentences, paragraphs, or other syntactic units. Following ExpertQA (Malaviya et al., 2024), this segmentation is done using the sentence tokenizer from the Python library `spaCy`.[1] The second approach (**ii**) for answer segmentation that we analyze is based on the work of PropSegment (Chen et al., 2023a), where text is segmented into *propositions*. A proposition is defined as a unique subset of tokens from the original sentence. We use the best-performing model from the paper, SegmenT5-large (Chen et al., 2023b), a fine-tuned version of the T5 checkpoint 1.1 (Chung et al., 2022). The third approach (**iii**) of segmenting an answer into claims utilizes pre-trained LLMs and prompting, as found in FactScore (Min et al., 2023). In their approach, the model is prompted to segment the answer into claims, and the resulting output is subsequently revised by human annotators. We replicate this method by using GPT-3.5 (turbo-0613) and the same prompt (*”Please breakdown the following sentence into independent facts:”*), amended with meta-information and instructions for the model on formatting the output. The prompt is in Appendix 7, Table 10.

Table 1 shows the differences between the three answer segmentation approaches. As expected, the average number of characters of the atomic facts created by GPT-3.5 and T5 is significantly smaller than the original sentence length. It is also noteworthy that the claims generated by GPT-3.5 are longer in characters and more numerous per sentence. In addition, the number of unique claims per answer and the number of claims per answer differ significantly by an average of 12% and up to 16.5% for SegmenT5. An error we observed is that the segmentation systems create duplicated claims for the same answer.

For a qualitative analysis of these segmented claims, we manually annotate 122 claims that the three systems generated for a randomly selected question *”A 55 year old male patient describes the sudden appearance of a slight tremor and having noticed his handwriting getting smaller, what are the possible ways you’d find a diagnosis?”*. The categories for annotations are aligned with (Chen et al., 2023a) and (Malaviya et al., 2024), and describe important claim properties. The properties are as follows: (1) **Atomic**: the claim contains a single atomic fact; (2) **Independent**: the claim can be verified without additional context; (3) **Useful**: the claim is useful for the question; (4) **Errorless**: the claim does not contain structural

[1] https://spacy.io/

errors, e.g., being an empty string; (5) **Repetition**: the claim is a repetition of another claim from the same segmentation system. Each category is binary, meaning a claim can be annotated with multiple categories. Given that the question is from the medical domain, the claims are expected to be more complex and require domain knowledge.

Table 2 shows the result of the qualitative analysis. The most noticeable outcome is that the `spaCy` segmentation system performs significantly differently compared to other systems. It simply tokenizes the responses into sentences and considers every sentence to be a claim, which is not realistic given the often quite long sentences generated by LLMs. Consequently, the score for ”Atomic” claims stands at 20% (3/15). Intriguingly, only 20% (3/15) of the sentences from the response are independently verifiable without additional context from the question or the rest of the response. Due to the complexity of the answer, most sentences reference a preceding sentence in the response, mentioning ”the patient” or ”the symptoms”.

The usefulness of the claims in answering the given questions is relatively high for spaCy sentence segmentation and GPT-3.5 segmentation but diminishes for the SegmenT5 segmentation. Although most claims are errorless, it is notable that all systems produce erroneous outputs. Specifically, for this question, `spaCy` segments four empty strings as individual sentences. It is plausible that errors in the other two segmentation systems stem from this issue, as they also rely on `spaCy`-tokenized sentences as input. This dependency also results in repetitions, primarily based on incorrect answer segmentation. This list provides a positive and a negative example claim for each category to give an idea of errors:

1. **Atomic** — *Positive:* ”Seeking a second opinion helps” (`gpt35_factscore`) – *Negative:* ”Brain tumors or structural abnormalities are among the possible causes that these tests aim to rule out.” (`gpt35_factscore`)
2. **Independent** — *Positive*: ”Parkinson’s disease is a cause of changes in handwriting.” (`segment5_propsegment`) – *Negative*: ”Imaging tests may be ordered.” (`segment5_propsegment`)
3. **Useful** — *Positive*: ”There are several possible diagnoses that could explain the sudden appearance of a slight tremor and smaller handwriting.” (`gpt35_factscore`) – *Negative*: ”The patient is a 55-year-old male.” (`segment5_propsegment`)
4. **Errorless** — *Positive*: ”The patient is experiencing smaller handwriting.” (`gpt35_factscore`) – *Negative*: ”The sentence is about something.” (`segment5_propsegment`)

Based on these findings, we conclude that automatic answer segmentation faces three main challenges and we give three desiderata for successful answer segmentation: (1) To provide independently verifiable claims, the segmentation system requires more context than just the sentence, possibly the entire paragraph and the question; (2) the segmentation system needs to be capable of handling domain-specific language, such as the complex medical domain; (3) if the goal is to identify individual atomic facts, the segmentation system needs to operate at a more granular level than sentences.

4.2 Claim Relevance

The relevance (usefulness) of a claim is evaluated based on its relation to the question. We define it as: *Given a question or query q and a corresponding answer a, a claim c with $c \in a$ is relevant if it provides information to satisfy the user’s information need.* Most attribution publications do not perform the relevance evaluation automatically, relying instead on annotators (Min et al., 2023). Due to limited resources, we want to investigate whether this can be performed automatically. We adopt the approach of FactCheck-Bench (Wang et al., 2024b), who implement it with a GPT-3.5 prompt – the prompt is in Appendix 7, Table 10. They classify a claim into four classes of ”check-worthiness”: factual claim, opinion, not a claim (e.g., *Is there something else you would like to know?*), and others (e.g., *As a language model, I cannot access personal information*).

To evaluate the performance, we use the same 122 claims from Table 2 and annotate with the LLM and manually. The agreement for ”factual claim” class is very high (79 annotations the same out of 85), while the biggest confusion is between ”not a claim” and ”other”. This shows that automatic assessment can reliably be used to determine the claim relevance. Therefore, we apply the prompt to automatically label all the claims from Table 1. The results are shown in Table 3. We observe that 86.3% claims generated by GPT3.5 FactScore system are factual. These 2,317 claims will be used in further steps for attribution evaluation.

4.3 Evidence Retrieval

The evidence retrieval step in the attribution process is arguably the most important, especially in a post-hoc retrieval system – its goal is to find the evidence to which a claim can be attributed to. Evidence sources can be generated directly from LLM’s memory (Ziems et al., 2023), retrieved from a static trusted corpus like Sphere (Piktus et al., 2021) or Wikipedia (Peng et al., 2023), or dynamically queried from Google (Gao et al., 2023). We use the Google approach: we take each claim (labeled as unique and factual in the

Table 2: Comparison of different claim properties for the different segmentation systems. The fractions show the number of occurrences divided by the total number of atomic claims generated by that system.

	Atomic	Independent	Useful	Errorless	Repetition
gpt35_factscore	53/56	8/56	44/56	48/56	13/56
segment5_propsegment	40/53	6/53	28/53	34/53	18/53
spaCy_sentences	3/15	3/15	10/15	11/15	3/15

Table 3: Claim relevance distribution of different segmentation systems.

Segmentation System	Unique # c	# factual	# not a claim	# opinion	# other
spaCy_sentences	855	550	244	26	35
gpt35_factscore	2684	2317	258	68	41
segment5_propsegment	2232	1878	290	36	28

previous steps) and query Google with it, take the top 3 results, scrape their entire textual content from HTML, and split it (with *NLTK*) into chunks of either 256 or 512 character length. We embed each chunk with a Sentence-BERT embedder *all-mpnet-base-v2* (Song et al., 2020) and store the chunks into a FAISS-vector database (Douze et al., 2024). After that, we query each claim against the vector store for that question and retrieve the top 5 most similar chunks.

Table 4: NLI predictions between a claim and its respective evidence snippets found on Google.

Method & CW	Contr.	Entail.	No Relation
GPT3.5 - 256	2	111	82 (36.0%)
GPT3.5 - 512	1	126	88 (38.6%)
DeBERTa - 256	12	37	179 (78.5%)
DeBERTa - 512	11	64	153 (67.1%)
Human - 256	8	54	166 (72.8%)
Human - 512	9	81	138 (60.5%)

We want to automatically determine whether the retrieved evidence chunk is related to the claim. We model this as a Natural Language Inference (NLI) task, following the idea from SimCSE (Gao et al., 2021), where two pieces of text are semantically related if there is an entailment or contradiction relation between them and unrelated otherwise. For this purpose, we use GPT-3.5 with a few-shot prompt (Appendix 7, Table 11) and DeBERTa-v3-large model fine-tuned on multiple NLI datasets from (Laurer et al., 2024), since DeBERTa was shown to be the most powerful encoder-only model for NLI.

We take 228 claim-evidence pairs and annotate them both manually and automatically with the two models (GPT and DeBERTa). The results are in Table 4. The results show that the DeBERTa-NLI model was by far more correlated with human judgment and that GPT-3.5 was overconfident in predicting the entailment relation, i.e., classifying a lot of irrelevant chunks as relevant. Additionally, the longer context window led to these longer evidence chunks being more related to the claim. The stricter nature of DeBERTa predictions makes it better suited for claim-evidence relation prediction. Therefore, we will use DeBERTa as the main NLI model in the next section, with a 512-character context window.

5 DEVELOPING SOLUTIONS

In this section, we propose certain solutions for selected key issues identified in the previous section. We use the existing answer attribution pipeline and enhance individual components to assess their effects on the overall system.

5.1 Answer Segmentation

One of the primary reasons for the weak performance of previous systems was the lack of independence among claims. Even when tasked to create atomic claims, most existing systems fail to provide sufficient context, making it difficult for the claims to stand alone. This leads to significant error propagation and misleading outcomes in evidence retrieval and attribution evaluation. There are three different types of claims produced by current systems that require additional context for accurate evaluation:

1. **Anaphoric References (Coreference Resolution).** Claims that include one or more anaphors referring to previously mentioned entities or concepts. — **Example:** "The purpose of *these strategies* is to reduce energy consumption.", "*They* ensure the well-being of everyone."
2. **Conditioning (Detailed Contextualization).** Claims that lack entire sentences or conditions necessary for proper contextualization. While not always obvious from the claim itself, this information is crucial for accurately evaluating the claims. — **Example:** "Chemotherapy is no longer the recommended course of action."
3. **Answer Extracts (Hypothetical Setup).** Claims that arise from questions describing a hypothetical scenario. Current answer systems often replicate parts of the scenario in the answer, leading to claims that cannot be evaluated independently of the scenario itself. — **Example:** "A young girl is running in front of cars."

We propose two strategies to provide more context during answer segmentation: (1) claim enrichment, and (2) direct segmentation with context. In the first approach, we edit extracted claims to incorporate the necessary context from both the answer and the question. A system employing this strategy would implement the function $f_{\text{enrich}}(q, r, c_{\text{non-independent}})$, where $c_{\text{non-independent}}$ is the non-independent claim, r is the response, and q is the question. In the second approach, we suggest a system that directly segments the answer into multiple independent claims, each supplemented with the required context. This system would use the function $f_{\text{segment}}(r, q)$, differing from the initial systems (as in Section 4), by incorporating the entire answer and question rather than basing the segmentation on individual sentences.

5.1.1 Claim Enrichment

We want to enrich only the non-independent claims. In the previous section, we manually labeled the claims for independence (Table 2). We now want to automate this task. For this purpose, we test whether the GPT-3.5 (turbo-0613) and GPT-4 (turbo-1106) systems can perform this task with a one-shot prompt (in Appendix 7, Table 13) that assesses the independence. The results are compared with human evaluation from Table 2. Table 5 shows the results. It is evident that both GPT-3.5 and GPT-4 exhibit significantly high precision, with GPT-4 outperforming in terms of recall and F1 score. We conclude that claim independence can be detected by LLMs (0.84 F1 in GPT4) and utilize the claims classified as ”non-independent” by GPT-4 to assess the performance of the function $f_{\text{enrich}}(q, r, c_{\text{non-independent}})$.

Table 5: Non-Independence detection performance compared to human evaluation.

	GPT3.5			GPT4		
System	Prec.	R	F1	Prec.	Rec.	F1
Overall	0.94	0.27	0.42	**0.96**	**0.74**	**0.84**
factscore	0.93	0.29	0.44	0.95	0.75	0.84
segmenT5	0.90	0.19	0.32	0.96	0.66	0.78
spaCy	1.0	0.5	0.67	1.0	1.0	1.0

To test the enrichment, we utilize only the GPT-3.5 system, as described in Table 3. From the 2,317 unique and factual claims, as segmented by the original GPT-3.5 system, we take a random sample of 500 and assess their independence using the GPT-4 prompt from the previous step. We observe **290 out of 500** were deemed to be ”not independent” by GPT-4. We then perform the enrichment by applying a one-shot prompt with both GPT-3.5 and GPT-4 to implement the function $f_{\text{enrich}}(q, a, c_{\text{non-independent}})$ and compare the results to the original claims. The comparison is conducted using the non-independence detection system described above. The quality of this step is measured in the reduction of non-independent claims. The results are presented in Figure 2.

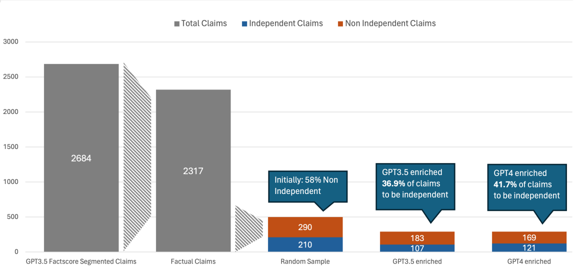


Figure 2: Statistics of contextualization of the 290 created claims by GPT3.5 and GPT4, evaluated by GPT4.

The enrichment function managed to make an additional 107/290 with GPT-3.5 and 121/290 with GPT-4 claims independent, i.e., further reducing the number of non-independent claims by 36.9% (GPT-3.5) and 41.7% (GPT-4). This is a considerable improvement that increases the number of claims usable for later attribution steps. Nevertheless, many claims still remained without context. Another observation is that the enrichment has noticeably increased the average number of characters of the claims. Initially, the average number of characters for independent claims was 66.0 and 59.4 for non-independent claims. The revision by GPT-4 increased it to 155.6 characters, and the enrichment by GPT-3.5 to 145.9 characters. Later, we evaluate the impact of claim enrichment on the evidence retrieval process (Section 5.3).

5.1.2 Answer Segmentation with Context – Direct Segmentation

An alternative to enrichment is directly segmenting the answer into multiple independent claims with context. This approach implements the function $f_{\text{segment}}(r, q)$ by using a one-shot prompt and GPT3.5 and GPT4 as LLMs. To evaluate the result quantitatively, we compare the average number of claims and the length of claims with those from alternative approaches to answer segmentation. This step is done on a subset of 100 question-answer pairs from ExpertQA. The prompt requests the model to print out a structured list of claims. The exact prompt can be found in Appendix 7, Table 14. The results are presented in Table 6.

Upon applying the segmentation to the responses from GPT-4, an increase in the number of claims was observed, aligning with the levels obtained through the original FactScore segmentation. This implementation aims to diminish non-independence, given that the original FactScore segmentation relied on SpaCy sen-

Table 6: Descriptive comparison of adopted answer segmentation approaches.

Segmentation System	Number of c	Unique # c	avg. len(c)	c / Sentence
GPT3.5 direct	644	644	102.8	0.75
GPT4 direct	948	948	84.1	1.11
spaCy_sentences	938	855	103.2	1.00
gpt35_factscore	3016	2684	61.4	3.22

Table 7: Comparison of claim enrichment on the retrieval performance.

Model	Contr.	Entail.	No Rel.
Original **Independent**	5.6%	42.2%	**52.2%**
Original **Non-Ind.**	3.6%	24.1%	**71.3%**
Enriched **Independent**	6.1%	35.4%	**58.6%**
Enriched **Non-Ind.**	1.3%	20.5%	**78.2%**

tences, which exhibited non-independence in 80% of instances. As generating independent claims from non-independent inputs is not possible, employing GPT-4 as a baseline may mitigate this issue.

5.2 Factuality & Independence

The next step in the evaluation involves analyzing the factuality of individual claims. This is done employing the same methodology as described in Section 4.2, with previous results in Table 3. The outcomes of the direct answer segmentation are depicted in Figure 3.

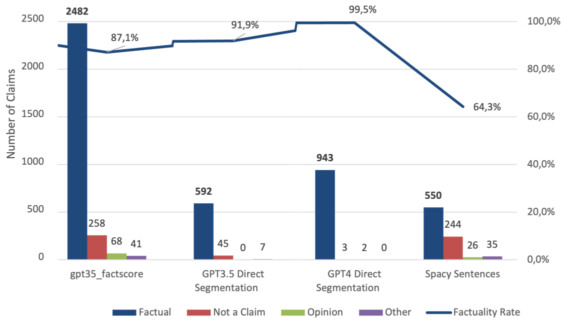


Figure 3: Visualization of the factuality evaluation statistics for the four different systems.

This figure clearly demonstrates an improvement in the factuality rate of the claims generated by both GPT-4 and GPT-3.5 compared to SpaCy sentence segmentation, with the factuality rate increasing from 64.3% to 99.5% for GPT-4 and to 91.9% for GPT-3.5. These results suggest that this approach is a significantly better alternative to spaCy tokenization.

5.3 Impact on Evidence Retrieval

The evaluation of the impact of claim enrichment on evidence retrieval is conducted using the same 2,317 (question, response, claim) triplets, which were classified by the GPT-3.5 system as factual, as in the previous setup. The retrieval process is conducted using the same GPT-3.5 enriched claim-based retrieval system For assessing the impact of claim enrichment on retrieval (function $f_{\text{enrich}}(q, a, c_{\text{non-independent}})$), we compare a sampled yet stratified set of claims across four categories: originally independent, originally non-independent, enriched (by GPT4) non-independent, and enriched (by GPT4) independent claims. The enriched claims are based on the originally non-independent claims. We utilize DeBERTa to evaluate the claim-evidence relation.

The findings are presented in Table 7. The table reveals several interesting findings: Firstly, it is evident that originally independent claims highly outperform originally non-independent claims in the evidence retrieval pipeline. Upon enriching the originally non-independent claims with GPT-4, as described in the previous section ($f_{\text{enrich}}(q, a, c_{\text{non-independent}})$), the claims that were successfully enriched show a big improvement in performance within the retrieval pipeline. This indicates that enriching (contextualizing) claims enhances the retrieval performance. The successfully enriched claims approach the performance of the originally independent claims, with a ”No Relation” share of 58.6%. However, claims that were not successfully enriched exhibit worse performance than the originally non-independent claims, with a ”No Relation” share of 78.2%. Overall, the effect of claim enrichment is a 16.2 percentage point reduction (69.9 to 53.7) of claim-source pairs with no relation.

Additionally, we evaluate the impact of direct answer segmentation on the retrieval process. For that, we use the random sample of 40 (question, response, claim) triplets per direct segmentation system, as described in Section 5.1.2. The results are presented in Table 8. As above, we analyze the share of (claim, evidence) pairs that are classified as ”Missing” or ”No Relation” by DeBERTa; a lower share means a better retrieval process. The table shows the claims were yet again enhanced when compared to the previous enrichment approach. Direct segmentation by GPT-4 records a combined ”Missing + No Relation” share of 48.5% for independent claims and 81.6% for non-independent claims. This represents a significant improvement for independent claims compared to both enriched and original claims.

Table 8: Comparison of direct answer segmentation on the retrieval performance (more Entailment is better).

Model	Contradiction	Entailment	Missing	No Relation
Original **Independent**	5.6%	42.2%	0.0%	**52.2%**
Original **Non-Ind.**	3.6%	24.1%	2.4%	**69.9%**
GPT3.5 Direct – **Independent**	4.2%	47.2%	0%	**48.6%**
GPT3.5 Direct – **Non-Ind.**	0%	27.0%	2.7%	**70.3%**
GPT4 Direct – **Independent**	0%	51.5%	0%	**48.5%**
GPT4 Direct – **Non-Ind.**	2.0%	14.3%	2.0%	**81.6%**

Table 9: Comparison of different embedding models and context window splitters on the retrieval performance (more Entailment indicates better performance).

Model	Contradiction	Entailment	No Relation
Ada 2.0	2.9%	41.0%	**56.0%**
AnglE	2.9%	39.5%	**57.5%**
SBert + Recursive CW	0.0%	22.1%	**76.1%**
SBert Baseline (Macro)	0.9%	35.7%	**62.5%**

To summarize the findings, it can be concluded that direct segmentation with context by GPT-4 significantly surpasses both the original and enriched claims and outperforms comparative methods in aspects of retrieval, time efficiency, and independent claim generation. It nearly matches the performance of GPT-4 in enriching non-independent claims regarding the creation of independent claims and surpasses it in the retrieval process at the macro level.

5.4 Analysis of Evidence Retrieval

As a final step, we briefly evaluate the evidence retrieval process itself, analyzing different embedding models and context window sizes. We utilize claims generated by GPT-4 Direct, as this system was shown to be the best performer in the previous steps. We use the same random sample of 40 questions. We modify two dimensions of the retrieval process: the embedding model and the context window splitter. Instead of Sentence-BERT, we employ OpenAI Ada 2.0, which provides embeddings from GPT-3.5, and AnglE-Embeddings (Li and Li, 2023) from a pre-trained sentence-transformer model optimized for retrieval. Rather than using a simple sliding window approach, we implement a recursive text splitter with overlap to capture more relevant information.

The search engine (Google Search Custom Search Engine) remains unchanged. The results are presented in Table 9. The results demonstrate that the Ada 2.0 Embeddings with the fixed 512c-size context window splitter outperform the overall SBert baseline, which was used in our previous experiments and depicted the best performance. The AnglE embeddings, optimized for retrieval, also outperform the Sentence-BERT baseline but fall behind the GPT-based Ada 2.0 embeddings. Interestingly, the recursive context window splitter with SBert embeddings performs significantly worse than the fixed context window splitter.

6 DISCUSSION

The evaluation of various attribution methods revealed that the main challenge lies in the precise retrieval of relevant evidence snippets, especially considering the complexity of the query or the intended user need. A crucial aspect of effective retrieval is in formulating claims for subsequent search in a way that they are atomic, independent, and properly contextualized. Additionally, addressing the shortcomings in answer segmentation and independence was essential for improving the attribution process. Segmenting answers into independent (contextualized) claims was most effectively done using GPT-4, yet it did not achieve an 80% success rate. This indicates that a general-purpose language model might not be the best choice for this task and could be improved in the future by a more specialized and smaller model tailored specifically for this purpose. Future work could involve fine-tuning models for detecting non-independent claims and exploring alternative approaches for source document retrieval. Additionally, future research should focus on expanding the scope of embedding models and their context windows for semantic search of evidence.

7 CONCLUSION

In this paper, we analyzed automated answer attribution, the task of tracing claims from generated LLM responses to relevant evidence sources. By splitting the task into constituent components of answer segmentation, claim relevance detection, and evidence retrieval, we performed a case analysis of current systems, determined their weaknesses, and proposed essential improvements to the pipeline. Our improvements led to an increase in performance in all three aspects of the answer attribution process. We hope our study will help future developments of this emerging NLP task.

REFERENCES

Afzal., A., Vladika., J., Braun., D., and Matthes., F. (2023). Challenges in domain-specific abstractive summarization and how to overcome them. In *Proceedings of the 15th International Conference on Agents and Artificial Intelligence - Volume 3: ICAART*, pages 682–689. INSTICC, SciTePress.

Augenstein, I., Baldwin, T., Cha, M., Chakraborty, T., Ciampaglia, G. L., Corney, D., DiResta, R., Ferrara, E., Hale, S., Halevy, A., Hovy, E., Ji, H., Menczer, F., Miguez, R., Nakov, P., Scheufele, D., Sharma, S., and Zagni, G. (2023). Factuality challenges in the era of large language models.

Bohnet, B., Tran, V. Q., Verga, P., Aharoni, R., Andor, D., Soares, L. B., Ciaramita, M., Eisenstein, J., Ganchev, K., Herzig, J., Hui, K., Kwiatkowski, T., Ma, J., Ni, J., Saralegui, L. S., Schuster, T., Cohen, W. W., Collins, M., Das, D., Metzler, D., Petrov, S., and Webster, K. (2023). Attributed question answering: Evaluation and modeling for attributed large language models.

Chen, S., Buthpitiya, S., Fabrikant, A., Roth, D., and Schuster, T. (2023a). PropSegmEnt: A large-scale corpus for proposition-level segmentation and entailment recognition. In *Findings of the Association for Computational Linguistics: ACL 2023*.

Chen, S., Zhang, H., Chen, T., Zhou, B., Yu, W., Yu, D., Peng, B., Wang, H., Roth, D., and Yu, D. (2023b). Sub-sentence encoder: Contrastive learning of propositional semantic representations. *arXiv preprint arXiv:2311.04335*.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2022). Scaling instruction-finetuned language models.

Dhuliawala, S., Komeili, M., Xu, J., Raileanu, R., Li, X., Celikyilmaz, A., and Weston, J. (2023). Chain-of-verification reduces hallucination in large language models.

Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., and Jégou, H. (2024). The faiss library.

Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., Zhao, V., Lao, N., Lee, H., Juan, D.-C., and Guu, K. (2023). RARR: Researching and revising what language models say, using language models. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508, Toronto, Canada. Association for Computational Linguistics.

Gao, T., Yao, X., and Chen, D. (2021). SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Guo, Z., Schlichtkrull, M., and Vlachos, A. (2022). A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., and McHardy, R. (2023). Challenges and applications of large language models.

Kamalloo, E., Jafari, A., Zhang, X., Thakur, N., and Lin, J. (2023). HAGRID: A human-llm collaborative dataset for generative information-seeking with attribution. *arXiv:2307.16883*.

Kasneci, E., Seßler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., et al. (2023). Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274.

Laurer, M., van Atteveldt, W., Casas, A., and Welbers, K. (2024). Less annotating, more classifying: Addressing the data scarcity issue of supervised machine learning with deep transfer learning and bert-nli. *Political Analysis*, 32(1):84–100.

Lewis, M. (2023). Generative artificial intelligence and copyright current issues. *Morgan Lewis LawFlash*.

Li, D., Sun, Z., Hu, X., Liu, Z., Chen, Z., Hu, B., Wu, A., and Zhang, M. (2023). A survey of large language models attribution.

Li, X. and Li, J. (2023). Angle-optimized text embeddings.

Liu, N., Zhang, T., and Liang, P. (2023). Evaluating verifiability in generative search engines. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7001–7025, Singapore. Association for Computational Linguistics.

Liu, Y., Yao, Y., Ton, J.-F., Zhang, X., Guo, R., Cheng, H., Klochkov, Y., Taufiq, M. F., and Li, H. (2024). Trustworthy llms: a survey and guideline for evaluating large language models' alignment.

Malaviya, C., Lee, S., Chen, S., Sieber, E., Yatskar, M., and Roth, D. (2024). ExpertQA: Expert-curated questions and attributed answers. In *2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Min, S., Krishna, K., Lyu, X., Lewis, M., Yih, W.-t., Koh, P., Iyyer, M., Zettlemoyer, L., and Hajishirzi, H. (2023). FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Bouamor, H., Pino, J., and Bali, K., editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore. Association for Computational Linguistics.

Nori, H., King, N., McKinney, S. M., Carignan, D., and Horvitz, E. (2023). Capabilities of gpt-4 on medical challenge problems.

Peng, B., Galley, M., He, P., Cheng, H., Xie, Y., Hu, Y., Huang, Q., Liden, L., Yu, Z., Chen, W., and Gao, J. (2023). Check your facts and try again: Improving large language models with external knowledge and automated feedback.

Piktus, A., Petroni, F., Karpukhin, V., Okhonko, D., Broscheit, S., Izacard, G., Lewis, P., Oğuz, B., Grave, E., Yih, W.-t., et al. (2021). The web is your oyster-knowledge-intensive nlp against a very large web corpus. *arXiv preprint arXiv:2112.09924*.

Rashkin, H., Nikolaev, V., Lamm, M., Aroyo, L., Collins, M., Das, D., Petrov, S., Tomar, G. S., Turc, I., and Reitter, D. (2023). Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840.

Singh, C., Inala, J. P., Galley, M., Caruana, R., and Gao, J. (2024). Rethinking interpretability in the era of large language models.

Song, K., Tan, X., Qin, T., Lu, J., and Liu, T.-Y. (2020). Mpnet: masked and permuted pre-training for language understanding. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Vladika, J. and Matthes, F. (2023). Scientific fact-checking: A survey of resources and approaches. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6215–6230, Toronto, Canada. Association for Computational Linguistics.

Vladika, J. and Matthes, F. (2024). Improving health question answering with reliable and time-aware evidence retrieval. In Duh, K., Gomez, H., and Bethard, S., editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4752–4763, Mexico City, Mexico. Association for Computational Linguistics.

Wang, B., Chen, W., Pei, H., Xie, C., Kang, M., Zhang, C., Xu, C., Xiong, Z., Dutta, R., Schaeffer, R., Truong, S. T., Arora, S., Mazeika, M., Hendrycks, D., Lin, Z., Cheng, Y., Koyejo, S., Song, D., and Li, B. (2024a). Decodingtrust: A comprehensive assessment of trustworthiness in gpt models.

Wang, Y., Reddy, R. G., Mujahid, Z. M., Arora, A., Rubashevskii, A., Geng, J., Afzal, O. M., Pan, L., Borenstein, N., Pillai, A., Augenstein, I., Gurevych, I., and Nakov, P. (2024b). Factcheck-bench: Fine-grained evaluation benchmark for automatic fact-checkers.

Wei, A., Haghtalab, N., and Steinhardt, J. (2024). Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems*, 36.

Yue, X., Wang, B., Chen, Z., Zhang, K., Su, Y., and Sun, H. (2023). Automatic evaluation of attribution by large language models. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4615–4635, Singapore. Association for Computational Linguistics.

Zhang, J., Bao, K., Zhang, Y., Wang, W., Feng, F., and He, X. (2023a). Is chatgpt fair for recommendation? evaluating fairness in large language model recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pages 993–999.

Zhang, Y., Li, Y., Cui, L., Cai, D., Liu, L., Fu, T., Huang, X., Zhao, E., Zhang, Y., Chen, Y., Wang, L., Luu, A. T., Bi, W., Shi, F., and Shi, S. (2023b). Siren's song in the ai ocean: A survey on hallucination in large language models.

Zhao, H., Chen, H., Yang, F., Liu, N., Deng, H., Cai, H., Wang, S., Yin, D., and Du, M. (2024a). Explainability for large language models: A survey. *ACM Transactions on Intelligent Systems and Technology*, 15(2):1–38.

Zhao, Y., Zhang, J., Chern, I., Gao, S., Liu, P., He, J., et al. (2024b). Felm: Benchmarking factuality evaluation of large language models. *Advances in Neural Information Processing Systems*, 36.

Ziems, N., Yu, W., Zhang, Z., and Jiang, M. (2023). Large language models are built-in autoregressive search engines. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2666–2678, Toronto, Canada. Association for Computational Linguistics.

APPENDIX

This is the appendix with additional material.

Technical Setup and Manual Annotation

All GPT 3.5 and GPT 4 models were accessed through the official OpenAI API. Version *turbo-0125* for GPT 3.5 and *0125-preview* for GPT 4, or as indicated in the text. For local experiments (such as model embeddings with sentence transformers, DeBERTa entailment prediction, etc.), one A100 GPU with 40GB of VRAM was used, for a duration of one computation hour per experiment. No fine-tuning was performed by us, models like SegmenT5 and DeBERTa-v3 were used out-of-the-box, found in cited sources and HuggingFace.

Whenever we refer to manual annotation of data examples, this was done by two paper authors, who have a master's degree in computer science and are pursuing a PhD degree in computer science. None of the annotations required in-depth domain knowledge and were mostly reading comprehension tasks.

Prompts

The used prompts are given in Tables 10–14.

Table 10: Overview of applied prompts for GPT answer segmentation and claim relevance (check-worthiness) detection.

Use Case	Prompt Content
FactScore Answer Segmentation with GPT 3.5	Please breakdown the following sentence into independent facts. Don’t provide meta-information about sentence or you as a system. Just list the facts and strictly stick to the following format: 1. ”Fact 1” 2. ”Fact 2” 3. ”...” The sentence is:
Claim Relevance / Check-Worthiness Detection	You are a factchecker assistant with task to identify a sentence, whether it is 1. a factual claim; 2. an opinion; 3. not a claim (like a question or a imperative sentence); 4. other categories. Let’s define a function named checkworthy(input: str). The return value should be a python int without any other words, representing index label, where index selects from [1, 2, 3, 4]. For example, if a user call checkworthy(”I think Apple is a good company.”) You should return 2 If a user call checkworthy(”Friends is a great TV series.”) You should return 1 If a user call checkworthy(”Are you sure Preslav is a professor in MBZUAI?”) You should return 3 If a user call checkworthy(”As a language model, I can’t provide these info.”) You should return 4 Note that your response will be passed to the python interpreter, no extra words. checkworthy(”input”)

Table 11: Overview of applied prompt for claim-evidence relation detection, i.e., entailment recognition (NLI) between the claim and retrieved evidence chunk with GPT 3.5.

Use Case	Prompt Content
Claim-Evidence Entailment Recognition	Your task is to determine if a claim is supported by a document given a specific question. Implement the function nli(question: str, claim: str, document: str) -> str which accepts a question, a claim, and a document as input. The function returns a string indicating the relationship between the claim and the document in the context of the question. The possible return values are: ”entailed” if the claim is supported by the document, ”contradicted” if the claim is refuted by the document, ”no_relation” if the claim has no relevant connection to the document given the question. Your evaluation should specifically consider the context provided by the question. The output should be a single string value without additional comments or context, as it will be used within a Python interpreter. Examples: Question: ”You are patrolling the local city center when you are informed by the public about a young girl behaving erratically near traffic. What are your initial thoughts and actions?” Claim: ”Trained professionals should handle situations like this.” Document: ”Every trained professional football player should be adept at managing high-stress situations on the field.” Output: ”no_relation” Question: ”You are patrolling the local city center when you are informed by the public about a young girl behaving erratically near traffic. What are your initial thoughts and actions?” Claim: ”Trained professionals should handle situations like this.” Document: ”Standard police officer training includes procedures for managing public disturbances and emergencies.” Output: ”entailed”

Table 12: Overview of applied prompt for the claim independence detection.

Use Case	Prompt Content
Claim Independence Detection	You are tasked with determining whether a given claim or statement can be verified independently. A claim is considered ”independent” if it contains sufficient information within itself to assess its truthfulness without needing additional context or external information. Your response must strictly be either ”independent” or ”not independent.” Adhere to this format precisely, as your output will be processed by a Python interpreter. Guidelines: Evaluate if the claim provides enough detail on its own to be verified. Do not consider external knowledge or context not present in the claim. Respond only with ”independent” if the claim is self-sufficient for verification; otherwise, respond with ”not independent.” The below examples contain Rationales for explanation, which are not allowed in your response. Examples: Input: ”The sun rises in the east.” Output: independent Input: ”Chemotherapy is no longer the recommended course of action.” Output: not independent Rationale: The claim would require additional context, for example the type of cancer or the patient’s medical history. Input: ”Opening up the aperture can overexpose the image slightly.” Output: independent Input: ”A young girl is running in front of cars.” Output: not independent Rationale: The claim is situational and lacks specific details that would allow for independent verification. Input: ”They ensure the well-being of everyone involved.” Output: not independent Rationale: The claim is vagues, it is not known who ”They” are.

Table 13: Overview of applied prompt for the claim enrichment process.

Use Case	Prompt Content
Claim Enrichment Prompt	Your task involves providing context to segmented claims that were originally part of a larger answer, making each claim verifiable independently. This involves adding necessary details to each claim so that it stands on its own without requiring additional information from the original answer. The claim should stay atomic and only contain one specific statement or piece of information. Do not add new information or more context than necessary! Ensure that all pronouns or references to specific situations or entities (e.g., ”He,” ”they,” ”the situation”) are clearly defined within the claim itself. Your output should consist solely of the context-enhanced claim, without any additional explanations, as it will be processed by a Python interpreter. Example: Question: ”How to track the interface between the two fluids?” Answer: ”To track the interface between two fluids, you can use various techniques depending on the specific situation and the properties of the fluids. Here are a few common methods: ... 4. Ultrasonic Techniques: Ultrasonic waves can be used to track the interface between fluids. By transmitting ultrasonic waves through one fluid and measuring the reflected waves, you can determine the position of the interface. ... It’s important to note that the choice of method depends on the specific application and the properties of the fluids involved.” Claim: ”Reflected waves can be measured.” Revised Claim: ”Reflected waves can be measured to determine the position of the interface between two fluids.”

Table 14: Overview of the prompt for direct claim segmentation with added context.

Use Case	Prompt Content
Direct Claim Segmentation with Context	Objective: Transform the answer to a question into its discrete, fundamental claims. Each claim must adhere to the following criteria: Conciseness: Formulate each claim as a brief, standalone sentence. Atomicity: Ensure that each claim represents a single fact or statement, requiring no further subdivision for evaluation of its truthfulness. Note that most listing and ”or-combined” claims are not atomic and must be split up. Independence: Craft each claim to be verifiable on its own, devoid of reliance on additional context or preceding information. For instance, ”The song was released in 2019” is insufficiently specific because the identity of ”the song” remains ambiguous. Make sure that there is no situational dependency in the claims. Consistency in Terminology: Utilize language and terms that reflect the original question or answer closely, maintaining the context and specificity. Non-reliance: Design each claim to be independent from other claims, eliminating sequential or logical dependencies between them. Exhaustiveness: Ensure that the claims cover all the relevant information in the answer, leaving no important details unaddressed. Strictly stick to the below output format, which numbers any claims and separates them by a new line. This is important, as the output will be passed to a python interpreter. Don’t add any explanation or commentary to the output. Example: Input: Question: As an officer with the NYPD, I am being attacked by hooligans. What charges can be pressed? Answer: If you’re an NYPD officer and you’re being assaulted by hooligans, you have the right to press charges for assault on a police officer, which is recognized as a criminal offense under New York law. Specifically, the act of assaulting a police officer is addressed under New York Penal Law § 120.08, designating it as a felony. Offenders may face severe penalties, including time in prison and monetary fines. Output: 1. An NYPD officer assaulted by hooligans has the right to press charges for assault on a police officer. 2. Assault on a police officer is deemed a criminal offense in New York. 3. The act of assaulting a police officer is specified under New York Penal Law § 120.08. 4. Under New York law, assaulting a police officer is categorized as a felony. 5. Conviction for assaulting a police officer in New York may result in imprisonment. 6. Conviction for assaulting a police officer in New York may lead to monetary fines.

MERGE App: A Prototype Software for Multi-User Emotion-Aware Music Management

Pedro Lima Louro[1][a], Guilherme Branco[1][b], Hugo Redinho[1][c], Ricardo Correia[1][d], Ricardo Malheiro[1,2][e], Renato Panda[1,3][f] and Rui Pedro Paiva[1][g]

[1] *University of Coimbra, Centre for Informatics and Systems of the University of Coimbra (CISUC), Department of Informatics Engineering, and LASI, Portugal*

[2] *Polytechnic Institute of Leiria School of Technology and Management, Portugal*

[3] *Ci2 — Smart Cities Research Center, Polytechnic Institute of Tomar, Portugal*

pedrolouro@dei.uc.pt, guilherme.m.branco@tecnico.ulisboa.pt, redinho@student.dei.uc.pt, {ricardocorreia, rsmal, panda, ruipedro}@dei.uc.pt

Keywords: Music Information Retrieval, Music Emotion Recognition, Machine Learning, Deep Learning, Software.

Abstract: We present a prototype software for multi-user music library management using the perceived emotional content of songs. The tool offers music playback features, song filtering by metadata, and automatic emotion prediction based on arousal and valence, with the possibility of personalizing the predictions by allowing each user to edit these values based on their own emotion assessment. This is an important feature for handling both classification errors and subjectivity issues, which are inherent aspects of emotion perception. A path-based playlist generation function is also implemented. A multi-modal audio-lyrics regression methodology is proposed for emotion prediction, with accompanying validation experiments on the MERGE dataset. The results obtained are promising, showing higher overall performance on train-validate-test splits (73.20% F1-score with the best dataset/split combination).

1 INTRODUCTION

The digital era has brought an unprecedented amount of music right at our fingertips through digital marketplaces and streaming services. With the sudden availability of millions of songs to users, the necessity to automatically organize and find relevant music emerged. Current recommendation systems provide personalized suggestions to users based on listening patterns and using tags, such as genre, style, etc. However, options are lacking when we consider recommendations based on the automatic analysis of the emotional content of songs.

The field of Music Emotion Recognition (MER) has seen considerable advances in recent years in terms of the more classical approaches. Panda et al. (2020) proposed a new set of features that considerably increased the performance of these systems, achieving a 76.4% F1-score with the top 100 ranked features. Although the feature evaluation is limited to one dataset, the improvements are significant compared to the best results from similar systems that reached a glass ceiling Hu et al. (2008).

One drawback of audio-only methodologies is their shortcomings when differentiating valence. Various systems have been proposed using a bimodal approach leveraging both audio and lyrics, attaining considerable improvements when compared to systems using only one or the other Delbouys et al. (2018); Pyrovolakis et al. (2022). Such systems have also implemented Deep Learning (DL) architectures to skip the time-consuming feature engineering and extraction steps from the classical systems and considerably speed up the inference process of the overall system.

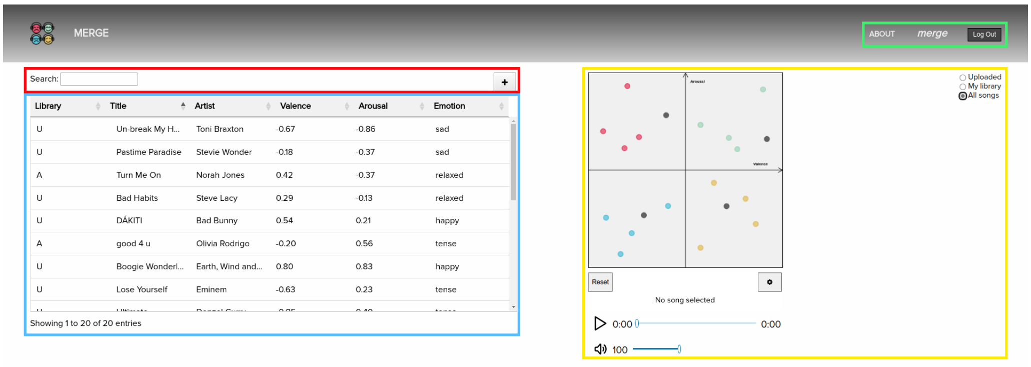
In this study, we present the MERGE[1] application,

[a] https://orcid.org/0000-0003-3201-6990
[b] https://orcid.org/0000-0003-4073-1716
[c] https://orcid.org/0009-0004-1547-2251
[d] https://orcid.org/0000-0001-5663-7228
[e] https://orcid.org/0000-0002-3010-2732
[f] https://orcid.org/0000-0003-2539-5590
[g] https://orcid.org/0000-0003-3215-3960

[1] MERGE is the acronym of ”Music Emotion Recognition nExt Generation”, a research project funded by the Portuguese Science Foundation.

Louro, P., Branco, G., Redinho, H., Correia, R., Malheiro, R., Panda, R. and Paiva, R.
MERGE App: A Prototype Software for Multi-User Emotion-Aware Music Management.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 159-166
ISBN: 978-989-758-716-0; ISSN: 2184-3228



Library	Title	Artist	Valence	Arousal	Emotion
U	Un-break My H...	Toni Braxton	-0.67	-0.86	sad
U	Pastime Paradise	Stevie Wonder	-0.18	-0.37	sad
A	Turn Me On	Norah Jones	0.42	-0.37	relaxed
U	Bad Habits	Steve Lacy	0.29	-0.13	relaxed
U	DÁKITI	Bad Bunny	0.54	0.21	happy
A	good 4 u	Olivia Rodrigo	-0.20	0.56	tense
U	Boogie Wonderl...	Earth, Wind and...	0.80	0.83	happy
U	Lose Yourself	Eminem	-0.63	0.23	tense

Showing 1 to 20 of 20 entries

Figure 1: MERGE application interface. The AV plot, alongside music playback and song display controls, is seen in yellow. The table view is highlighted in blue with red highlight filtering search bar and button for adding music. Finally, green highlights the buttons for application information and user logout.

which automatically predicts the arousal and valence of songs based on Russell's Circumplex Model Russell (1980). Two axes make up this model: arousal (Y-axis), which depicts whether the song has high or low energy, and valence (X-axis), which represents whether the emotion of the song has a negative or positive connotation.

The integrated model used for prediction is also presented in this study alongside validation experiments, which received both audio and lyrics information to map the song more accurately into the above mentioned model.

The MERGE application is a follow-up of the MOODetector application, previously created by our team Cardoso et al. (2011). The new MERGE app was built from scratch, with significant code refactoring and optimization, while keeping the overall user interface of the MOODetector app. In addition, significant novel features and improvements were implemented, namely: i) a bimodal app, which exploits the combination of audio and lyrics data for improved classification (unlike the single audio modality in the MOODetector app); ii) an improved classification model, training with the MERGE dataset Louro et al. (2024b) and following a deep learning approach Louro et al. (2024a); iii) and a shift from the monolithic single-user paradigm to the web-based multi-user paradigm.

2 MERGE APPLICATION

The MERGE application is implemented using JavaScript, with the addition of the jQuery library to handle AJAX, for its frontend, while the backend is served using the Express library on top of Node.js. The application interface is depicted in Figure 1.

2.1 Application Overview

The components can be broken down as: i) the Russell's Circumplex model where all songs can be seen (highlighted in yellow); ii) a table view of the songs with various options for sorting (highlighted in blue); iii) options to filter and add new songs (seen in the red region, from left to right); iv) information about the application, the current user, and an option to logout (highlighted in green, also from left to right).

Songs are placed in the plot described in i) according to the estimated arousal and valence (AV) values in an interval of [-1, 1] for each axis. The process for obtaining these values is described in Section 3. Beyond the AV positioning, each point on the plane is also color-coded depending on the quadrant: green (happy), red (tense), blue (sad), and yellow (relaxed).

The view of the graph can be switched between a) ”Uploaded” to show only songs uploaded by the current user, b) ”My Library” to display a user's library, i.e., the songs uploaded by the current user, plus songs uploaded by other users added by the current user, and c) ”All songs” to show all the songs available in the database. A note regarding the latter option is the differentiation of songs not added by the user, appearing as grey dots in the plot, also depicted in Figure 1.

Each user can change the song's position directly by moving the point in the plane view, or by editing the AV values through the table view. These new AV values are unique to the user.

The application can be used as an audio playback software, thus offering usual features such as:

- Playback controls for mp3 files (play, pause, seek);
- Volume controls, including mute;
- Double-clicking a song to be played either in the plot or table view;

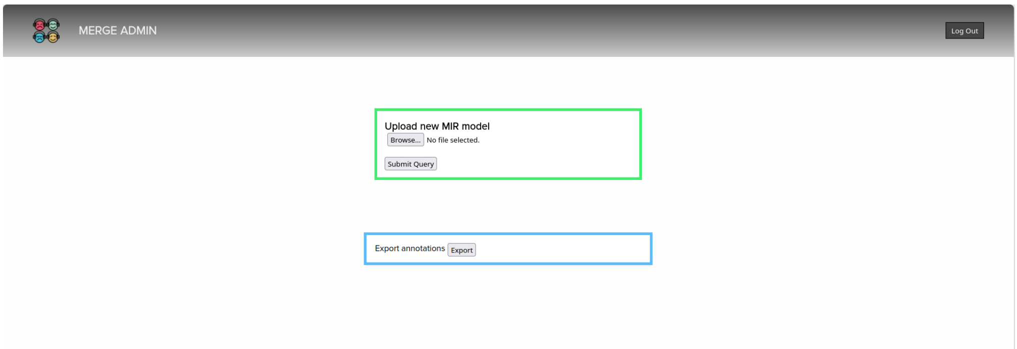
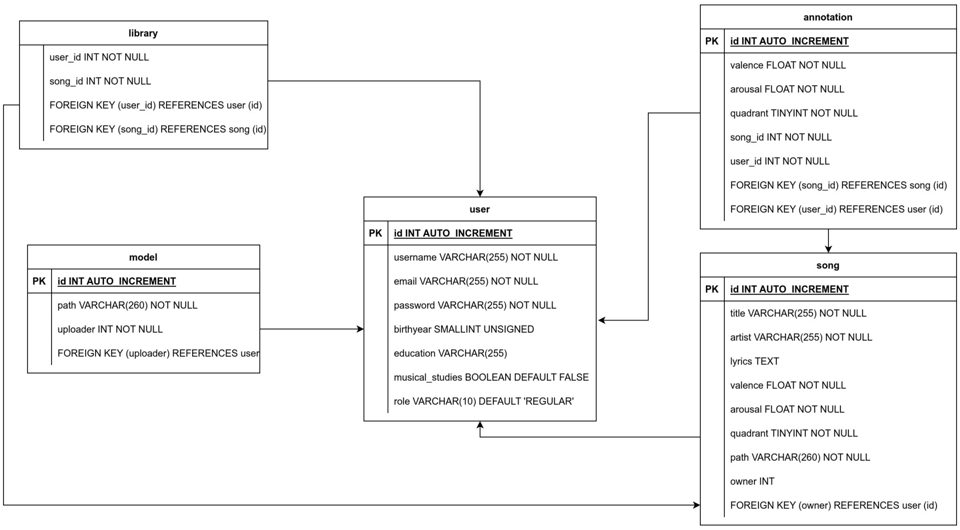


Figure 2: MERGE App Backoffice. Users with administrative privileges can overwrite the model used for AV values' prediction (highlighted in green) and export a CSV file with information regarding the annotations for all users to each song in the database (highlighted in blue).



library
user_id INT NOT NULL
song_id INT NOT NULL
FOREIGN KEY (user_id) REFERENCES user (id)
FOREIGN KEY (song_id) REFERENCES song (id)

model
PK id INT AUTO_INCREMENT
path VARCHAR(260) NOT NULL
uploader INT NOT NULL
FOREIGN KEY (uploader) REFERENCES user

user
PK id INT AUTO_INCREMENT
username VARCHAR(255) NOT NULL
email VARCHAR(255) NOT NULL
password VARCHAR(255) NOT NULL
birthyear SMALLINT UNSIGNED
education VARCHAR(255)
musical_studies BOOLEAN DEFAULT FALSE
role VARCHAR(10) DEFAULT 'REGULAR'

annotation
PK id INT AUTO_INCREMENT
valence FLOAT NOT NULL
arousal FLOAT NOT NULL
quadrant TINYINT NOT NULL
song_id INT NOT NULL
user_id INT NOT NULL
FOREIGN KEY (song_id) REFERENCES song (id)
FOREIGN KEY (user_id) REFERENCES user (id)

song
PK id INT AUTO_INCREMENT
title VARCHAR(255) NOT NULL
artist VARCHAR(255) NOT NULL
lyrics TEXT
valence FLOAT NOT NULL
arousal FLOAT NOT NULL
quadrant TINYINT NOT NULL
path VARCHAR(260) NOT NULL
owner INT
FOREIGN KEY (owner) REFERENCES user (id)

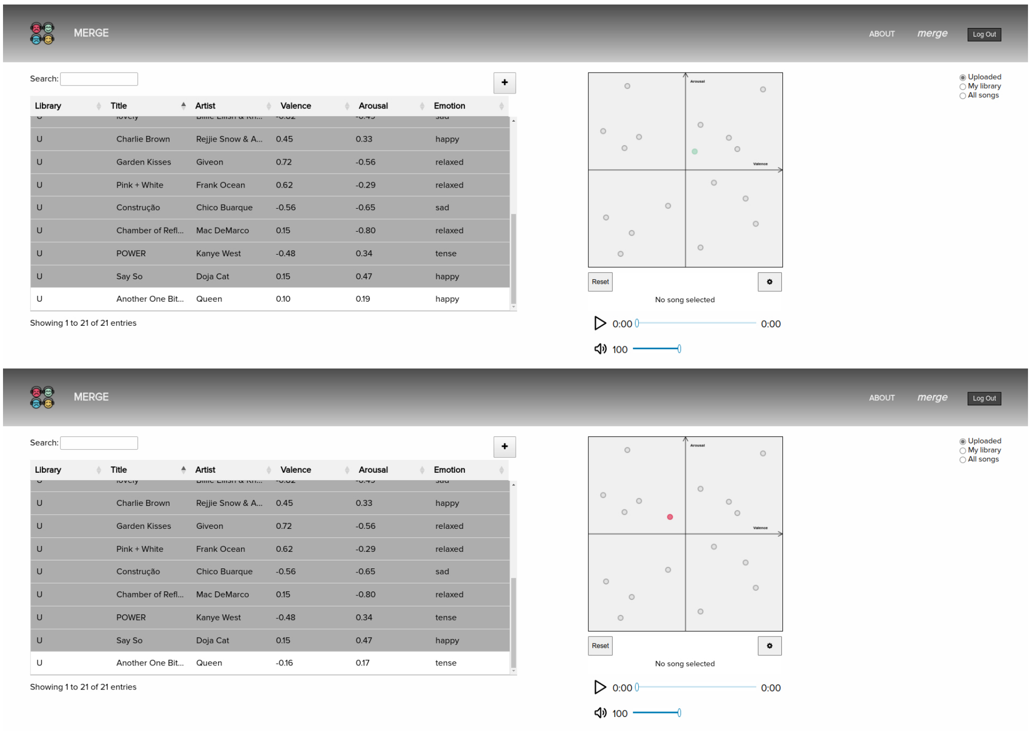
Figure 3: Entity relation diagram for the application's database.

- Filtering and sorting by any of the available song properties (title, artist, valence, arousal, emotion);
- Adding and deleting songs from the user's library.

The application also provides a backoffice for users with administrative privileges, pictured in Figure 2. After logging in, the user can perform one of two actions: upload and deploy a new model for AV prediction, and export a CSV with the existing user AV annotations. The latter option is designed to easily retrieve each user's available annotations for songs in their respective libraries. In this way, the MERGE app can be used as a crowdsourcing data collection and annotation tool, promoting the creation of sizeable and quality MER datasets, a current key need in MER research Panda et al. (2020).

Regarding the database used to store all the relevant data from users and songs, the corresponding entity relationship diagram is depicted in Figure 3. The user table stores the user's personal information, as well as the user role, for access privileges purposes. The path for the model used for AV prediction is saved in the corresponding table and identifies the user that uploaded the currently deployed model. The song table stores all song-related information, including metadata, the AV values first predicted by the presently deployed model, the mapped quadrant in the plot view, the path for the uploaded audio clip, and the reference to the user who first added the song.

The user-specific annotations are stored in the annotation table, which stores the AV values defined per the user's perception, the corresponding quadrant, and a reference to the song and user for that annotation. Finally, the library table stores all user libraries, containing only songs added by the user, either through uploading or from other users' libraries.



Library	Title	Artist	Valence	Arousal	Emotion
U	lovely	Billie Eilish & Kh...	-0.82	-0.49	sad
U	Charlie Brown	Rejjie Snow & A...	0.45	0.33	happy
U	Garden Kisses	Giveon	0.72	-0.56	relaxed
U	Pink + White	Frank Ocean	0.62	-0.29	relaxed
U	Construção	Chico Buarque	-0.56	-0.65	sad
U	Chamber of Refl...	Mac DeMarco	0.15	-0.80	relaxed
U	POWER	Kanye West	-0.48	0.34	tense
U	Say So	Doja Cat	0.15	0.47	happy
U	Another One Bit...	Queen	0.10	0.19	happy

Showing 1 to 21 of 21 entries

Library	Title	Artist	Valence	Arousal	Emotion
U	lovely	Billie Eilish & Kh...	-0.82	-0.49	sad
U	Charlie Brown	Rejjie Snow & A...	0.45	0.33	happy
U	Garden Kisses	Giveon	0.72	-0.56	relaxed
U	Pink + White	Frank Ocean	0.62	-0.29	relaxed
U	Construção	Chico Buarque	-0.56	-0.65	sad
U	Chamber of Refl...	Mac DeMarco	0.15	-0.80	relaxed
U	POWER	Kanye West	-0.48	0.34	tense
U	Say So	Doja Cat	0.15	0.47	happy
U	Another One Bit...	Queen	-0.16	0.17	tense

Showing 1 to 21 of 21 entries

Figure 4: At the top, ”Another One Bites The Dust” by Queen is added to the library and placed in the plane according to the predicted AV values. At the bottom, the point representing the song is moved to a more accurate position, according to the user.

2.2 Building an Emotionally-Aware Library

After adding a new song from an available MP3 file, AV values are automatically predicted, and a point is added to the plot alongside a new entry on the table. Should the user disagree with the predicted values, these can be easily changed by editing the entry from the table view or moving the point in the plot. An example of the initially predicted position for a newly added song and the final position after adjustment can be seen in Figure 4. This personalization mechanism provides users with the ability to address the intrinsic subjectivity in MER. However, tackling this issue continues to present a significant challenge.

2.3 Path-Based Playlist Generation

The ability to generate a playlist based on a user-drawn path is currently implemented, as depicted in Figure 5. This feature allows users to freely create an emotionally-varying playlist.

This is done by computing the distance of the user-defined *N* closest songs to the reference points that make up the drawn path. The user may also configure how far the songs can be from the path to be considered into the calculations. This threshold is defined as a decimal number between the plane interval ([-1, 1]).

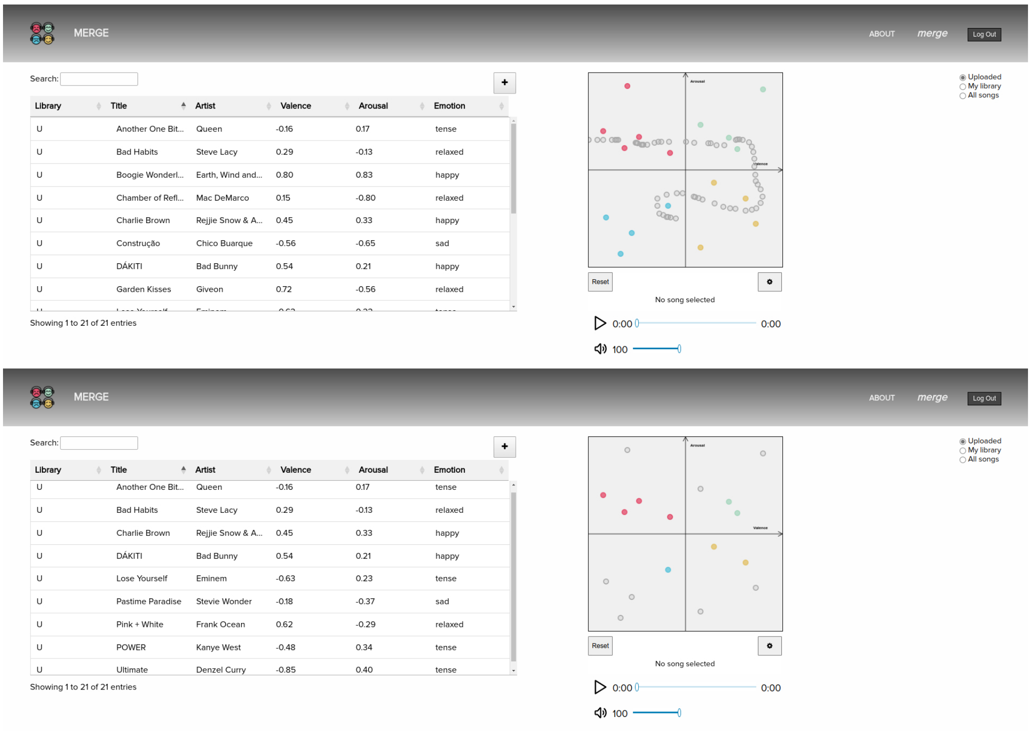
3 SONG EMOTION PREDICTION

In this section, we discuss the methodology used to predict AV values for a given song. First, the DL model's architecture is presented, followed by the pre-processing steps for each modality, a description of the optimization used, and the evaluation conducted.

3.1 Model Architecture

The proposed architecture, depicted in Figure 6, is based on the one by Delbouys et al. Delbouys et al. (2018). Distinct audio and lyrics branches receive Mel-spectrogram representations and word embeddings, respectively. The learned features for each modality are then fused and further processed by a small Dense Neural Network (DNN), finally outputting the AV values prediction.

We opted for a bimodal audio-lyrics approach considering that both modalities have relevant information for the different axes of Russell's Circumplex Model. Audio has been shown to better predict



Search:

Library	Title	Artist	Valence	Arousal	Emotion
U	Another One Bit...	Queen	-0.16	0.17	tense
U	Bad Habits	Steve Lacy	0.29	-0.13	relaxed
U	Boogie Wonderl...	Earth, Wind and...	0.80	0.83	happy
U	Chamber of Refl...	Mac DeMarco	0.15	-0.80	relaxed
U	Charlie Brown	Rejjie Snow & A...	0.45	0.33	happy
U	Construção	Chico Buarque	-0.56	-0.65	sad
U	DÁKITI	Bad Bunny	0.54	0.21	happy
U	Garden Kisses	Giveon	0.72	-0.56	relaxed
U	Lose Yourself	Eminem	-0.63	0.23	tense

Showing 1 to 21 of 21 entries

Search:

Library	Title	Artist	Valence	Arousal	Emotion
U	Another One Bit...	Queen	-0.16	0.17	tense
U	Bad Habits	Steve Lacy	0.29	-0.13	relaxed
U	Charlie Brown	Rejjie Snow & A...	0.45	0.33	happy
U	DÁKITI	Bad Bunny	0.54	0.21	happy
U	Lose Yourself	Eminem	-0.63	0.23	tense
U	Pastime Paradise	Stevie Wonder	-0.18	-0.37	sad
U	Pink + White	Frank Ocean	0.62	-0.29	relaxed
U	POWER	Kanye West	-0.48	0.34	tense
U	Ultimate	Denzel Curry	-0.85	0.40	tense

Showing 1 to 21 of 21 entries

Figure 5: The user can be seen drawing a path to generate a playlist with the desired emotional trajectory at the top. The result of the path-based playlist generation is presented at the bottom.

arousal, while lyrical information is more relevant for valence prediction Louro et al. (2024b).

Starting in the audio branch, Mel-spectrogram representations of each sample are fed to the feature learning portion of the baseline architecture presented in Louro et al. Louro et al. (2024a). It is composed of four convolutional blocks, composed of a 2D Convolutional layer, followed by a Batch Normalization, Dropout, and Max Pooling layer, finishing with ReLU activation. As for the lyrics branch, the word embeddings of lyrics are also fed to four convolutional blocks, each comprising a 1D Convolutional layer, followed by Max Pooling and a ReLU activation layers. To balance the information from each modality, we significantly reduce the overwhelming amount of learned features from lyrics using a Dense layer before merging the learned features from both branches.

The classification portion of the model is composed of alternating Droupout and Dense layers, which reduce and further process the set of features respectively, finally outputting one of Russell's Circumplex model's four quadrants.

3.2 Pre-Processing Steps

A set of pre-processing steps is necessary to obtain the data representations used for each branch of the architecture detailed above.

The librosa library McFee et al. (2015) is used to obtain the Mel-spectrogram representation for the audio branch. The audio samples, provided as mp3 files, are first converted to waveforms (.wav) and downsampled from 22.5 to 16kHz. This is done to reduce the complexity of the model, along with the computational cost for optimization. The downsampling has been shown to provide similar results to higher sampling rates, showing the robustness of DL approaches Pyrovolakis et al. (2022). The spectral representations are then generated using default parameters for the length of the Fast Fourier Transform window (2048) as well as the hop size (512).

As for word embeddings, the Sentence Transformer library from Hugging Face was used, specifically, the all-roberta-large-v1 pre-trained model. The embedder receives a context of up to 512 tokens and outputs a 1024 embedded vector. Given that the best results were provided by using the full context window, some of the lyrics had to be cut off at some point. After some simple tokenization steps, namely removing new line characters and converting all text to lowercase, the embeddings were obtained up to the already mentioned context size.

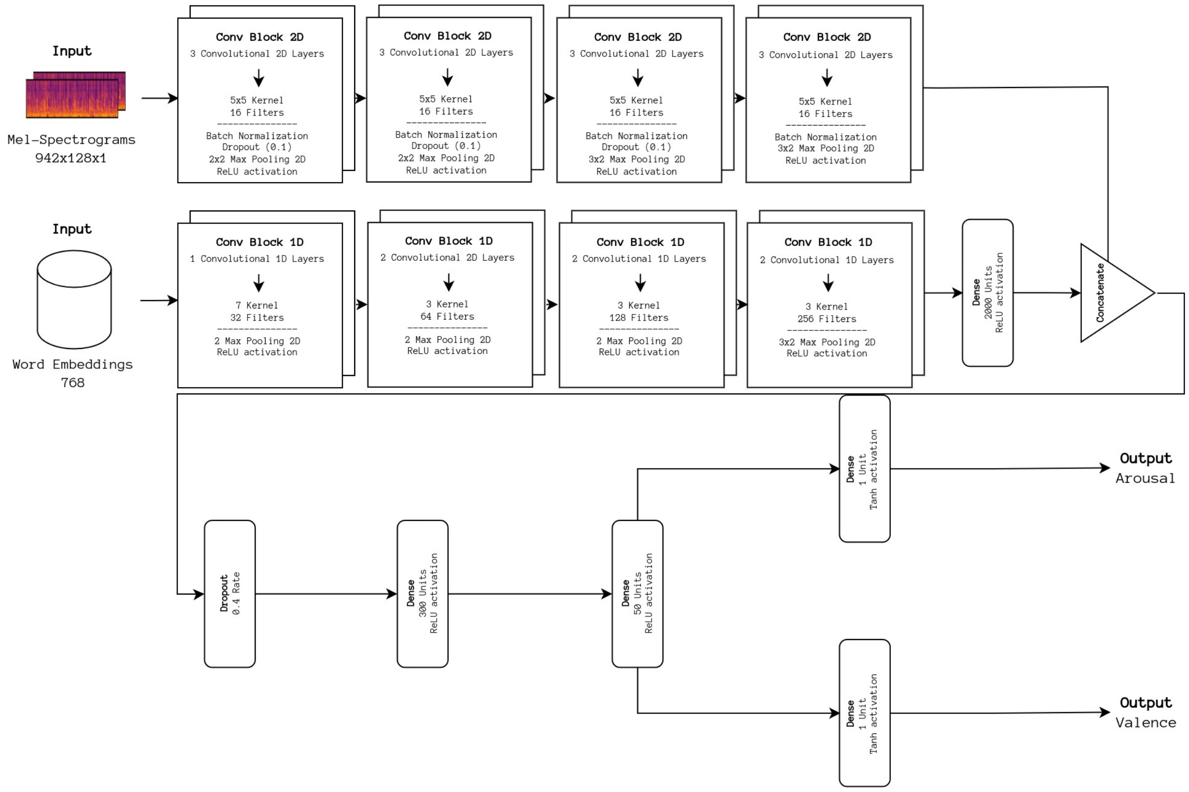


Figure 6: The multi-modal audio-lyrics regression model. Emotionally-relevant features are learned for both the audio representation in the Mel-spectrogram-receiving branch and the lyrics representation in the branch receiving the previously generated word embeddings. AV values are predicted after concatenating and processing the learned features from both branches.

3.3 Model Optimization

Model optimization was conducted using the Bayesian optimization implementation of the Keras Tuner library O'Malley et al. (2019). This method finds the best combination of hyperparameters in previously defined intervals for each, either maximizing or minimizing an objective function defined by the user.

Since our methodology is based on a regression task to predict arousal and valence for a given sample, the objective is defined as minimizing the sum of the mean squared error (MSE) for both. This ensures that none is prioritized, leveraging both audio's better predictability in terms of arousal and the same for lyrics' predictability of valence. The intervals for each considered hyperparameter, namely, batch size, optimizer, and corresponding learning rate, are presented in Table 1.

Table 1: Optimal Hyperparameters For Each Dataset.

Best Hyperparameters		
Batch Size	Optmizer	Learning Rate
64	SGD	1e-2

The optimization process is run over ten trials, per the library's default, starting at the lower end of each interval. For each trial, the model is trained to a maximum of 200 epochs, with an early stopping strategy defined to check for no improvements to the validation loss for 15 consecutive epochs. This considerably reduces the time needed to conduct the full optimization phase since less time is spent on underperforming sets of hyperparameters. We used a 70-15-15 train-validate-test (TVT) split as our validation strategy, as defined in Louro et al. (2024b). The resulting models for each trial are backed up for later usage, including the evaluation phase, which is discussed next.

3.4 Data and Evaluation

The MERGE Bimodal Complete dataset was used for validating our approach. Proposed in Louro et al. (2024b), it comprises a set of 2216 bimodal samples (audio clips and corresponding lyrics). For each sample, the dataset provides a 30-second audio excerpt of the most representative part of the song, links to the full lyrics, labels corresponding to each of the quadrants in Russell's Circumplex model, and AV values, used to obtain the previously mentioned labels, cal-

Table 2: TVT 70-15-15 Results For MERGE Audio Complete.

F1-score	Precision	Recall	R2 (A/V)	RMSE (A/V)
73.20%	74.53%	73.49%	0.454 0.506	0.133 0.339

culated based on the extracted emotion-related tags available in AllMusic [2].

The above-mentioned AV values are obtained through the following process. First, the available tags for each song in the dataset are obtained from the All Music platform. Using Warriner's Adjective Dictionary Warriner et al. (2013), the existing tags are translated to arousal and valence values. Finally, The values are then averaged across all tags corresponding to a specific song, obtaining its final mapping on Russell's Circumplex model.

For the TVT strategy, both the training and validation sets are used in the optimization function. The set of optimal hyperparameters is found using the latter. After training the model for each dataset, the following metrics are computed between the actual and predicted AV values in the test set for each class as well as for the overall performance: F1-score, Precision, Recall, R^2 (squared Pearson's correlation), and Root Mean Squared Error (RMSE).

Before computing these metrics, the predicted and real AV values were mapped to Russell's Circumplex model to obtain classes for calculating Precision, Recall, and F1-score.

4 EXPERIMENTAL RESULTS AND DISCUSSION

Tables 2 and 3 show the overall results for the discussed methodology. The arousal and valence standalone results for the R^2 and RMSE metrics are presented in consecutive lines in the order displayed in the tables.

The obtained results for both datasets are lower than those obtained in previous studies focused on static MER as a categorical problem Louro et al. (2024a). The best result attained is a 73.20% F1-score, which is around 6% lower than the results obtained for the same dataset and evaluation strategy in the mentioned article. The lower results are mostly due to the semi-automatic approach to obtain AV values (see Section 3.4, considering that the tags available on All Music are user-generated and its curation is unknown.

Table 3: TVT 70-15-15 Results Confusion Matrix For MERGE Audio Complete.

		Predicted			
		Q1	Q2	Q3	Q4
Actual	Q1	61.3%	10.4%	6.6%	21.7%
	Q2	9.8%	82.4%	5.9%	2.0%
	Q3	1.4%	4.3%	78.3%	15.9%
	Q4	7.3%	0.0%	18.2%	74.5%

As shown in Table 2, the R^2 metric for valence outperformed the one for arousal, although having a larger RMSE. This indicates that the relative valence throughout songs is reasonably captured, despite the larger RMSE error.

Although the attained results show room for improvement, they are a good starting point for the user. Given the subjective nature of each user's emotional perception, we believe that the personalization feature included in the MERGE app is a valuable mechanism for handling subjectivity in MER.

In terms of the results for separate quadrants (Table 3), we can see that some Q1 songs are confused with Q4 songs (21.77% Q1 songs are incorrectly classified as Q4). Moreover, there is also some confusion between Q3 and Q4 (15.9% of Q3 songs are predicted as Q4 and 18.2% of Q4 songs are classified as Q3). This is a known difficulty in MER, as discussed in Panda et al. (2020) that needs further research.

5 CONCLUSION AND FUTURE WORK

We presented the prototype for the MERGE application. Currently, the initial version has implemented music playback features, the ability to add and filter songs to a shared database, list and plane views, the latter based on Russell's Circumplex model, and user management functionalities. Moreover, a bimodal audio-lyrics model is incorporated into the backend of the prototype to allow for AV value prediction of user-uploaded songs. Path-based playlist generation has also been implemented, enabling users to craft a playlist that follows a specific emotional trajectory they have selected.

Still, many more functionalities are planned for the application in future iterations. The highlighted

[2] https://www.allmusic.com/

functionalities include user-generated tags for a more customized filtering experience that would be available to other users; automatic lyrics for the full song scraped from an available API, e.g., Genius; and Music Emotion Variation Detection (MEVD) prediction support, including visualization with the same color code used in the plot. A standalone desktop application is also planned without the cross-user features. in addition to implementing these upcoming features, We plan to conduct in-depth user experience studies to gain a more comprehensive understanding of the system's efficacy and user satisfaction.

Validation experiments on two recently proposed datasets are provided alongside a thorough system description, relaying insights into the obtained results. These are still below the categorical approach presented in Louro et al. (2024b) due to the already discussed semi-automatic AV mapping approach in Section 3.4. Despite this, the predictions are a good starting point to be further adjusted to the user's perception.

Regarding the actual model, neither feature learning portion may be ideal for the problem at hand since they were originally developed for a categorical problem. Developing more suitable architectures should thus be considered future work. Furthermore, the data representations, especially the word embeddings, may also be further improved, considering that the pre-trained model used is limited to a context window of 512 tokens.

To conclude, we believe the proposed app might be useful for music listeners. Although there is room for improvement (as the attained classification results show), the personalization mechanism is a useful feature for handling prediction errors and subjectivity. Finally, the personalization feature and the multi-user environment have the potential to acquire quality user annotations, leading to a future larger and more robust MER dataset.

ACKNOWLEDGEMENTS

This work is funded by FCT - Foundation for Science and Technology, I.P., within the scope of the projects: MERGE - DOI: 10.54499/PTDC/CCI-COM/3171/2021 financed with national funds (PIDDAC) via the Portuguese State Budget; and project CISUC - UID/CEC/00326/2020 with funds from the European Social Fund, through the Regional Operational Program Centro 2020. Renato Panda was supported by Ci2 - FCT UIDP/05567/2020.

We thank all reviewers for their valuable suggestions, which help to improve the article.

REFERENCES

Cardoso, L., Panda, R., and Paiva, R. P. (2011). Moodetector: A prototype software tool for mood-based playlist generation. In *Simpósio de Informática - INForum 2011*, Coimbra, Portugal.

Delbouys, R., Hennequin, R., Piccoli, F., Royo-Letelier, J., and Moussallam, M. (2018). Music Mood Detection Based On Audio And Lyrics With Deep Neural Net. In *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pages 370–375, Paris, France.

Hu, X., Downie, J. S., Laurier, C., Bay, M., and Ehmann, A. F. (2008). The 2007 Mirex Audio Mood Classification Task: Lessons Learned. In *Proceedings of the 9th International Society for Music Information Retrieval Conference*, pages 462–467, Drexel University, Philadelphia, Pennsylvania, USA.

Louro, P. L., Redinho, H., Malheiro, R., Paiva, R. P., and Panda, R. (2024a). A Comparison Study of Deep Learning Methodologies for Music Emotion Recognition. *Sensors*, 24(7):2201.

Louro, P. L., Redinho, H., Santos, R., Malheiro, R., Panda, R., and Paiva, R. P. (2024b). MERGE – A Bimodal Dataset for Static Music Emotion Recognition.

McFee, B., Raffel, C., Liang, D., Ellis, D., McVicar, M., Battenberg, E., and Nieto, O. (2015). Librosa: Audio and Music Signal Analysis in Python. In *Python in Science Conference*, pages 18–24, Austin, Texas.

O'Malley, T., Bursztein, E., Long, J., Chollet, F., Jin, H., Invernizzi, L., et al. (2019). Keras Tuner. https://github.com/keras-team/keras-tuner.

Panda, R., Malheiro, R., and Paiva, R. P. (2020). Novel Audio Features for Music Emotion Recognition. *IEEE Transactions on Affective Computing*, 11(4):614–626.

Pyrovolakis, K., Tzouveli, P., and Stamou, G. (2022). Multi-Modal Song Mood Detection with Deep Learning. *Sensors*, 22(3):1065.

Russell, J. A. (1980). A circumplex model of affect. *Journal of Personality and Social Psychology*, 39(6):1161–1178.

Warriner, A. B., Kuperman, V., and Brysbaert, M. (2013). Norms of valence, arousal, and dominance for 13,915 English lemmas. *Behavior Research Methods*, 45(4):1191–1207.

Contrato*360* 2.0: A Document and Database-Driven Question-Answer System Using Large Language Models and Agents

Antony Seabra[1,2][a], Claudio Cavalcante[1,2][b], João Nepomuceno[1][c], Lucas Lago[1][d], Nicolaas Ruberg[1][e] and Sergio Lifschitz[2][f]

[1]*BNDES, Área de Tecnologia da Informação, Rio de Janeiro, Brazil*

[2]*PUC-Rio, Departamento de Informática, Rio de Janeiro, Brazil*

Keywords: Information Retrieval, Question Answer, Large Language Models, Documents, Databases, Prompt Engineering, Retrieval Augmented Generation, Text-to-SQL.

Abstract: We present a question-and-answer (Q&A) application designed to support the contract management process by leveraging combined information from contract documents (PDFs) and data retrieved from contract management systems (database). This data is processed by a large language model (LLM) to provide precise and relevant answers. The accuracy of these responses is further enhanced through the use of Retrieval-Augmented Generation (RAG), text-to-SQL techniques, and agents that dynamically orchestrate the workflow. These techniques eliminate the need to retrain the language model. Additionally, we employed Prompt Engineering to fine-tune the focus of responses. Our findings demonstrate that this multi-agent orchestration and combination of techniques significantly improve the relevance and accuracy of the answers, offering a promising direction for future information systems.

1 INTRODUCTION

Contract management in large corporations involves overseeing legally binding agreements from their initiation through to execution and finalization. This process encompasses ensuring that services or products are delivered in accordance with contractual terms, monitoring their execution, and continuously evaluating both operational and financial performance throughout the service or product lifecycle. In the case of public sector companies, this process becomes even more complex due to stringent regulatory frameworks. In Brazil, for instance, Law No. 14,133/2021 mandates that contract management includes a wide range of activities, such as technical and administrative oversight, adherence to contract duration, reevaluation of economic and financial terms, modifications to service scope, and the enforcement of penalties and fines when necessary. These regulations impose an additional layer of complexity on the contract management process, demanding a robust and systematic approach to ensure compliance and efficiency.

Beyond contract managers, dedicated organizational units are essential to support the contract management process, ensuring that the diverse range of activities associated with contract execution is managed efficiently. Often, these units require specialized knowledge to handle complex services effectively. Notable examples include information and communication technology (ICT) services, property and asset management, and construction and engineering projects, each of which demands a high level of expertise. Additionally, these units typically rely on Contract Management Systems (CMS) to streamline their operations. Public companies may either develop these systems in-house or opt for widely-used market solutions, such as SAP Contract Life-cycle Management and IBM Emptoris Contract Management, among others.

While these systems efficiently handle general contract information, such as signatures, expiration dates, payment terms, and contract agents, many specific details required to support effective management activities remain accessible only through the original documents. For instance, traditional Contract Management Systems (CMS) are often unable to respond to inquiries concerning particular aspects of a contract, such as penalties, discounts, or fines associated with delays in service or product delivery. More-

[a] https://orcid.org/0009-0007-9459-8216
[b] https://orcid.org/0009-0007-6327-4083
[c] https://orcid.org/0009-0004-5441-8426
[d] https://orcid.org/0009-0001-4094-1978
[e] https://orcid.org/0009-0005-4388-4656
[f] https://orcid.org/0000-0003-3073-3734

Seabra, A., Cavalcante, C., Nepomuceno, J., Lago, L., Ruberg, N. and Lifschitz, S.
Contrato360 2.0: A Document and Database-Driven Question-Answer System Using Large Language Models and Agents.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 167-178
ISBN: 978-989-758-716-0; ISSN: 2184-3228

over, they lack the capability to provide insights into comparative characteristics across different contracts, such as penalty clauses related to database support agreements. These tasks are highly time-consuming.

The objective of this study is to provide a solution that aids contract managers in addressing queries related to both contract documents and data housed within traditional Contract Management Systems. One of the key challenges faced by contract managers is the time-consuming process of searching for and retrieving relevant information from lengthy and complex contract texts. To address this, we leverage state-of-the-art large-scale Language Modeling (LLM) technologies to analyze and extract pertinent details from contract documents efficiently. This not only improves the accuracy of the information retrieved but also significantly enhances the productivity of contract managers by reducing the manual effort required to locate specific information. Additionally, our approach integrates data from traditional Contract Management Systems, ensuring that responses are both relevant and comprehensive, thereby streamlining contract management activities.

In this work, we evaluated and integrated several Natural Language Processing (NLP) techniques to develop a Q&A system specifically designed for IC contracts, using contract PDF files and data from Contract Management Systems (CMS) as primary data sources. To enhance the relevance of user queries, prior work by (Seabra et al., 2024) employed Retrieval-Augmented Generation (RAG) techniques and a static approach to text-to-SQL for extracting relevant metadata from contract systems. Building upon this, our approach utilizes agents to dynamically improve the accuracy and contextual relevance of responses, with a particular focus on a context-aware text-to-SQL agent that interprets user queries more effectively. Furthermore, similar to (Seabra et al., 2024), we applied Prompt Engineering techniques to standardize responses and ensure greater precision in the answers provided.

One of the primary challenges in interpreting contract documents lies in distinguishing between relevance and similarity, a complexity that arises due to the standardized formats and repetitive textual structures commonly found in these documents. This standardization is a challenge for LLMs because there is a great deal of textual similarity, which does not necessarily translate into relevance. Using a mix of NLP techniques, we developed a solution that minimizes the impact of standardization and provides relevant answers. This approach made it possible to design a solution without needing traditional *fine-tuning* or re-training of language models.

The paper is organized as follows: Section 2 provides technical background on LLMs, *RAGs text-to-SQL*, *agents*, and *prompt* engineering. Section 3 discusses the methodology of the use of the presented techniques, while Section 4 details the architecture of our solution. Section 5 describes how we evaluated the proposed solution and the experimentation of the Q&A application. Finally, Section 6 concludes our study and proposes directions for future research in this field.

2 BACKGROUND

The dissemination of several applications in the area of Natural Language Processing (NLP) was made possible by Large Scale Language Models (LLMs), including question and answer (Q&A) systems. Recently, the use of agents has been introduced as a crucial component in LLM-based systems to orchestrate and manage task execution dynamically. Agents, such as router agents, SQL agents, and RAG agents, enable the efficient allocation of tasks by directing queries to the most suitable processing modules, enhancing system adaptability and performance. This approach allows LLMs to better handle complex queries, making responses more accurate and contextually relevant by integrating external data sources and specialized processing routines (Mialon et al., 2023).

2.1 Large Language Models

Large-scale Language Models (LLMs) have revolutionized the field of natural language processing with their ability to understand and generate human-like text. In their architecture, they utilize a specific neural network structure, *Transformers*, which allows the model to weight the influence of different parts of the input texts at different times (Vaswani et al., 2017).

Conversational applications, a specific use case for LLMs, specialize in generating text that is coherent and contextualized. This is achieved through training, in which the models are fed vast amounts of conversational data, allowing them to learn the nuances of dialogue (OpenAI, 2023a). In this way, LLMs have established a new paradigm for NLP. Moreover, by expanding the search space with external data or specializing through fine-tuning, LLMs become platforms for building specialized applications. In this work, all language models utilized were based on OpenAI's GPT series. Specifically, we employed the *text-davinci-002* model for generating embeddings and the *gpt-4-turbo* model for generating answers to user queries.

2.2 Retrieval-Augmented Generation (RAG)

According to (Chen et al., 2024), LLMs face significant challenges such as factual hallucination, outdated knowledge, and lack of domain-specific *expertise*. In response to these challenges, RAG represents a paradigm shift in the way LLMs process and generate text. The principle behind RAG involves using vector storage to retrieve text fragments similar to the input query (Gao et al., 2023b). This technique converts both the query text and the information database into high-dimensional vectors, allowing one to retrieve similar information, which is then fed to an LLM.

(Gao et al., 2023b) and (Feng et al., 2024) describe *frameworks* that exploit the advantages of this technique by providing additional data to the LLM without re-training the (Li et al., 2022) model. By dividing the available text into manageable chunks and embedding these chunks in high-dimensional vector spaces, it is possible to quickly retrieve contextually relevant information in response to a query, which informs the next processing steps. As shown in Figure 1, the first step (1) involves reading the textual content of the PDF documents into manageable chunks (*chunks*), which are then transformed (*embedding*) (2) into high-dimensional vectors. The text in vector format captures the semantic properties of the text, a format that can have 1536 dimensions.

These *embeddings* vectors are stored in a *vectorstore* (3), a database specialized in high-dimensional vectors. The vector store allows efficient querying of vectors through their similarities, using the distance for comparison (whether *Manhatan*, Euclidean or cosine).

Once the similarity metric is established, the query is *embedded* in the same vector space (4); this allows a direct comparison between the vectorized query and the vectors of the stored chunks, retrieving the most similar chunks (5), which are then transparently integrated into the LLM context to generate a *prompt* (6). The *prompt* is then composed of the question, the texts retrieved from the *vectorstore*, the specific instructions and, optionally, the *chat* history, all sent to the LLM which generates the final response (7).

In RAG, the *chunking* strategy is important because it directly influences the quality of the retrieved information. A well-designed chunk generation ensures that the information is cohesive and semantically complete, capturing its essence.

A key aspect of RAG is the difference between similarity and relevance. Similar passages may not contain the information relevant to answering a query, posing a challenge to accurately retrieve information, especially in cases where data comes from multiple documents with similar structure. In such contexts, documents may share a high degree of structural and lexical similarity, making it difficult for retrieval algorithms to distinguish between content that is merely similar in form and content that is truly relevant to a query.

2.3 Text-to-SQL

Text-to-SQL is a technology that enables the conversion of natural language queries into SQL commands based solely on the database schema, eliminating the need for knowledge of the underlying data (Liu et al., 2023). This approach leverages the capabilities of LLMs to understand and interpret human language, allowing users to retrieve data from databases through plain text input without requiring specialized knowledge of SQL syntax (Gao et al., 2023a).

By translating natural language into SQL queries, *text-to-SQL* brings complex database structures and end users closer together, making access more intuitive and efficient. This technique is particularly useful because it allows non-expert users to access databases by asking natural language queries. It improves data accessibility, reduces the learning curve associated with database querying, and speeds up data analysis processes, enabling more users to make data-driven decisions.

The main distinction between RAG and *text-to-SQL* techniques lies in how information is retrieved. RAG relies on retrieving text segments from a vector store that are similar to the user's question, and using these segments to generate a coherent and contextually relevant answer. This method is effective for questions where the answer can be synthesized from existing text. However, it is not always possible to identify the information expected as the answer. In another aspect, *text-to-SQL* translates natural language queries into SQL commands, as demonstrated in (Pinheiro et al., 2023), which are then executed against a structured database to retrieve exact data matches. This ensures that if the text-to-SQL translation is accurate, the user will receive a highly specific answer directly from the database fields.

Therefore, while RAG operates on the principle of textual similarity and generative capabilities, *text-to-SQL* offers a more intrusive mechanism for data retrieval by executing queries that directly match the user's intent, making it particularly effective for data investigations.

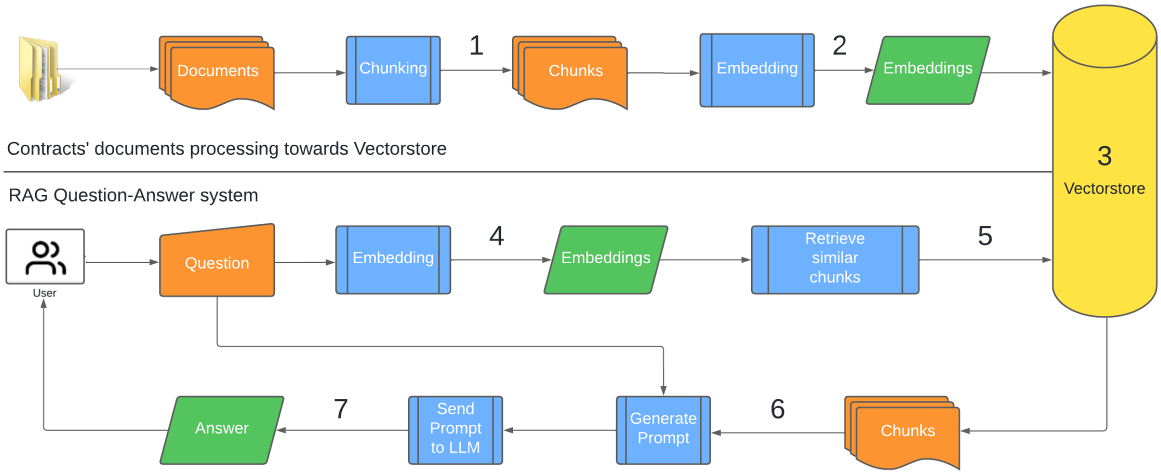


Figure 1: Retrieval-Augmented Generation.

2.4 Prompt Engineering

Prompt engineering is the art of designing and optimizing *prompts* to guide LLMs in generating desired outputs. The goal of *prompt* engineering is to maximize the potential of LLMs by providing them with instructions and context (OpenAI, 2023b).

In the context of *prompt* engineering, prompts are a fundamental part of the process. Through prompts, engineers can outline the script for a response, specifying the desired style and format for the LLM response (White et al., 2023) (Giray, 2023). For example, to define the style of a conversation, a *prompt* could be formulated as ”Use professional language and treat the customer with respect” or ”Use informal language and emojis to convey a friendly tone.” To specify the format of dates in responses, a *prompt* instruction could be ”Use the American format, MM/DD/YYYY, for all dates.”

On the other hand, context refers to the information provided to LLMs along with the main prompts. The most important aspect of context is that it can provide additional information to support the response given by the LLM, which is very useful when implementing Q&A systems. This supplemental context can include relevant background details, specific examples, and even previous dialogue exchanges, which collectively help the model generate more accurate, detailed, and contextually appropriate responses. According to (Wang et al., 2023), *prompts* provide guidance to ensure that the model generates responses that are aligned with the user's intent. As a result, well-crafted *prompts* significantly improve the effectiveness and appropriateness of responses.

Recent studies have begun to explore the synergistic integration of these techniques with LLMs to create more sophisticated Q&A systems. For example, (Jeong, 2023) reinforces the importance of using Prompt Engineering with RAG to improve the retrieval of relevant documents, which are then used to generate both contextually relevant and information-rich answers. Similarly, (Gao et al., 2023a) explores the integration of *text-to-SQL* with Prompt Engineering to enhance the model's ability to interact directly with relational databases, thereby expanding the scope of queries that can be answered accurately.

2.5 Agents

The use of agents in applications built around Large Language Models (LLMs) is relatively recent but has already became common. Agents act as intelligent intermediaries that route, process, and present information in ways tailored to the context of the query. These agents leverage recent advancements in AI, such as Retrieval-Augmented Generation (RAG) and tool utilization, to perform more complex and contextually aware tasks (Lewis et al., 2020). They play a pivotal role in orchestrating complex tasks, integrating various data sources, and ensuring that the system responds accurately and efficiently to user queries.

In a complex LLM-based system, different tasks often require specialized handling. Agents enable task orchestration by directing queries to the most appropriate component, whether it's for retrieving data, performing calculations, or generating visualizations. For example, an application may have a Text-to-SQL agent to perform queries over a relational database and a Graph agent to visualize graphs after an answer, if appropriate. According to (Jin et al., 2024), applying LLMs to text-to-database management and query optimization is also a novel research direction in natural language to code generation task. By converting natural language queries into SQL statements, LLMs help developers quickly generate efficient database query code. In the realm of integrating heterogeneous data sources, Q&A applications often need to access data from documents, databases, APIs, and other repositories. Agents facilitate the seamless integration of these heterogeneous data sources, allowing the system to extract relevant information dynamically.

There are several agent types. As outlined in (Singh et al., 2024), agent workflows allow LLMs to operate more dynamically by incorporating specialized agents that manage task routing, execution, and optimization. These agents serve as intelligent intermediaries, directing specific tasks—such as data retrieval, reasoning, or response generation—to the most suitable components within the system. One of the most important ones in place are the Router Agents, as they are the decision-makers of the system. When a user poses a query, the router agent analyzes the input and decides the best path forward. For instance, if a query is identified as needing factual data, the router agent might direct it to a RAG model. If the question involves specific data retrieval from a database, it will engage an SQL agent instead.

As mentioned before, RAG and SQL Agents are very relevant too. According to (Saeed et al., 2023), SQL agents can effectively manage data retrieval tasks by leveraging LLMs. The SQL queries are transformed into prompts for LLMs, allowing the system to interact with unstructured data stored in the model, mimicking traditional database operations. (Fan et al., 2024) provides a comprehensive overview of the integration of RAG techniques in LLMs but moreover, (Wang et al., 2024) introduces a novel approach that combines RAG techniques with a drafting-verification process to improve the reasoning capabilities of LLMs when handling retrieved documents. The RAG agent, termed the ”drafter,” generates multiple answer drafts based on retrieved results, while a larger generalist LLM, the ”verifier,” assesses these drafts and selects the most accurate one. This approach effectively integrates retrieval and generation, enhancing the overall performance of LLMs in knowledge-intensive tasks such as question answering and information retrieval systems.

3 METHODOLOGY

To address the challenges faced by contract managers in terms of complex information retrieval, we propose Contrato360, a Q&A system supported by an LLM and orchestrated by agents. The system employs a range of techniques designed to enhance the relevance of responses while mitigating the risks associated with the standardized textual structures of contracts.

To achieve this goal of increasing the relevance of the responses obtained by Contrato*360*, we combined four techniques: 1) Retrieval-Augmented Generation (RAG) to increase the relevance of information about contracts contained in PDF documents; 2) Agents to orchestrate and route the flow of execution, enabling the dynamic selection of the most appropriate approach for each query context; 3) Text-to-SQL agent to retrieve the relevant information from contract systems; 4) Prompt Engineering techniques to standardize and ensure greater accuracy in the responses produced.

3.1 Applying RAG

One of the first decisions to be made is to choose the best strategy to segment the document, that is, how to perform the *chunking* of the PDF files. A common *chunking* strategy involves segmenting documents based on a specific number of *tokens* and an overlap (*overlap*). This is useful when dealing with sequential texts where it is important to maintain the continuity of the context between the *chunks*.

Contracts have a standardized textual structure, organized into contractual sections. Therefore, sections with the same numbering or in the same vicinity describe the same contractual aspect, that is, they have similar semantics. For example, in the first section of contract documents, we always find the object of the contract. In this scenario, we can assume that the best *chunking* strategy is to separate the *chunks* by section of the document. In this case, the *overlap* between the *chunks* occurs by section, since the questions will be answered by information contained in the section itself or in previous or subsequent sections. For the contract page in the example in Figure 3, we would have a *chunk* for the section on the object of the contract, another *chunk* for the section on the term of the contract, that is, a *chunk* for each clause of the contract and its surroundings. This approach ensures that each snippet represents a semantic unit, making retrievals more accurate and aligned with queries.

Having the contract section as the limit of the *chunks* improves the relevance of the responses within a single contract. However, when increasing the number of contracts that the Contract*360* intends to respond to, we observe the problem in correctly determining the contract to be treated. In the following example, we detail this aspect:

Consider the contract documents shown in Figure 3. showcases two service contracts be-

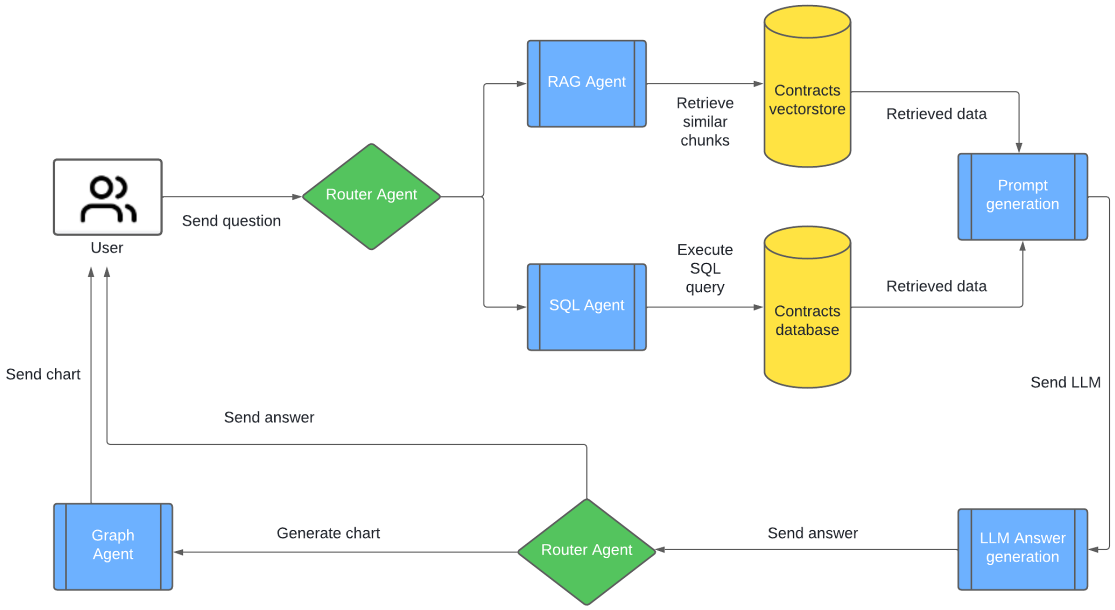


Figure 2: Methodology Workflow Combining Different Techniques.

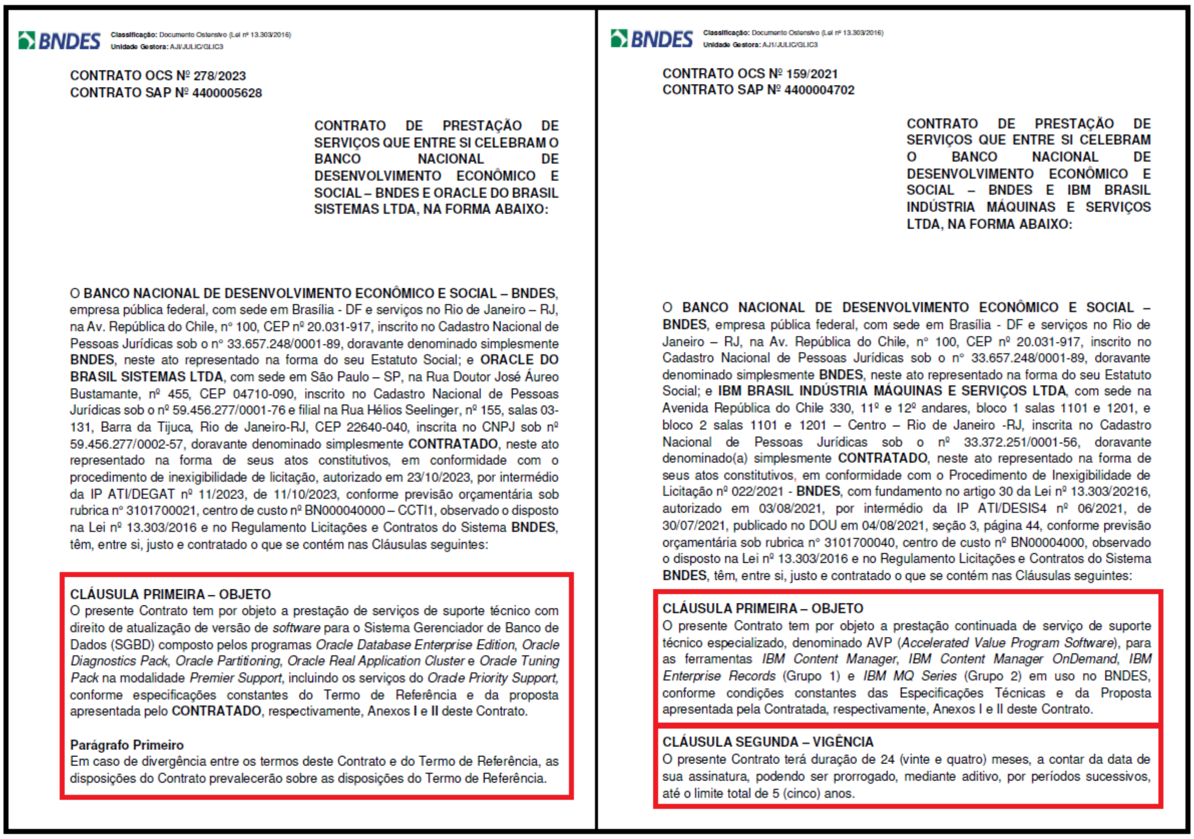
tween BNDES (Banco Nacional de Desenvolvimento Econômico e Social) and companies (Oracle do Brasil Sistemas Ltda. and IBM Brasil Indústria Máquinas e Serviços Ltda.), highlighting key clauses relevant to the provision of technical support and software updates. The contracts are presented in Portuguese, reflecting the original legal terms and specific obligations of each party. For instance, the contract with Oracle (Contract No. 278/2023) details the provision of services for Oracle Database and associated technologies, emphasizing software support and entitlement to updates. Similarly, the contract with IBM (Contract No. 159/2021) focuses on support services related to IBM Content Management software. The `"CLÁUSULA PRIMEIRA - OBJETO"` (first clause - object) details the object of the contract and a frequently asked question is: *"What is the object of contract OCS 278/2023?"*. In this example, the RAG will store vectors containing the sections of both contracts, since this clause is common to both. However, when we inspect what is expressed in the *chunk*, its content does not contain the contract number, Figure 3. Thus, with great probability, a query about a specific contract may return a segment (*chunk*) unrelated to the contract, for example OCS 159/2021, being retrieved instead of the contract we want. In the case of our example, the *chunk* referring to the question that should be returned is related to contract OCS 278/2023.

To overcome this issue, it is necessary to add semantics to the *chunks*, by including document metadata. And when accessing the *vectorstore*, use this metadata to filter the information returned. In this way, we improve the relevance of the retrieved texts. Figure 4 displays the most relevant metadata for the contracts (source, contract and clause). Where source is the name of the contract PDF file), contract is the OCS number and clause is the section title. Thus, for the question *"What is the object of contract OCS 278/2023?"*, the *chunks* of contract OCS 278/2023 are retrieved and then the similarity calculation is applied, retrieving the text segments to be sent to the LLM.

3.2 Applying *Text-to-SQL*

Contracts are dynamic and undergo several events like operational changes and management adjustments throughout their life-cycle. To deal with this complexity, organizations use contract monitoring systems, such as *SAP Contract Life-cycle Management* and *IBM Emptoris Contract Management*. These systems control several aspects, such as the technical person responsible for the contract, changes in the contractor's representative, and the end of the provision of services. During the contract term, these events can occur and significantly affect contract management.

The Contract*360* retrieves those events from the Contract Management System (CMS) and incorporates them so the LLM can provide relevant responses to the user. Therefore, a *text-to-SQL* technique was natural to implement the reasoning and decision-making task (Yao et al., 2023) to obtain relevant responses from the CMS database to the contract managers.



BNDES **Classificação:** Documento Ostensivo (Lei nº 13.303/2016)
Unidade Gestora: AJ/JULIC/GLIC3

CONTRATO OCS Nº 278/2023
CONTRATO SAP Nº 4400005628

CONTRATO DE PRESTAÇÃO DE SERVIÇOS QUE ENTRE SI CELEBRAM O BANCO NACIONAL DE DESENVOLVIMENTO ECONÔMICO E SOCIAL – BNDES E ORACLE DO BRASIL SISTEMAS LTDA, NA FORMA ABAIXO:

O **BANCO NACIONAL DE DESENVOLVIMENTO ECONÔMICO E SOCIAL – BNDES**, empresa pública federal, com sede em Brasília - DF e serviços no Rio de Janeiro – RJ, na Av. República do Chile, nº 100, CEP nº 20.031-917, inscrito no Cadastro Nacional de Pessoas Jurídicas sob o nº 33.657.248/0001-89, doravante denominado simplesmente **BNDES**, neste ato representado na forma do seu Estatuto Social; e **ORACLE DO BRASIL SISTEMAS LTDA**, com sede em São Paulo – SP, na Rua Doutor José Áureo Bustamante, nº 455, CEP 04710-090, inscrito no Cadastro Nacional de Pessoas Jurídicas sob o nº 59.456.277/0001-76 e filial na Rua Hélios Seelinger, nº 155, salas 03-131, Barra da Tijuca, Rio de Janeiro-RJ, CEP 22640-040, inscrita no CNPJ sob nº 59.456.277/0002-57, doravante denominado simplesmente **CONTRATADO**, neste ato representado na forma de seus atos constitutivos, em conformidade com o procedimento de inexigibilidade de licitação, autorizado em 23/10/2023, por intermédio da IP ATI/DEGAT nº 11/2023, de 11/10/2023, conforme previsão orçamentária sob rubrica nº 3101700021, centro de custo nº BN000040000 – CCTI1, observado o disposto na Lei nº 13.303/2016 e no Regulamento Licitações e Contratos do Sistema **BNDES**, têm, entre si, justo e contratado o que se contém nas Cláusulas seguintes:

CLÁUSULA PRIMEIRA – OBJETO
O presente Contrato tem por objeto a prestação de serviços de suporte técnico com direito de atualização de versão de *software* para o Sistema Gerenciador de Banco de Dados (SGBD) composto pelos programas *Oracle Database Enterprise Edition*, *Oracle Diagnostics Pack*, *Oracle Partitioning*, *Oracle Real Application Cluster* e *Oracle Tuning Pack* na modalidade *Premier Support*, incluindo os serviços do *Oracle Priority Support*, conforme especificações constantes do Termo de Referência e da proposta apresentada pelo **CONTRATADO**, respectivamente, Anexos **I** e **II** deste Contrato.

Parágrafo Primeiro
Em caso de divergência entre os termos deste Contrato e do Termo de Referência, as disposições do Contrato prevalecerão sobre as disposições do Termo de Referência.

BNDES **Classificação:** Documento Ostensivo (Lei nº 13.303/2016)
Unidade Gestora: AJ1/JULIC/GLIC3

CONTRATO OCS Nº 159/2021
CONTRATO SAP Nº 4400004702

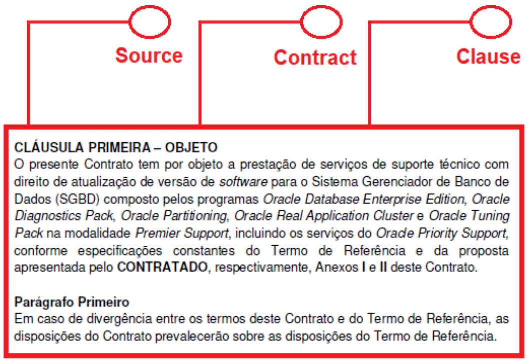
CONTRATO DE PRESTAÇÃO DE SERVIÇOS QUE ENTRE SI CELEBRAM O BANCO NACIONAL DE DESENVOLVIMENTO ECONÔMICO E SOCIAL – BNDES E IBM BRASIL INDÚSTRIA MÁQUINAS E SERVIÇOS LTDA, NA FORMA ABAIXO:

O **BANCO NACIONAL DE DESENVOLVIMENTO ECONÔMICO E SOCIAL – BNDES**, empresa pública federal, com sede em Brasília - DF e serviços no Rio de Janeiro – RJ, na Av. República do Chile, nº 100, CEP nº 20.031-917, inscrito no Cadastro Nacional de Pessoas Jurídicas sob o nº 33.657.248/0001-89, doravante denominado simplesmente **BNDES**, neste ato representado na forma do seu Estatuto Social; e **IBM BRASIL INDÚSTRIA MÁQUINAS E SERVIÇOS LTDA**, com sede na Avenida República do Chile 330, 11º e 12º andares, bloco 1 salas 1101 e 1201, e bloco 2 salas 1101 e 1201 – Centro – Rio de Janeiro -RJ, inscrita no Cadastro Nacional de Pessoas Jurídicas sob o nº 33.372.251/0001-56, doravante denominado(a) simplesmente **CONTRATADO**, neste ato representado na forma de seus atos constitutivos, em conformidade com o Procedimento de Inexigibilidade de Licitação nº 022/2021 - **BNDES**, com fundamento no artigo 30 da Lei nº 13.303/20216, autorizado em 03/08/2021, por intermédio da IP ATI/DESIS4 nº 06/2021, de 30/07/2021, publicado no DOU em 04/08/2021, seção 3, página 44, conforme previsão orçamentária sob rubrica nº 3101700040, centro de custo nº BN00004000, observado o disposto na Lei nº 13.303/2016 e no Regulamento Licitações e Contratos do Sistema **BNDES**, têm, entre si, justo e contratado o que se contém nas Cláusulas seguintes:

CLÁUSULA PRIMEIRA – OBJETO
O presente Contrato tem por objeto a prestação continuada de serviço de suporte técnico especializado, denominado AVP (*Accelerated Value Program Software*), para as ferramentas *IBM Content Manager*, *IBM Content Manager OnDemand*, *IBM Enterprise Records* (Grupo 1) e *IBM MQ Series* (Grupo 2) em uso no BNDES, conforme condições constantes das Especificações Técnicas e da Proposta apresentada pela Contratada, respectivamente, Anexos I e II deste Contrato.

CLÁUSULA SEGUNDA – VIGÊNCIA
O presente Contrato terá duração de 24 (vinte e quatro) meses, a contar da data de sua assinatura, podendo ser prorrogado, mediante aditivo, por períodos sucessivos, até o limite total de 5 (cinco) anos.

Figure 3: Chunking applied to Contracts.



CLÁUSULA PRIMEIRA – OBJETO
O presente Contrato tem por objeto a prestação de serviços de suporte técnico com direito de atualização de versão de *software* para o Sistema Gerenciador de Banco de Dados (SGBD) composto pelos programas *Oracle Database Enterprise Edition*, *Oracle Diagnostics Pack*, *Oracle Partitioning*, *Oracle Real Application Cluster* e *Oracle Tuning Pack* na modalidade *Premier Support*, incluindo os serviços do *Oracle Priority Support*, conforme especificações constantes do Termo de Referência e da proposta apresentada pelo **CONTRATADO**, respectivamente, Anexos **I** e **II** deste Contrato.

Parágrafo Primeiro
Em caso de divergência entre os termos deste Contrato e do Termo de Referência, as disposições do Contrato prevalecerão sobre as disposições do Termo de Referência.

Figure 4: Contracts metadata.

The LangChain SQL Agent (Langchain, 2024) has proven to be a highly flexible tool for interacting with the CMS database. Upon system startup, our SQL agent establishes an authenticated connection to the database and retrieves the schema. When it receives a user question, it performs Entity Recognition, maps those entities to the database tables and columns, and prepares the SQL statement.

Ensuring the safety of our SQL agent is central. We validate each generated query to ensure it does not contain harmful commands, such as 'UPDATES,' 'DROP TABLE,' 'INSERT,' or any other command that can alter the database, providing a sense of security about the system's integrity.

Finally, the output generated from the executed SQL statement goes to a prompt generation stage for further analysis of the LLM.

3.3 Applying Prompt Engineering

The *prompt* engineering technique provides a pattern for the style of responses and the reuse of the solution when accessing the LLM, as it provides instructions and context. Based on these observations, instructions were developed in the application to improve the responses. The instructions include basic guidelines, such as *"Do not use prior knowledge"*, which ensures that the responses are based only on *vectorstore* contracts, and specific instructions, such as *"Whenever you answer a question about a contract, provide the OCS number."* Thus, the question *"Do we have an Oracle Support contract?"* would have as a possible answer *"Yes, we have an Oracle Database Support contract. The OCS number is 278/2023."*.

Maintenance and guidelines on how to use the chat context were also applied to ensure uniformity and coherence. For example, we inform the expected style for responses: *"You should use a formal and objective tone."*, determining the style of LLM responses. Another guideline instructs LLM: *"Given the chat history and the question asked, construct the response completely, without the user needing to review the history"*.

Finally, the context passed to the LLM can be useful to establish the style of the answers according to the role of the user of the Q&A system. In the case of Contract*360*, we have three roles: 1) contract manager; 2) contract management support; and 3) manager of the contract management support unit. For each of these roles a specific context was defined, for example for role 3 we have: *"You are an assistant specialized in answering questions about administrative contracts, who provides management and summarized information about the contracts."*

With these three techniques we obtained more relevant answers. In the following section, we detail the implementation and the components used in the development of the system.

3.4 Applying Agents

In Contrato360, Agents play a pivotal role in orchestrating the flow of execution and enhancing the overall efficiency of the question-and-answer process. Also, considering the workflow on understanting the user query, an agent approach is a clever choice to implement this several specialized activities that needs to be taken in building the correct answer for the user. We designed three agents to implement this workflow.

As shown in figure 2, the Router Agent is central to its architecture, acting as the primary decision-making entity that orchestrates the flow of tasks needed to answer a user's question. The "Router Agent" decides if the user's question is related to the Contract Manager domain, e.g., "How are you?", "Will Bologna FC win the 2025 Champions League?" or "Who is the contract manager for the Database support?". An out-of-topic question is redirected to the LLM with a context limiting its role to the domain of contract management. In A question on the contract domain will follow our workflow to find a relevant answer.

In the sequel, the Router Agent sends the user question to two specialized agents: a) SQL agent and b) RAG agent. The RAG agent retrieves from the *vectorstore* chunks of documents similar to the user question. In parallel, a SQL agent retrieves form the CMS database content related to the user question. This architectural choice proved to be robust in the reports of the contract managers, as it semantically enriches the contract information, as shown in Figure 1.

One of the specialized agents in Contrato360 is the RAG (Retrieval Augmented Generation) Agent, responsible for retrieving relevant information from the contracts *vectorstore*. When directed by the Router Agent, the RAG Agent searches for similar data chunks that can help contextualize the question. Another specialized component is the SQL Agent, which handles queries requiring structured data extraction from the contracts database. Upon receiving routing instructions from the Router Agent, the SQL Agent executes SQL queries to retrieve specific data points relevant to the user's question.

With all textual and information retrieve, another "Router Agent" craft an answer. If needed to add an visual information, the Graph Agent and LLM Answer Generation Agent add further depth to Contrato360's response capabilities. The Graph Agent is tasked with creating visual representations, such as charts, when the Router Agent determines that a visual answer would better serve the user's needs. This agent ensures that complex data can be conveyed in a clear and understandable format, enhancing user comprehension. Meanwhile, the LLM Answer Generation Agent works closely with the prompt generation module to produce coherent and contextually relevant textual responses. Together, these agents provide a multi-faceted approach to answering questions, combining data retrieval, visualization, and language generation to deliver comprehensive solutions.

4 ARCHITECTURE

The architecture of the Contrato360 application illustrates a comprehensive system designed to facilitate a question-answering application that integrates Large Language Models (LLMs), document processing, and databases. The architecture consists of three main layers: the User Interface Layer, the Backend Layer, and the Language Model Integration Layer, each playing its role in delivering accurate and context-aware responses to users.

The User Interface Layer is represented by the User Interface (Streamlit), which serves as the front-end of the application. This layer provides an interactive platform where users can input their queries and view the responses generated by the system. The interface directly communicates with the backend layer, sending user inputs for processing and displaying the responses generated by the various integrated components.

At the heart of the system lies the Backend Layer, which is primarily managed by the Backend Agents

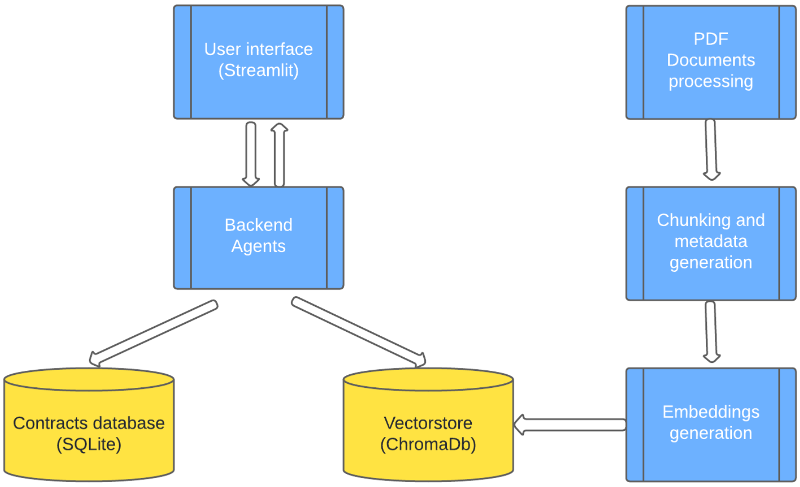

User interface (Streamlit)
PDF Documents processing
Backend Agents
Chunking and metadata generation
Contracts database (SQLite)
Vectorstore (ChromaDb)
Embeddings generation

Figure 5: Application architecture.

(Python and Langchain). This layer orchestrates interactions between the document processing, vector storage, contracts database, and the language model integration layer. The backend layer leverages Python and Langchain to handle the logic, task execution, and chat functionalities, particularly through OpenAI's chat models. It processes user inputs received from the interface and interacts with both the Contracts Database and Vectorstore (ChromaDb) to retrieve relevant information necessary for formulating comprehensive answers.

Within the backend layer, the Contracts Database (SQLite) serves as the structured data source, storing structured information related to contracts. This component allows the system to handle contract-related questions by processing SQL queries generated by the backend agents. The contracts database responds to these queries with relevant data, which is then used to construct natural language responses for the user.

The Vectorstore (ChromaDb) is another vital component of the backend layer, acting as a storage solution for vectorized data, including document embeddings. It plays a key role in efficient similarity searches and retrieval tasks, enhancing the system's ability to provide context-aware responses. The backend agents utilize the Vectorstore to match user queries against stored embeddings, enabling advanced semantic search capabilities. This component also stores embeddings generated from document processing, ensuring that data is readily available for future query matching.

The Language Model Integration Layer is responsible for transforming and embedding data for use within the system. This layer includes the PDF Documents Processing module, which ingests and preprocesses documents, particularly PDFs, to make them suitable for use within the application. This step involves reading and extracting text and relevant metadata, preparing the content for the next stages of processing. The Chunking and Metadata Generation component further refines the documents by dividing them into manageable chunks and generating metadata that improves retrieval efficiency, ensuring that the data is optimally split for better embedding generation and response times.

The final stage of the language model integration layer is the Embeddings Generation module, which converts the chunked documents and metadata into vector embeddings using LLM-based models like OpenAI Embeddings. These embeddings capture the semantic nuances of the text, facilitating efficient search and retrieval tasks within the system. Once generated, these embeddings are stored in the Vectorstore (ChromaDb), where they can be accessed for matching against user queries.

The overall workflow begins when a user inputs a question through the User Interface Layer, initiating a sequence of processes across the backend and language model integration layers. The backend agents handle query processing, interacting with the Contracts Database for SQL queries and performing semantic searches using embeddings from the Vectorstore. The document processing involves preprocessing PDFs, chunking the content, and generating embeddings that are then stored for efficient retrieval. The backend agents combine data retrieved from the contracts database and the Vectorstore to generate a coherent response, which is then presented back to the user through the User Interface Layer.

This architecture effectively combines the User Interface Layer, Backend Layer, and Language Model Integration Layer, enabling Contrato360 to function as a robust and powerful application for answering questions based on complex data sources. The seamless integration of multiple technologies ensures that users receive accurate and contextually relevant responses, enhancing the overall functionality and usability of the system.

5 EVALUATION

The experiment to validate the application was conducted by two IT contract specialists from BNDES. The system was prepared with 75 contracts (PDFs and data from the contract system). And to validate the relevance of the answers, *benchmark* questions were prepared, from two distinct groups: ”direct” and ”indirect” questions. ”Direct” questions are those that can be answered through the PDFs and their metadata. ”Indirect” questions are those that obtain better relevance when searched in the contract system data. In Tables 1 and 2 we present the users' perception of the quality of the answers. In the evaluation, the relevance of the answers was categorized as ”Correct” and ”Incomplete”.[1]

We can observe that for the ”direct” questions the system presents relevant answers for all experiments. However, in the ”indirect” questions, despite being satisfactory, the results in one specific question were limited and incomplete. In our evaluation, these questions require a more elaborate semantic evaluation. In the first case, we realized that the concept of ”Waiver of Bidding” was not well captured. We believe that an adjustment in the queries and/or in the prompt can add this type of semantics.

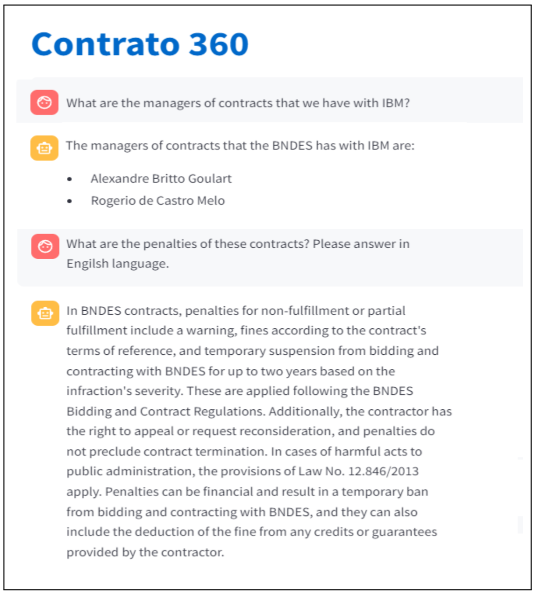
Table 1: Direct Questions.

Question	Correct	Incomplete
What is the subject of the OCS nnn/yy contract?	10	-
Do we have any contract whose subject is xxxx?	9	1
Do we have any contract with the supplier xxx?	10	-
Who is the manager of the OCS nnn/yy contract?	10	-
Who is the supplier of the nnn/yy contract?	10	-
What is the term of the OCS nnn/yy contract?	10	-

A key aspect observed from the users is the solution's capability to combine answers from both the structured data store and the contract's texts. This integration is perceived as a significant time-saving feature, as users typically need to locate the relevant contracts, open the respective PDFs, and manually search for additional information. The example below illustrates this. It identifies contract managers and outlines the penalties associated with contractual non-compliance. The system's ability to deliver precise, context-relevant answers from contracts highlights its effectiveness in reducing manual search efforts for users.

Table 2: Indirect Questions.

Question	Correct	Incomplete
How many active IT contracts do we currently have?	10	-
List the contracts that will end in the year yy?	10	-
How many contracts do we have with supplier xxxx?	10	-
How many contracts have we signed due to inflexibility?	9	1
How many DLs (Exemptions from Tenders) were contracted in yy?	-	10
Who are the managers of the contracts we have with company xxxx?	8	2
How many contracts does employee xxxx have under his/her management?	8	2
Show a summary of contract nnn/yy.	10	-



Contrato 360

What are the managers of contracts that we have with IBM?

The managers of contracts that the BNDES has with IBM are:

- Alexandre Britto Goulart
- Rogerio de Castro Melo

What are the penalties of these contracts? Please answer in Engilsh language.

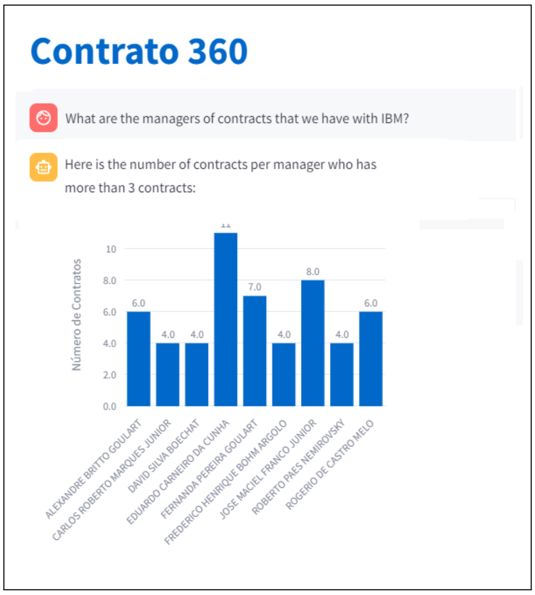
In BNDES contracts, penalties for non-fulfillment or partial fulfillment include a warning, fines according to the contract's terms of reference, and temporary suspension from bidding and contracting with BNDES for up to two years based on the infraction's severity. These are applied following the BNDES Bidding and Contract Regulations. Additionally, the contractor has the right to appeal or request reconsideration, and penalties do not preclude contract termination. In cases of harmful acts to public administration, the provisions of Law No. 12.846/2013 apply. Penalties can be financial and result in a temporary ban from bidding and contracting with BNDES, and they can also include the deduction of the fine from any credits or guarantees provided by the contractor.

Figure 6: Contracts Q&A Streamlit application.

In fact, by directly addressing questions with specific details, the system saves time and improves the user experience, as users can quickly access critical information without sifting through lengthy documents. Finally, the system's ability to automatically generate graphs using its Plotly agent, when a table of

[1] A third category would be ”Incorrect”, but this option was not obtained in any of the questions.

values is included in the response, has been positively received by users. This feature not only provides immediate visual insights, enhancing the understanding of the data, but also supports users in creating professional presentations. The integration of dynamic graph generation into the query response process significantly enriches the user experience, allowing for a more comprehensive analysis and efficient communication of contract-related information.



What are the managers of contracts that we have with IBM?

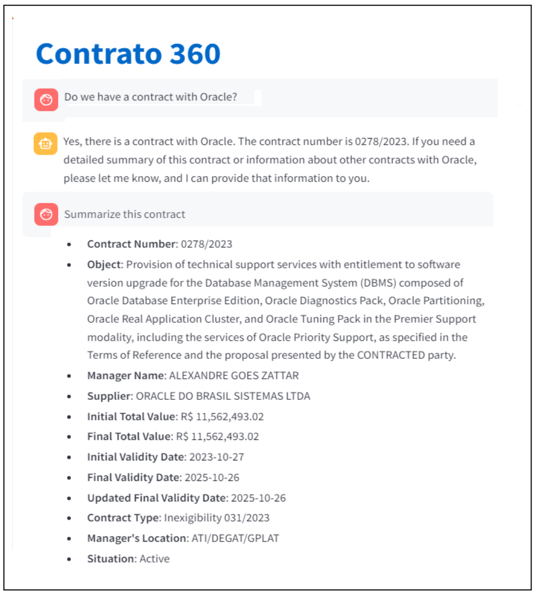
Here is the number of contracts per manager who has more than 3 contracts:

Figure 7: Plotly Agent.

6 CONCLUSIONS

We developed a Q&A application in the domain of service and product contracts, using PDF contracts and data from the Contract Management System as information sources. In this development, we employed four techniques to improve the relevance of the answers: 1) Augmented Retrieval (RAG) combined with semantic augmentation using metadata to retrieve information from PDFs; 2) Text-to-SQL, aggregating dynamic information from the contracts made available in the Contract Management System; 3) Prompt Engineering to contextualize, instruct and direct the answers produced by the LLM; and 4) Agents to call the most appropriate approach depending on query context and determining the flow of execution of tasks in the system.

The 8 demonstrates the ability of Contrato360 in retrieving and summarizing contract information related to Oracle through a question-and-answer interface. When asked if there is a contract with Oracle, the system efficiently identifies the relevant contract, numbered 0278/2023, and provides a concise summary of its key details stored in the database. The summarized information includes the contract's object, which covers technical support and software upgrades for Oracle's Database Management System (DBMS), details about the contract manager, supplier, total value, validity dates, and the current situation. This functionality highlights the system's ability to streamline access to specific contract data, facilitating quick and accurate information retrieval for users by directly interacting with the database through natural language queries



Do we have a contract with Oracle?

Yes, there is a contract with Oracle. The contract number is 0278/2023. If you need a detailed summary of this contract or information about other contracts with Oracle, please let me know, and I can provide that information to you.

Summarize this contract

- **Contract Number**: 0278/2023
- **Object**: Provision of technical support services with entitlement to software version upgrade for the Database Management System (DBMS) composed of Oracle Database Enterprise Edition, Oracle Diagnostics Pack, Oracle Partitioning, Oracle Real Application Cluster, and Oracle Tuning Pack in the Premier Support modality, including the services of Oracle Priority Support, as specified in the Terms of Reference and the proposal presented by the CONTRACTED party.
- **Manager Name**: ALEXANDRE GOES ZATTAR
- **Supplier**: ORACLE DO BRASIL SISTEMAS LTDA
- **Initial Total Value**: R$ 11,562,493.02
- **Final Total Value**: R$ 11,562,493.02
- **Initial Validity Date**: 2023-10-27
- **Final Validity Date**: 2025-10-26
- **Updated Final Validity Date**: 2025-10-26
- **Contract Type**: Inexigibility 031/2023
- **Manager's Location**: ATI/DEGAT/GPLAT
- **Situation**: Active

Figure 8: Contract Summarization.

In our experiment, we addressed an initial set of questions that were able to produce a robust system that meets current user needs. However, exploring other questions in depth will allow us to enrich the metadata and the set of queries that extract information from traditional systems.

Finally, to consolidate the techniques developed to address our application, we envision that building a system in a different problem domain may shed light on limitations and the possible need for refinement or adaptation. Such future exploration will not only reinforce confidence in the implementation of these techniques in real-world scenarios, but also pave the way for their optimization and possible customization for specific domains, ultimately increasing the utility and impact of LLMs in enterprise applications.

REFERENCES

Chen, J., Lin, H., Han, X., and Sun, L. (2024). Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., and Li, Q. (2024). A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

Feng, Z., Feng, X., Zhao, D., Yang, M., and Qin, B. (2024). Retrieval-generation synergy augmented large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11661–11665. IEEE.

Gao, D., Wang, H., Li, Y., Sun, X., Qian, Y., Ding, B., and Zhou, J. (2023a). Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., and Wang, H. (2023b). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Giray, L. (2023). Prompt engineering with chatgpt: a guide for academic writers. *Annals of biomedical engineering*, 51(12):2629–2633.

Jeong, C. (2023). A study on the implementation of generative ai services using an enterprise data-based llm application architecture. *arXiv preprint arXiv:2309.01105*.

Jin, H., Huang, L., Cai, H., Yan, J., Li, B., and Chen, H. (2024). From llms to llm-based agents for software engineering: A survey of current, challenges and future. *arXiv preprint arXiv:2408.02479*.

Langchain (2024). Langchain agents documentation. https://python.langchain.com/v0.1/docs/use_cases/sql/agents/. Accessed: 2024-09-06.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Li, H., Su, Y., Cai, D., Wang, Y., and Liu, L. (2022). A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*.

Liu, A., Hu, X., Wen, L., and Yu, P. S. (2023). A comprehensive evaluation of chatgpt's zero-shot text-to-sql capability. *arXiv preprint arXiv:2303.13547*.

Mialon, G., Dessì, R., Lomeli, M., Nalmpantis, C., Pasunuru, R., Raileanu, R., Rozière, B., Schick, T., Dwivedi-Yu, J., Celikyilmaz, A., et al. (2023). Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

OpenAI (2023a). Chatgpt fine-tune description. https://help.openai.com/en/articles/6783457-what-is-chatgpt. Accessed: 2024-03-01.

OpenAI (2023b). Chatgpt prompt engineering. https://platform.openai.com/docs/guides/prompt-engineering. Accessed: 2024-04-01.

Pinheiro, J., Victorio, W., Nascimento, E., Seabra, A., Izquierdo, Y., Garcıa, G., Coelho, G., Lemos, M., Leme, L. A. P. P., Furtado, A., et al. (2023). On the construction of database interfaces based on large language models. In *Proceedings of the 19th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST*, pages 373–380. INSTICC, SciTePress.

Saeed, M., De Cao, N., and Papotti, P. (2023). Querying large language models with sql. *arXiv preprint arXiv:2304.00472*.

Seabra, A., Nepomuceno, J., Lago, L., Ruberg, N., and Lifschitz, S. (2024). Contrato360: uma aplicação de perguntas e respostas usando modelos de linguagem, documentos e bancos de dados. In *Anais do XXXIX Simpósio Brasileiro de Bancos de Dados*.

Singh, A., Ehtesham, A., Kumar, S., and Khoei, T. T. (2024). Enhancing ai systems with agentic workflows patterns in large language model. In *2024 IEEE World AI IoT Congress (AIIoT)*, pages 527–532. IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, M., Wang, M., Xu, X., Yang, L., Cai, D., and Yin, M. (2023). Unleashing chatgpt's power: A case study on optimizing information retrieval in flipped classrooms via prompt engineering. *IEEE Transactions on Learning Technologies*.

Wang, Z., Wang, Z., Le, L., Zheng, H. S., Mishra, S., Perot, V., Zhang, Y., Mattapalli, A., Taly, A., Shang, J., et al. (2024). Speculative rag: Enhancing retrieval augmented generation through drafting. *arXiv preprint arXiv:2407.08223*.

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., and Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023). React: Synergizing reasoning and actin in langage models. *arXiv preprint arXiv:2210.03629v3*.

Reviewing Machine Learning Techniques in Credit Card Fraud Detection

Ibtissam Medarhri[1][a], Mohamed Hosni[2][b], Mohamed Ettalhaoui[2], Zakaria Belhaj[1]
and Rabie Zine[3][c]

[1]*MMCS Research Team, LMAID, ENSMR-Rabat, Morocco*

[2]*MOSI Research Team, LM2S3, ENSAM, Moulay Ismail University of Meknes, Meknes, Morocco*

[3]*School of Science and Engineering, Al Akhawayn University in Ifrane, Ifrane, Morocco*

medarhri@enim.ac.ma, m.hosni@umi.ac.ma, m.ettalhaoui19@gmail.com, zakaria.belhaj@hps-worldwide.com, r.zine@aui.ma

Keywords: Credit Card Fraud, Machine Learning, Classification, Systematic Mapping Study.

Abstract: The growing use of credit cards for transactions has increased the risk of fraud, as fraudsters frequently attempt to exploit these transactions. Consequently, credit card companies need decision support systems that can automatically detect and manage fraudulent activities without human intervention, given the vast volume of daily transactions. Machine learning techniques have emerged as a powerful solution to address these challenges. This paper provides a comprehensive overview of the knowledge domain related to the application of machine learning techniques in combating credit card fraud. To achieve this, a review of published work in academic journals from 2018 to 2023 was conducted, encompassing 131 papers. The review classifies the studies based on eight key aspects: publication trends and venues, machine learning approaches and techniques, datasets, evaluation frameworks, balancing techniques, hyperparameter optimization, and tools used. The main findings reveal that the selected studies were published across various journal venues, employing both single and ensemble machine learning approaches. Decision trees were identified as the most frequently used technique. The studies utilized multiple datasets to build models for detecting credit card fraud and explored various preprocessing steps, including feature engineering (such as feature extraction, construction, and selection) and data balancing techniques. Python and its associated libraries were the most commonly used tools for implementing these models.

1 INTRODUCTION

The advancement of technology has significantly influenced the transition from traditional payment methods to online transactions (Mienye et al., 2023), (Taha and Malebary, 2020). Modern banking systems are now offering a wide array of payment options to enhance customer experience, including card payments, internet banking, and various e-services.

Globally, credit cards remain the most widely used payment method. According to the Nil Report (Report, 2023), there are 1,103 credit card issuers worldwide. In 2021, the combined purchase volume of the top 150 portfolios reached 12.695 trillion, reflecting a 9.4% increase compared to 2020.

While credit cards offer convenience for online purchases of goods and services, they also expose users to the risk of fraudulent transactions (Kim et al., 2019). In 2021 alone, 32.34 billion payment cards were compromised globally due to fraud (Report, 2023). Projections estimate that fraud-related losses will reach 408 billion over the next decade.

Current fraud detection systems predominantly rely on manually designed rules, which are often inefficient, subjective, and vulnerable to manipulation by fraudsters (Kim et al., 2019; Carcillo et al., 2018). As a result, there is a pressing need for automated detection systems. The growing adoption of electronic payment systems provides credit card issuers with extensive customer data, which can be leveraged to develop data-driven models that effectively detect fraud and minimize losses (Carcillo et al., 2018; Cheon et al., 2021; Pozzolo et al., 2018).

Machine Learning (ML) techniques have emerged

[a] https://orcid.org/0009-0003-0052-8702
[b] https://orcid.org/0000-0001-7336-4276
[c] https://orcid.org/0000-0002-0882-1327

Medarhri, I., Hosni, M., Ettalhaoui, M., Belhaj, Z. and Zine, R.
Reviewing Machine Learning Techniques in Credit Card Fraud Detection.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 179-187
ISBN: 978-989-758-716-0; ISSN: 2184-3228

as a powerful tool for tackling credit card fraud (Pozzolo et al., 2018; Leevy et al., 2023; Salekshahrezaee et al., 2023). ML models, once deployed, can efficiently process large volumes of transactions in real-time, assuming the appropriate infrastructure is in place. The success of ML techniques has been demonstrated across various domains.

This paper presents a systematic mapping study aimed at gaining insights into the use of ML techniques in developing decision support systems for detecting fraudulent credit card transactions. The study examines key aspects, including publication trends and venues, ML approaches and techniques, datasets used for constructing Credit Card Fraud (CCF) models, evaluation frameworks, preprocessing techniques, hyperparameter optimization methods, and tools employed in model development.

The structure of the paper is as follows: Section 2 outlines the research protocol used in the study. Section 3 presents and discusses the findings for each mapping question. Finally, Section 4 concludes the paper and offers suggestions for future research.

2 RESEARCH PROTOCOL

This study aims to consolidate existing research on the application of ML in developing automated systems for credit card fraud management. To accomplish this, a systematic mapping study was conducted following the methodology outlined by (Petersen et al., 2008), which has been widely adopted in various research fields, including software engineering (Hosni and Idri, 2018), medical informatics (Hosni et al., 2019), and urban flood hazard mapping (El baida et al., 2024). The mapping process consists of several steps, which are described in detail in the following subsections.

2.1 Mapping Questions

The goal of this review is to provide a comprehensive understanding of how ML techniques, particularly classification methods, are utilized in the development of CCF systems. To fulfill this objective, we formulated eight research questions (MQs), each designed to explore specific aspects of ML application in CCF. Table 1 lists these MQs along with the motivations behind each question.

2.2 Search Strategy

This step aims to identify candidate papers relevant to the topic of this study. The primary sources of papers are digital libraries that index research published by leading publishers worldwide. For this study, we selected the Scopus digital library as our primary source of candidate papers. The initial task was to construct a search string to be used as input for the Scopus search engine.

The search string was formulated based on the authors' expertise and knowledge. The search query used was:

TITLE-ABS-KEY((fraud OR ”Fraud detection” OR ”Fraud Analytics”) AND (”credit card” OR ”card payment*” OR ”Transaction Fraud”) AND (”Machine learning”))

The searches were conducted on metadata of titles, abstracts, and keywords of research works between the years 2018 and 2023. We have limited our search to articles in peer-reviewed journals. We set this limitation to ensure that the papers selected have undergone a satisfactory peer-reviewing process and hence command a high level of academic integrity and reliability.

2.3 Study Selection

The pool of candidate papers obtained through the Scopus search needed further filtering based on predefined inclusion and exclusion criteria. This step was crucial to ensure that only relevant papers addressing our MQs were included. To maintain accuracy, three researchers independently performed the filtering process. A paper was included if it met at least one inclusion criterion and none of the exclusion criteria. If the decision was unclear based on the metadata, the researcher proceeded to read the full paper. The inclusion and exclusion criteria were as follows:

Inclusion Criteria:

- Papers that specifically focus on building credit card fraud detection systems using ML techniques.
- Papers that aim to enhance existing ML techniques for credit card fraud detection.
- Papers that compare different ML techniques in the context of credit card fraud detection.

Exclusion Criteria:

- Papers not written in English.
- Papers that do not utilize ML techniques for credit card fraud detection.
- Papers that focus on detecting fraudulent transactions unrelated to credit cards.

Table 1: Mapping Questions and their Motivations.

Mapping Questions	Motivations
Which journal venues are the primary targets for the use of ML techniques in credit card fraud detection? And what is the frequency of publication has changed over time?	To identify the specific journal venues where research related to ML techniques in credit card fraud detection is being published and discover the publication trend over time.
What are the ML approaches used in credit card fraud detection? Additionally, which specific ML techniques are commonly utilized?	To identify the various types of ML techniques used in CCFD systems and provide an enumeration of specific ML techniques that have been adopted in building these systems.
What are the main datasets used in credit card fraud detection?	To identify the prevalent datasets that researchers rely on when developing and evaluating CCFD systems.
What are the performance frameworks used to build and assess the credit card fraud detection model?	To identify the evaluation methods used to build the CCFD systems and enumerate the performance indicators used to assess the built models.
What techniques are used to handle the balancing problem in credit card fraud detection?	To identify the techniques used to handle the balancing problem present in CCF datasets.
What feature engineering stages have been investigated in the context of credit card fraud detection? Additionally, what are the techniques that have been used in each of these stages?	To identify the feature engineering stages that have been treated in literature. Furthermore, enumerate the techniques used in each of the identified stages.
What are the optimization techniques used to fine-tune the hyperparameters of the ML techniques in credit card fraud detection systems?	To identify the optimization techniques used to fine-tune the hyperparameters of the ML techniques in credit card fraud detection.
What tools are used to build credit card fraud detection models?	To identify the tools used to build credit card fraud detection models.

2.4 Data Extraction and Synthesis

After selecting the papers relevant to our MQs, data extraction was performed independently by three researchers. The extracted data were systematically recorded in detailed forms, ensuring alignment with each MQ.

Following a comprehensive review of the extracted data, synthesis was conducted by summarizing and aggregating the findings for each MQ from all selected papers. Two synthesis methods were employed: narrative synthesis and the counting method, which allowed for the consistent tabulation of data in line with the MQs. Visualization tools, such as bar charts and pie charts, were used to present the aggregated data.

3 RESULTS AND DISCUSSION

This section presents and discusses the results obtained from the mapping study, organized according to the research questions listed in Table 1.

3.1 Results Overview

A total of 790 candidate papers were retrieved through the automatic search in the Scopus database using the search string specified in Section 2.2. The search was restricted in two ways: first, by time frame, including only papers published between 2018 and 2023, and second, by selecting only journal articles. The search was conducted on June 24, 2024. The primary reason for limiting the search to 2023 is to facilitate the replication of the search results, as the likelihood of additional papers being indexed for that year is minimal. In contrast, selecting an ongoing year could pose challenges since the indexing process for papers published within the same year may take time to complete.

Following the study selection process and the application of inclusion and exclusion criteria, 131 papers were selected. Relevant information was then extracted from these papers to address the research questions (MQs). It is worth noting that both the selection and data extraction processes were performed independently by three researchers. Additionally, not all 131 papers provided answers to all the research questions. Details of the selected papers and extracted data are available upon request.

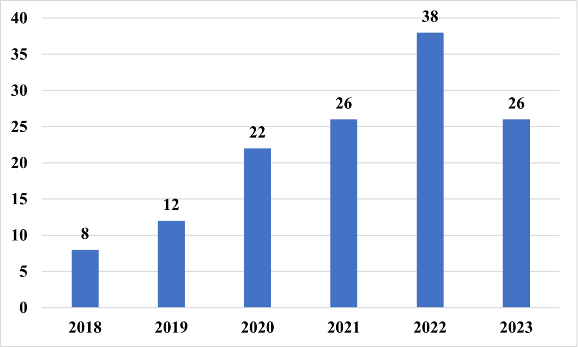


Figure 1: Publication Trends over Time.

3.2 Publication Venues and Trends (MQ1)

This review identified 86 different venues where the 131 selected papers were published. The IEEE Access journal had the highest number of publications, with 13 papers, followed by the Journal of Theoretical and Applied Information Technology with five publications and the Journal of Big Data with four. Seven journals published three papers each, while thirteen journals published two papers each. Additionally, 63 venues published only one paper each. Table 2 lists the main sources that published more than three papers.

Regarding publication trends, an upward trajectory in the number of publications was observed over time. It is important to note that only papers published in journals over the last five years were included in this review. The highest number of publications occurred in 2022, with 38 papers published across 28 different venues. IEEE Access led with four papers, followed by seven journals that published two papers each, while the remaining papers were distributed among 20 other journals, each publishing one paper. Figure 1 illustrates the publication trends over the search period.

3.3 Machine Learning: Approaches and Techniques (MQ2)

The objective of the MQ2 is to identify the most prevalent ML approaches used by researchers and to catalog the specific ML techniques employed in the selected studies.

Figure 2 illustrates the distribution of ML approaches used in the reviewed papers. The findings show that 39% of the selected studies (51 out of 131 papers) focused exclusively on single ML approaches. Meanwhile, 29% of the papers (39 out of 131) explored ensemble ML approaches alone. Notably, 32% of the papers (42 out of 131) investigated both single and ensemble approaches.

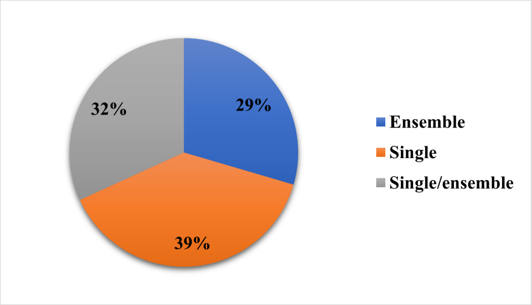


Figure 2: Publication Trends over Time.

Table 3 provides a comprehensive list of ML techniques that have been applied in developing decision support systems for detecting fraudulent credit card transactions (CCFD). The review identified 11 single classification techniques commonly explored in CCFD literature. Among these, Decision Tree (DT) was the most frequently used technique, appearing in 82 instances. Artificial Neural Networks (ANN) were investigated 67 times, while Regression techniques were utilized 47 times. Support Vector Machines (SVM) were employed in 32 instances. Notably, four techniques were each used only once.

Out of the 131 selected papers, 60 focused on investigating a single ML technique, and nine papers examined two ML techniques. The study that explored the highest number of ML techniques, totaling 31, was (Randhawa et al., 2018).

Ensemble methods were explored in 113 instances within the selected studies. The primary type of ensemble investigated was homogeneous, particularly the combination of a single base technique with a meta ensemble technique. Among the meta ensemble techniques, Boosting was the most commonly used, with XGBoost being the most extensively studied, appearing in 22 cases. Other meta ensemble techniques, such as Random Subspace and Bagging, were also explored. Additionally, heterogeneous ensembles were investigated in the selected studies (Baker, 2022).

3.4 Datasets Used (MQ3)

The construction of CCF models primarily relies on historical transaction data. This MQ aims to identify and catalog the datasets used in the selected studies for building CCF models. A total of 29 different datasets were identified across the selected studies. Table 4 lists the datasets that were utilized more than four times. Notably, the ”Credit Card Fraud Dataset,” containing 284,807 records, was the most frequently

Table 2: Publication Venues.

Journal	Number
IEEE Access	13
Journal of Theoretical and Applied Information Technology	5
Journal of Big Data	4
Multimedia Tools and Applications	3
International Journal of Intelligent Engineering and Systems	3
International Journal of Interactive Mobile Technologies	3
International Journal on Recent and Innovation Trends in Computing and Communication	3
Applied Sciences (Switzerland)	3
Electronics (Switzerland)	3
Mathematics	3

Table 3: ML techniques used in the Selected Studies.

Technique	Number
Ensemble	113
DT	82
ANN	67
Regression	47
SVM	32
KNN	25
NB	23
Rule	3
Independent component analysis	1
K-means	1
Local Outlier Factor	1
PCA	1

used, appearing in 85 out of the 131 selected papers. This dataset is publicly available on the Kaggle platform. Additionally, 16 papers employed more than one dataset, with the maximum number of datasets used in a single study being three, as reported in three papers (Arora et al., 2020; de Zarzà et al., 2023; Zhu et al., 2020).

The review also identified several studies that utilized private datasets, including those collected from organizations in China (Zheng et al., 2020; Li et al., 2021b), various European countries (Marco et al., 2022), and financial institutions in South Korea (Kim et al., 2019), among others. It is important to note that most of the datasets used suffered from the problem of data imbalance, where the fraudulent class was significantly underrepresented compared to the non-fraudulent class.

3.5 Evaluation Framework: Evaluation Methods and Performance Metrics (MQ4)

The MQ4 aims to identify the evaluation frameworks used to assess CCF models in the selected studies. It specifically focuses on the evaluation methods employed to develop CCF models and the performance indicators used to measure their predictive capabilities. The review identified 38 different performance criteria. Table 5 lists the nine performance indicators that were used more than ten times to evaluate the predictive capabilities of the ML techniques applied in the selected studies.

The most frequently used performance criterion was Sensitivity, appearing in 115 instances. Precision and Accuracy were used 95 and 89 times, respectively. The F1-score and ROC AUC were also commonly adopted, appearing 79 and 69 times, respectively. One of the selected studies utilized ten performance indicators to assess the proposed models. Notably, 121 out of the 131 selected papers employed more than one performance criterion to evaluate their models.

Regarding the validation techniques used in building the ML models, Table 6 lists the different validation approaches investigated in the literature along with their frequency of use. A total of four validation approaches were identified. The Holdout validation technique was the most frequently used, appearing in 61 research papers. It was followed by the K-fold cross-validation technique, employed in 42 papers. Among these, 10-fold cross-validation was the most common, appearing in 21 papers, followed by 5-fold cross-validation. Notably, four papers did not specify the number of folds used. The stratified K-fold and cross-validation techniques were each adopted in six papers. It is also worth noting that some papers did not provide details about the validation technique used to develop their models.

3.6 Handling Balancing Problem (MQ5)

This MQ aims to explore how the issue of imbalanced datasets is addressed in the selected studies. Imbalanced datasets, where the number of fraudulent trans-

Table 4: Datasets used in the selected studies.

Dataset	Number
Credit Card Fraud Detection Dataset	85
Default of Credit Card Clients Dataset	7
Vesta IEEE-CIS	5
Financial company in China	5
BankSim	4
Generated Dataset	4
Dataset emerges from Kaggle	4
cc Fraud dataset	4
UCSD-FICO dataset	4

Table 5: Performance indicators used in the selected studies.

Performance Criterion	Number
Sensitivity	115
Precision	95
Accuracy	89
F1-score	79
AUC	69
Specificity	41
MCC	17
AUC-PR	15
False Positive Rate	15

Table 6: Validation techniques used in the selected studies.

Validation techniques	K	Number
Stratified	5 fold	3
	10 fold	3
K-cross validation	K-fold	4
	2 fold	1
	3 fold	1
	4 fold	1
	5 fold	13
	10 fold	21
	15 fold	1
Holdout		61
Cross validation		6

actions is significantly lower than that of legitimate transactions, pose challenges in training ML models effectively. Table 7 lists the balancing techniques that were used more than three times to handle class imbalance in the selected papers. A total of 32 techniques were identified.

The most widely adopted technique was SMOTE (Synthetic Minority Over-sampling Technique), which was used in 24% of the selected papers (31 out of 131). Following SMOTE, Random Under Sampling, Under Sampling, and Over Sampling techniques were utilized in 12, 11, and 10 papers, respectively. It is worth noting that four papers addressed the class imbalance problem without explicitly specifying the technique used (Bakhtiari et al., 2023), (Sadgali et al., 2021; Rakhshaninejad et al., 2022; Trisanto, 2021).

Table 7: Imbalanced techniques used in the selected studies.

Technique	Number
SMOTE	31
Random Under sampling	12
Under Sampling	11
Over Sampling	10
SMOTE-Edited Nearest Neighbors	7
Random Oversampling	5
SMOTE-Tomek	4
Addressed	4
Borderline SMOTE	3
Near Miss	3

Table 8: Feature Engineering aspects investigated in the selected studies.

Aspect	Number
Extraction	16
Feature Importance	4
Feature selection	41
Feature Construction	1

3.7 Feature Engineering: Steps Investigated, and Techniques Used (MQ6)

This MQ aims to explore the feature engineering approaches investigated by researchers in the selected studies and to identify the techniques employed at each step. Out of the 131 selected papers, 44 considered feature engineering as a crucial preprocessing step. Four key aspects of feature engineering were examined: feature construction, extraction, importance, and selection.

Among these aspects, feature selection was the

Table 9: Feature Extraction, Construction and Importance techniques used in the selected studies.

Extraction		Construction		Importance	
PCA	10	Feature Construction	1	XGBoost	2
Auto Encoder	4			LightGBM	1
Convolutional Neural Network	1			Shapley addictive explanations	1
Linear Discriminant Analysis	1				

Table 10: Feature Selection Techniques investigated in the selected studies.

Filter Techniques		Wrapper Techniques	
Correlation	10	Genetic Algorithm	2
Information Gain	5	Recursive Feature Elimination	2
Random Forest	3	Stepwise	2
Chi2	1	Rock Hyrax Swarm Optimization	1
Correlation based Feature Selection	1	SVM Recursive Elimination	1
Decision Tree	1	Quantum Algorithm Feature Selection by Q-SVM	1
Degree Centrality	1		
Distance based Feature Selection	1		
Entropy	1		
Extra Tree Ensemble	1		
Gain Ration	1		
LASSO	1		
Mutual Information	1		
ReliefF	1		
Factorial Analysis of Mixed Data	1		
Rough set	1		
standardized murals with ANOVA F-values	1		

Table 11: Hyperparameters Optimization techniques used in the selected studies.

Optimization technique	Number
Grid Search	27
Adam	9
Given	7
Bayesian	4
Genetic Algorithm	3
Particle Swarm Optimization	3
Randomized Search CV	2
Default Parameters	2
Differential Evolution Algorithm	2
Firefly Algorithm	2

most extensively studied, appearing in 41 experiments. Feature extraction was explored in 16 experiments, as detailed in Table 8.

Four feature extraction techniques were identified, as listed in Table 9. The most commonly used technique was Principal Component Analysis (PCA), which appeared in 10 instances. This was followed by the Auto Encoder technique, used four times. Regarding feature construction, only one study specifically focused on this aspect, utilizing both domain knowledge and statistical methods to create new features (Wu et al., 2019). For feature importance, three techniques were employed: XGBoost was used twice, while LightGBM and the Shapley Additive Explanations (SHAP) model were each used once.

Table 12: ML tools used in the selected papers.

Tool	Number
Python	71
Weka	11
MATLAB	4
Java	4
R	3
LibSVM	1
Orange	1
RapidMiner	1
SAS E-miner	1

Regarding feature selection techniques, as detailed in Table 10, this review identified two main categories: filter and wrapper techniques. Among the filter techniques, 17 different methods were used across the experiments in the selected papers. The most frequently employed filter technique was the correlation coefficient, such as Pearson correlation, which was used in 10 experiments. Information Gain and Random Forest were utilized in 5 and 3 experiments, respectively, while the remaining 14 techniques were each explored once.

For wrapper techniques, six methods were identified in the selected studies. The Genetic Algorithm, Recursive Feature Elimination, and Stepwise techniques were each explored twice, while the other three techniques were used once.

3.8 Hyperparameters Optimization Techniques (MQ7)

Hyperparameter optimization is crucial for enhancing the performance and generalization ability of ML models. This question aims to identify the hyperparameter optimization techniques employed in the selected studies.

In this review, 20 different optimization techniques were identified, used to fine-tune the hyperparameters of ML models. These techniques are listed in Table 11. Notably, Grid Search was the most frequently adopted optimization method, appearing in 27 research papers. The Adam optimizer was explored in 9 papers. Additionally, seven papers explicitly listed the parameter values of their employed ML techniques, while two papers used the default parameters provided by the tools used.

It is important to highlight that only 57 out of the 131 selected papers considered the hyperparameter optimization step. Moreover, seven studies employed multiple optimization techniques (Zhu et al., 2020; Li et al., 2021b; Tayebi and El, 2022; Li et al., 2021a; Yara et al., 2020; Grossi et al., 2022; Sharma et al., 2021). The study with the most comprehensive exploration of optimization techniques investigated seven different methods (Tayebi and El, 2022).

3.9 ML Tools (MQ8)

This question aims to identify the tools used to build decision support systems for detecting fraudulent credit card transactions. Table 12 provides a list of the nine identified tools.

The Python programming language was the most widely used, appearing in 71 papers. The Weka tool was utilized in 11 papers, while MATLAB and Java were each employed in four papers. Additionally, four tools were used in only one paper each.

The identified tools can be categorized into two groups: those with a **user interface**, such as RapidMiner, Orange, SAS E-miner, and Weka, and those that provide a **programming environment**, such as MATLAB, Java, R, Python, and the Weka API.

4 CONCLUSIONS AND FUTURE WORK

This paper presents a systematic mapping study that structures the body of knowledge on the use of ML techniques in developing decision support systems for detecting fraudulent credit card transactions. The study reviewed papers published in journal venues indexed in the Scopus database between 2018 and 2023. After applying the study selection process, including specific inclusion and exclusion criteria, 131 papers were selected to address eight mapping questions. The main findings related to each mapping question, as outlined in Table 1, are summarized below:

- The selected papers were published across 86 different journal venues.
- Both single ML approaches and ensemble approaches were investigated, with single ML approaches being the most prevalent.
- A total of 29 different datasets were utilized to build credit card fraud detection systems.
- Various performance indicators were used to evaluate the predictive capabilities of the models, with the Holdout validation technique being the most frequently employed.
- A total of 32 balancing techniques were identified, with SMOTE being the most commonly used method.
- Feature extraction, construction, and selection steps were explored in the selected studies.
- Only 27 studies optimized the hyperparameter settings of the ML techniques used.
- Nine tools were identified for building credit card fraud detection systems in the selected studies.

Future research directions could include exploring the construction and effectiveness of ensemble techniques in credit card fraud detection systems. Another promising area of investigation is identifying the most effective ML models for distinguishing between fraudulent and legitimate transactions, which could be systematically explored through a comprehensive literature review.

REFERENCES

Arora, V., Leekha, R. S., Lee, K., and Kataria, A. (2020). Facilitating user authorization from imbalanced data logs of credit cards using artificial intelligence. *Mobile Information Systems*, 2020(1):8885269.

Baker, M. R. (2022). Ensemble learning with supervised machine learning models to predict credit card fraud transactions.

Bakhtiari, S., Nasiri, Z., and Vahidi, J. (2023). Credit card fraud detection using ensemble data mining methods. *Multimedia Tools and Applications*, 82(19):29057–29075.

Carcillo, F., Dal Pozzolo, A., Le Borgne, Y.-A., Caelen, O., Mazzer, Y., and Bontempi, G. (2018). Scarff: a scalable framework for streaming credit card fraud detection with spark. *Information fusion*, 41:182–194.

Cheon, M.-j., Lee, D., Joo, H. S., and Lee, O. (2021). Deep learning based hybrid approach of detecting fraudulent transactions. *Journal of Theoretical and Applied Information Technology*, 99(16):4044–4054.

de Zarzà, I., de Curtò, J., and Calafate, C. T. (2023). Optimizing neural networks for imbalanced data. *Electronics*, 12(12):2674.

El baida, M., Hosni, M., Boushaba, F., Chourak, M., et al. (2024). A systematic literature review on classification machine learning for urban flood hazard mapping. *Water Resources Management*, pages 1–42.

Grossi, M., Ibrahim, N., Radescu, V., Loredo, R., Voigt, K., and Altrock, C. V. O. N. (2022). Mixed quantum – classical method for fraud detection with quantum feature selection. *IEEE Trans. Quantum Eng.*, 3(October):1–12.

Hosni, M., Carrillo-de Gea, J. M., Idri, A., Fernández-Alemán, J. L., and García-Berná, J. A. (2019). Using ensemble classification methods in lung cancer disease. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1367–1370. IEEE.

Hosni, M. and Idri, A. (2018). Software development effort estimation using feature selection techniques. In *New trends in intelligent software methodologies, tools and techniques*, pages 439–452. IOS Press.

Kim, E. et al. (2019). Champion-challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning. *Expert Syst. Appl.*, 128:214–224.

Leevy, J. L., Johnson, J. M., Hancock, J., and Khoshgoftaar, T. M. (2023). Threshold optimization and random undersampling for imbalanced credit card data. *J. Big Data*.

Li, C., Ding, N., Zhai, Y., and Dong, H. (2021a). Comparative study on credit card fraud detection based on different support vector machines. *Intelligent Data Analysis*, 25(1):105–119.

Li, Z., Huang, M., Liu, G., and Jiang, C. (2021b). A hybrid method with dynamic weighted entropy for handling the problem of class imbalance with overlap in credit card fraud detection. *Expert Syst. Appl.*, 175(February):114750.

Marco, G. et al. (2022). The role of diversity and ensemble learning in credit card fraud detection. *Adv. Data Anal. Classif.*

Mienye, I. D., Sun, Y., and Member, S. (2023). A deep learning ensemble with data resampling for credit card fraud detection. *IEEE Access*, 11(February):30628–30638.

Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering. In *12Th International Conference on Evaluation and Assessment in Software Engineering*, page 10.

Pozzolo, A. D., Boracchi, G., Caelen, O., and Alippi, C. (2018). Credit card fraud detection: A realistic modeling and a novel learning strategy. *IEEE Trans. Neural Networks Learn. Syst.*, 29(8):3784–3797.

Rakhshaninejad, M., Fathian, M., Amiri, B., and Yazdanjue, N. (2022). An ensemble-based credit card fraud detection algorithm using an efficient voting strategy. *The Computer Journal*, 65(8):1998–2015.

Randhawa, K., Loo, C. H. U. K., and Member, S. (2018). Credit card fraud detection using adaboost and majority voting. *IEEE Access*, 6:14277–14284.

Report, N. (October 2023). The world's top card issuers and merchant acquirers.

Sadgali, I., Sael, N., and Benabbou, F. (2021). Human behavior scoring in credit card fraud detection. *IAES International Journal of Artificial Intelligence*, 10(3):698.

Salekshahrezaee, Z., Leevy, J. L., and Khoshgoftaar, T. M. (2023). The effect of feature extraction and data sampling on credit card fraud detection. *J. Big Data*.

Sharma, P., Banerjee, S., Tiwari, D., and Patni, J. C. (2021). Machine learning model for credit card fraud detection- a comparative analysis. *The International Arab Journal of Information Technology*, 18(6):789–796.

Taha, A. A. and Malebary, S. J. (2020). An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE access*, 8:25579–25587.

Tayebi, M. and El, S. (2022). Performance analysis of metaheuristics based hyperparameters optimization for fraud transactions detection. *Evol. Intell.*, page 0123456789.

Trisanto, D. (2021). Modified focal loss in imbalanced xgboost for credit card fraud detection. *Int. J. Ind. Eng. Syst.*, 14(4):350–358.

Wu, Y., Xu, Y., and Li, J. (2019). Feature construction for fraudulent credit card cash-out detection. *Decis. Support Syst.*, page 113155.

Yara, A., Albatul, A., and A, R. M. (2020). A financial fraud detection model based on lstm deep learning technique. *J. Appl. Secur. Res.*, 0(0):1–19.

Zheng, L., Liu, G., Yan, C., Jiang, C., Zhou, M., and Li, M. (2020). Improved tradaboost and its application to transaction fraud detection. *IEEE Transactions on Computational Social Systems*, 7(5):1304–1316.

Zhu, H., Liu, G., Zhou, M., Xie, Y., and Abusorrah, A. (2020). Optimizing weighted extreme learning machines for imbalanced classification and application to credit card fraud detection. *Neurocomputing*, 407:50–62.

SHORT PAPERS

An Index Bucketing Framework to Support Data Manipulation and Extraction of Nested Data Structures

Jeffrey Myers II and Yaser Mowafi
School of Engineering and Applied Sciences, Western Kentucky University, Bowling Green, Kentucky, U.S.A.
jeffrey.myers648@topper.wku.edu, yaser.mowafi@wku.edu

Keywords: Nested Data Structures, Irregular Schema, Skewed Distribution, Information Loss, Duplication Explosion.

Abstract: Handling nested data collections in large-scale distributed data structures poses considerable challenges in query processing, often resulting in substantial costs and error susceptibility. These challenges are exacerbated in scenarios involving skewed, nested data with irregular inner data collections. Processing such data demands costly operations, leading to extensive data duplication and imposing challenges in ensuring balanced distribution across partitions—consequently impeding performance and scalability. This work introduces an index bucketing framework that amalgamates upfront computations with data manipulation techniques, specifically focusing on flattening procedures. The framework resembles principles from the bucket spreading strategy, a parallel hash join method that aims to mitigate adverse implications of data duplication and information loss, while effectively addressing both skewed and irregularly nested structures. The efficacy of the proposed framework is assessed through evaluations conducted on prominent question-answering datasets such as QuAC and NewsQA, comparing its performance against the Pandas Python API and recursive, iterative flattening implementations.

1 INTRODUCTION

The widespread rise in big data analytics has spurred interest in query processing systems that allow for performing complex analytical tasks over distributed data structures of arbitrary data types—including nested data collections. Implementations of languages integrated with query systems are evidenced in large-scale distributed data processing platforms (*Apache Flink. http://flink.apache.org/*; *Apache Spark, http://spark.apache.org/*; *Pandas Python, https://pandas.pydata.org/*). Despite their vaunted support of nested data, these systems provide no direct processing for nested data manipulation over different distributed collections, whose values may themselves be collections.

To stave off this penalty, declarative querying APIs have been employed for integrating data query languages with host programming languages' data processing features using higher-order operations—i.e., Google's F1 query (Samwel et al., 2018).

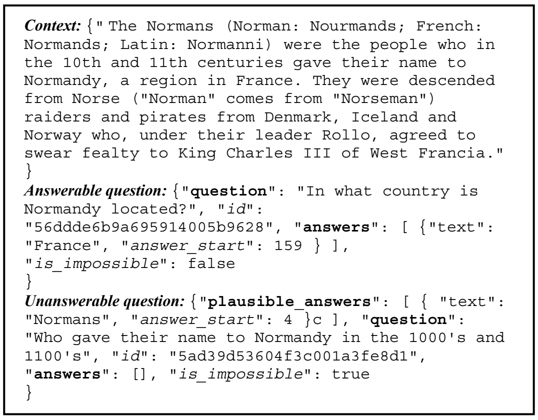
Apart from their intricate and computational challenges, unnesting and manipulating data collections inherently entail the generation of large amounts of duplicated data and redundant computations that significantly degrade the run-time performance of these techniques. These challenges are exacerbated for skewed nested data with irregular inner data collections – where loading unnecessarily large amounts of data to enforce balancing across partitions can lead to performance deficiency and error susceptibility (Diestelkämper et al., 2021; Smith, 2021).

To illustrate these challenges, consider the reading comprehension question-answering dataset. The dataset consists of questions where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable with an indeterminant plausible answer (Fig. 1).

The dataset articulates a schema that can be structured within the following relational database tables: Sources (src), Questions (qst), Answers (ans), and Plausible Answers (pls). For the sake of clarity and brevity, the number of records within a table is denoted as n. Table 1 comprises source records featuring *id* and *context* fields. The *id* field encompasses incremental integers (INC), $i = 1, ..., n,$ while *context* (ctx) stores textual excerpts (STR), extracted from source document paragraphs. Table 2 incorporates *id*, *text* (txt), and *i* fields. The *id* field embodies incremental integers (INC), $j = 1, ..., n,$

Myers II, J. and Mowafi, Y.
An Index Bucketing Framework to Support Data Manipulation and Extraction of Nested Data Structures.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 191-199
ISBN: 978-989-758-716-0; ISSN: 2184-3228



Context: {" The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France. They were descended from Norse ("Norman" comes from "Norseman") raiders and pirates from Denmark, Iceland and Norway who, under their leader Rollo, agreed to swear fealty to King Charles III of West Francia."
}
Answerable question: {"**question**": "In what country is Normandy located?", "*id*": "56ddde6b9a695914005b9628", "**answers**": [{"text": "France", "*answer_start*": 159 }], "*is_impossible*": false
}
Unanswerable question: {"**plausible_answers**": [{ "text": "Normans", "*answer_start*": 4 }c], "**question**": "Who gave their name to Normandy in the 1000's and 1100's", "*id*": "5ad39d53604f3c001a3fe8d1", "**answers**": [], "*is_impossible*": true
}

Figure 1: Question-answering dataset structure of answerable and unanswerable plausible answers.

housing textual representations (STR) of questions. The *i* field functions as a foreign key (FK) referencing records in Table 1. Table 3 encompasses the *id*, answer *start* (srt), answer *end,* and *j* fields. The *id* field spans incremental integers (INC), *k = 1, ..., n*, while *start* (srt) and *end* signify the index positions of answers within the context of the related resource dataset. The *j* field acts as a foreign key (FK) referring to records in Table 2. Table 4 accommodates plausible yet indeterminate answers to questions, acknowledging instances, where a definitive answer might be unattainable. Table 4 augments the dataset by mirroring fields akin to those in Table 3, with incremental integers (INC), *l = 1, ..., n,* representing its incremental *id*. The *j* field acts as a foreign key (FK) referring to records in Table 2.

Table 1: Sources (src).

id (i)	Context (ctx)
INC	STR

Table 2: Questions (qst).

id (j)	text (txt)	i
INC	STR	FK

Table 3: Answers (ans).

id (k)	start (srt)	end	j
INC	INT	INT	FK

Table 4: Plausible Answers (pls).

id (l)	Start (srt)	end	j
INC	INT	INT	FK

These interconnected tables establish a nested relationship structure, delineating diverse data distribution patterns, while exemplifying irregular schema through the inclusion of Table 4. To further visualize the nested data structure portrayed by the relational Tables 1, 2, 3, and 4, consider the tree representations in Fig. 2 of an *irregular* nested structure with a given source (src), *i* of a *context* (ctx), and *j* questions (qst). A given question (qst), *j* of a *text* (txt) may have *k* answers (ans) or *l* plausible answers (pls) or both, where each answer (ans), *k* or plausible answer (pls), *l* has a *start* (srt) and *end*.

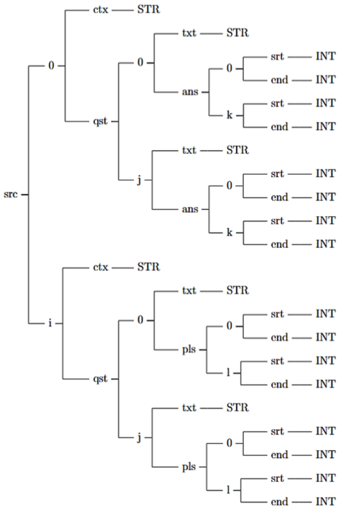


Figure 2: Irregular question answering nested structure.

With the tree-based representations, it becomes evident that sources might lack associated questions, and questions might encompass answers, plausible answers, both, or neither. This variability extends to the varying counts of answers and plausible answers within each question, along with fluctuations in the number of questions within each source. Such variability typifies an irregular nested structure marked by skewed data distribution. Next, we present the challenges associated with manipulating and information extraction of these nested data structures.

2 CHALLENGES

2.1 Duplication Explosion

Duplication explosion is a phenomenon characterized by an overwhelming proliferation of duplicated data during the flattening process. As the term implies, this explosion also known as a data avalanche or data storm results in an excessive replication of data, aka N + 1 query problems or avalanches (Grust et al., 2010). This often leads to severe memory utilization issues and potential system failures, especially when handling extensive datasets. Current flattening solutions, primarily relying on recursion, fail to mitigate the adverse effects of this rampant data duplication.

2.2 Skewed Distribution

Another hurdle to overcome in nested data collections is unbalanced distributions of information. When flattening such data, ensuring that each flattened instance contains all requisite keys introduces a problem akin to duplication explosion. However, in this case, missing keys necessitate filling with null values, requiring comprehensive parsing of the dataset to gather all keys. The challenge lies in distributing these missing keys throughout the flattened data. Strategies may involve parsing before flattening, allowing simultaneous filling, or conducting a secondary traversal after flattening, although the former, while superior, present implementation complexities (Smith et al., 2020).

2.3 Irregular Schema

Here, disparate data collections within the dataset may contain entirely different keys at the same nesting level, significantly complicating parsing and filling algorithms. Akin to skewed distribution, solving irregular schema involves filling in missing keys throughout the dataset. However, it presents an even more intricate challenge, where the endeavor to enforce balance across partitions escalates runtime inefficiencies and scalability limitations, exacerbating disk spillage and load imbalance issues (Smith et al., 2021).

2.4 Information Loss

The final challenge, information loss, poses some concern, describing the repercussions of processing nested data structures. The flattened data loses crucial information required for reconstructing the original nested form. Without incorporating metadata into the flattened dataset, reconstructing the initial hierarchical structure becomes unfeasible (Diestelkämper, 2021). Reverting to the original data necessitates reloading the data file or maintaining a copy of the original data, which could be time-consuming and can proliferate memory utilization problems, especially with large datasets.

To address these challenges, we propose a novel framework, which we refer to as index bucketing. The basis of our framework resembles principles from the bucket spreading strategy, a parallel hash join method that allows for handling irregular data distribution for relational database systems by utilizing bucketing mechanisms. The strategy aims to evenly distribute the load among processes, always fully exploiting (Kitsuregawa & Ogawa, 1990). Index bucketing draws on applying these principles to a tree-based nesting by mapping the data indexes corresponding to their respective hierarchical structure within the original data.

3 FRAMEWORK

This section delineates a concise implementation of the index bucketing framework provided by the following algorithmic classes (Algorithms 1, 2, 3, 4, 5, 6). The framework is designed to address the aforementioned challenges, accentuating the framework's prowess in surmounting the diverse challenges encountered in nested data structure manipulation.

3.1 Base Node – Algorithm 1

As a foundational base class, the NODE class serves as the common blueprint inherited by the LEAF, BRANCH, and ROOT classes within the index bucketing framework. The NODE class lays out the essential structural elements shared across all inheriting classes:

- NODE – This is the shared base constructor for all inheriting node classes and is responsible for setting the shared node attributes – *kdx*, *value*, *level*, and *parent*. The *kdx* attribute is a key or index value used for gathering index and key paths. The *value* attribute contains a collection of child NODE types or serves as a BASE value type for leaves. The *level* attribute is used to determine the depth of the node within the tree. The *parent* attribute is used to establish a link to the node's parent node.
- IBUCKET – By collecting a set of index paths, each aligning with the maximum *depth* of the nested data tree, this method is responsible for gathering the index bucket.

Algorithm 1: Node Class.

```
class NODE
    function NODE(kdx, value, level, parent)
        this.kdx ← kdx
        this.value ← value
        this.level ← level
        this.parent ← parent
    function IPATH
        return this.parent.IPATH()
    function KPATH
        return this.parent.KPATH()
    function IBUCKET(depth)
        ibucket ← SET[IPATH]()
        for all child ∈ this.value do
            ibucket.UPDATE(child.IBUCKET(depth))
        return ARR(ibucket).SORT()
    function FLATTEN(ipath)
```

This standardized class structure established by the NODE class ensures coherence and consistency in defining and organizing nodes across the index bucketing framework.

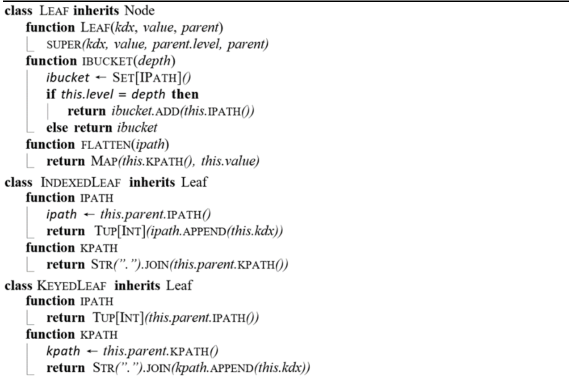
3.2 Leaf Node – Algorithm 2

Within the framework, the LEAF class, along with its inheriting classes – INDEXEDLEAF and KEYEDLEAF –

fulfill the role of nodes encapsulating the terminus of nested data structures. These classes define essential functionalities pivotal to handling leaf nodes within the index bucketing framework:

- LEAF – Rather than directly receiving the level parameter argument, the LEAF constructor derives its level value from the parent node, ensuring hierarchical consistency within the tree structure.
- IBUCKET – This method accepts the maximum depth value of the tree as a parameter argument. It validates whether the depth value matches its level, subsequently returning its index path enclosed in an index bucket set object if true; otherwise, an empty index bucket set object is returned. Employing a bottom-to-top algorithm, this method is invoked by non-leaf nodes to update and collate their child leaf node value fields into a set collection.
- FLATTEN – Disregarding the index path parameter argument, *ipath*, when invoked by the leaf nodes corresponding parent, this method returns a new mapping of the leaf node's key path and *value*, adhering to a top-to-bottom calling sequence and resulting in a bottom-to-top return sequence.
- IPATH & KPATH – Defined in the INDEXEDLEAF and KEYEDLEAF classes which serve to differentiate leaves based on their indexing nature: indexed with integers or keyed with strings during tree initialization, these class methods manage bottom-to-top index paths or key paths by integrating the leaf node's *kdx* field along with its parent's index or key path, respectively. In cases where index paths are gathered, the leaf node converts arrays of index values into tuples of the same size.

Algorithm 2: Leaf Classes.



```
class LEAF inherits Node
    function LEAF(kdx, value, parent)
        SUPER(kdx, value, parent.level, parent)
    function IBUCKET(depth)
        ibucket ← SET[IPATH]()
        if this.level = depth then
            return ibucket.ADD(this.IPATH())
        else return ibucket
    function FLATTEN(ipath)
        return MAP(this.KPATH(), this.value)
class INDEXEDLEAF inherits Leaf
    function IPATH
        ipath ← this.parent.IPATH()
        return TUP[INT](ipath.APPEND(this.kdx))
    function KPATH
        return STR(".").JOIN(this.parent.KPATH())
class KEYEDLEAF inherits Leaf
    function IPATH
        return TUP[INT](this.parent.IPATH())
    function KPATH
        kpath ← this.parent.KPATH()
        return STR(".").JOIN(kpath.APPEND(this.kdx))
```

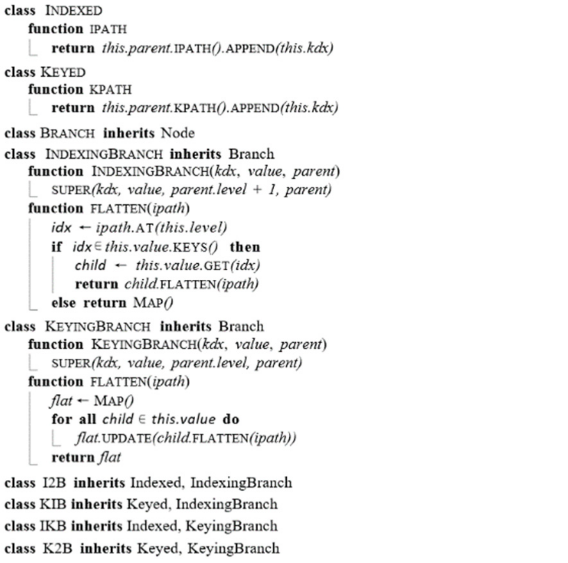
By segregating leaves between indexed and keyed types during tree initialization, the classes circumvent the need for conditional evaluations. This strategic segregation bolsters performance and scalability, especially in managing larger datasets.

3.3 Branch Node – Algorithm 3

The BRANCH class integrates into various specialized nodes, including I2B, KIB, IKB, and K2B which are defined by inheriting combinations of INDEXED and KEYED classes with INDEXINGBRANCH and KEYINGBRANCH classes.

- INDEXED – The INDEXED class encapsulates nodes indexed with integers, defining the IPATH method to append the current node's index value to the parent's index path.
- KEYED – The KEYED class represents nodes keyed with strings, providing the KPATH method to append the node's key value to the parent's key path.
- INDEXINGBRANCH – The INDEXINGBRANCH class inherits from BRANCH, designed for indexed branches. Its constructor sets attributes based on the provided values and parent node, and the FLATTEN method retrieves the corresponding child node based on the index path.
- KEYINGBRANCH – The KEYINGBRANCH class, also extending BRANCH, targets keyed branches. Its constructor initializes attributes, and the FLATTEN method iterates through child nodes, updating a map with their flattened results.
- I2B – The I2B class combines INDEXED and INDEXINGBRANCH functionalities.
- KIB – The KIB class combines KEYED and INDEXINGBRANCH functionalities.
- IKB – The IKB class combines INDEXED and KEYINGBRANCH functionalities.
- K2B – The K2B class combines KEYED and KEYINGBRANCH functionalities.

Algorithm 3: Branch Classes.



```
class INDEXED
    function IPATH
        return this.parent.IPATH().APPEND(this.kdx)
class KEYED
    function KPATH
        return this.parent.KPATH().APPEND(this.kdx)
class BRANCH inherits Node
class INDEXINGBRANCH inherits Branch
    function INDEXINGBRANCH(kdx, value, parent)
        SUPER(kdx, value, parent.level + 1, parent)
    function FLATTEN(ipath)
        idx ← ipath.AT(this.level)
        if idx ∈ this.value.KEYS() then
            child ← this.value.GET(idx)
            return child.FLATTEN(ipath)
        else return MAP()
class KEYINGBRANCH inherits Branch
    function KEYINGBRANCH(kdx, value, parent)
        SUPER(kdx, value, parent.level, parent)
    function FLATTEN(ipath)
        flat ← MAP()
        for all child ∈ this.value do
            flat.UPDATE(child.FLATTEN(ipath))
        return flat
class I2B inherits Indexed, IndexingBranch
class KIB inherits Keyed, IndexingBranch
class IKB inherits Indexed, KeyingBranch
class K2B inherits Keyed, KeyingBranch
```

These specialized branch classes cater to different scenarios, providing distinct methods for handling various types of nested data collections. Each class offers unique functionalities for efficient execution, minimizing conditional evaluations during execution.

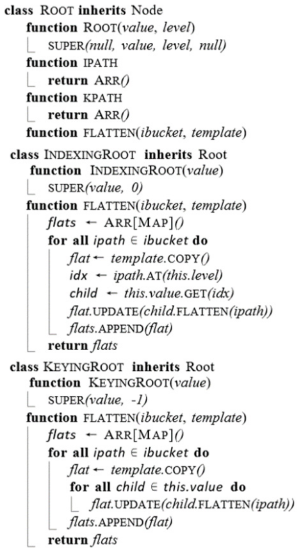
3.4 Root Node – Algorithm 4

The Root class, and its inheriting classes, mark the starting point of top-to-bottom processes and the conclusion of bottom-to-top processes within the index bucketing framework.

- ROOT – Inheriting from the Node class, the base Root class undergoes constructor modification, accepting solely *value* and *level* parameters. Root nodes lack *kdx* or *parent* attributes. Consequently, both the IPATH and KPATH methods return new empty arrays. Notably, the FLATTEN method's signature undergoes modification, now accepting the index bucket, *ibucket*, and flat *template* as parameters, and returning an array of flat mappings rather than a single mapping as seen in prior class definitions.
- INDEXINGROOT – This class inherits the base ROOT class, but its constructor configures the root node's level to 0 during instantiation, aligning its child node calling behavior with that of INDEXINGBRANCH nodes. Its FLATTEN method iterates over the index bucket, IBUCKET, dispatching each index path to the appropriate child nodes for further processing. An array of flat mappings, each of which is applied to a copy of the flat template, is gathered from the child nodes and is returned.
- KEYINGROOT – Also inheriting from the base ROOT class, the KEYINGROOT class sets its level to -1 within the constructor since its child-calling behavior does not utilize the indexes from the index bucket. Its FLATTEN method operates by passing index paths, IPATH, from the index bucket, IBUCKET, to its child nodes for further processing. Likewise, an array of flat mappings, each of which is applied to a copy of the flat *template*, is gathered from the child nodes and is returned.

By distinguishing between KEYINGROOT and INDEXINGROOT nodes, the tree's root node ensures that subsequent *level* attributes are set appropriately during initialization and the index bucket is distributed accordingly during execution.

Algorithm 4: Root Classes.



```
class ROOT inherits Node
    function ROOT(value, level)
        SUPER(null, value, level, null)
    function IPATH
        return ARR()
    function KPATH
        return ARR()
    function FLATTEN(ibucket, template)
class INDEXINGROOT inherits Root
    function INDEXINGROOT(value)
        SUPER(value, 0)
    function FLATTEN(ibucket, template)
        flats ← ARR[MAP]()
        for all ipath ∈ ibucket do
            flat ← template.COPY()
            idx ← ipath.AT(this.level)
            child ← this.value.GET(idx)
            flat.UPDATE(child.FLATTEN(ipath))
            flats.APPEND(flat)
        return flats
class KEYINGROOT inherits Root
    function KEYINGROOT(value)
        SUPER(value, -1)
    function FLATTEN(ibucket, template)
        flats ← ARR[MAP]()
        for all ipath ∈ ibucket do
            flat ← template.COPY()
            for all child ∈ this.value do
                flat.UPDATE(child.FLATTEN(ipath))
            flats.APPEND(flat)
        return flats
```

3.5 Tree Structure – Algorithm 5

The Tree class serves as the foundational structure to organize the nested dataset for the execution of the index bucketing algorithm. In the constructor, the initialization commences by setting the depth field to 0 and creating an empty set object for the key bucket, kbucket. These fields are then used to analyze the data parameter's nested structure while the tree itself is constructed and stored within the tree field which acts as a reference to the root node. Next, the algorithm gathers the index bucket, ibucket. Additionally, it constructs the template by iterating through the key bucket, compiling all key paths into a mapping with initial null values for each key path. This flat template formation streamlines the subsequent data organization process.

- FLATTEN – To facilitate the flattening process, the Tree class defines its own FLATTEN method. This method initiates the root node's FLATTEN method, passing along the index bucket and flat template.
- LEAF – The LEAF method initializes and returns the relevant LEAF class node. Additionally, the LEAF method identifies the maximum depth of the tree and aggregates key paths into the key bucket.
- BRANCH – The BRANCH method initializes and returns the relevant BRANCH class node. If the collection passed as *data* is empty, then the BRANCH method delegates the parameter arguments to the LEAF method with null passed for the *data* parameter's argument. Otherwise, respective to the nested data types, the BRANCH method directs nested information to either another BRANCH method call or a LEAF method call.

- ROOT – The ROOT method initializes and returns the relevant ROOT class node. The ROOT method returns null when the *data* parameter is an empty collection, indicating that no data is present. Otherwise, respective to the nested data types, the ROOT method directs nested information to either BRANCH method call or a LEAF method call.

Algorithm 5: Tree Class.

```
class TREE
    function TREE(data)
        this.depth ← 0
        this.kbucket ← SET()
        this.tree ← this.ROOT(data)
        this.ibucket ← this.tree.IBUCKET(this.depth)
        this.template ← MAP()
        for all kpath ∈ this.kbucket do
            this.template.UPDATE(MAP(kpath, null))
    function FLATTEN
        return this.tree.FLATTEN(this.ibucket, this.template)
    function LEAF(kdx, data, parent)
        if TYPE(kdx) = INT then
            leaf ← INDEXEDLEAF(kdx, data, parent)
        else leaf ← KEYEDLEAF(kdx, data, parent)
        this.depth ← MAX(this.depth, leaf.level)
        this.kbucket.ADD(leaf.KPATH())
        return leaf
    function BRANCH(kdx, data, parent)
        if LENGTH(data) ⩽ 0 then
            return this.LEAF(kdx, null, parent)
        if TYPE(data)'= MAP then
            data ← ENUMERATE(data)
            if TYPE(kdx) = STR then
                branch ← KIB(kdx, MAP(), parent)
            else branch ← I2B(kdx, MAP(), parent)
        else if TYPE(kdx) = STR then
            branch ← K2B(kdx, MAP(), parent)
        else branch ← IKB(kdx, MAP(), parent)
        for all kdx, value ∈ data.ITEMS() do
            if TYPE(value) = ITER then
                node ← this.BRANCH(kdx, value, branch)
            else node ← this.LEAF(kdx, value, branch)
            branch.value.UPDATE(kdx, node)
        return branch
    function ROOT(data)
        if LENGTH(data) ⩽ 0 then
            return null
        if TYPE(data) '= MAP then
            data ← ENUMERATE(data)
            root ← INDEXINGROOT(MAP())
        else root ← KEYINGROOT(MAP())
        for all kdx, value ∈ data.ITEMS() do
            if TYPE(value) = ITER then
                node ← this.BRANCH(kdx, value, branch)
            else node ← this.LEAF(kdx, value, branch)
            root.value.UPDATE(kdx, node)
        return root
```

3.6 Generator Alternative – Algorithm 6

To allow for the implementation flexibility of the index bucketing algorithm, ROOT and TREE class definitions are modified to transform the framework into a generator capable of delivering flattened data *incrementally* rather than in a single instance.

Instead of the ROOT node managing the index bucket within its FLATTEN method, this responsibility is shifted to the TREE class's FLATTEN method. Introducing a *count* field, initialized at 0, enables the tracking of index bucket progress. When the *count* reaches the end of the index bucket, it is reset to 0, and null is returned to signal completion. This generator-style implementation offers a controllable method to alleviate the adverse effects of duplication explosion which can otherwise overload memory usage. The adaptability of index bucketing as an algorithm allows for diverse implementations, offering various advantages to address challenges that stem from other recursion-intensive approaches.

Algorithm 6: Generator Implementation.

```
class INDEXINGROOT inherits Root
    function FLATTEN(ipath, template)
        idx ← ipath.AT(this.level)
        child ← this.value.GET(idx)
        flat ← template.COPY()
        flat.UPDATE(child.FLATTEN(ipath))
        return flat
class KEYINGROOT inherits Root
    function FLATTEN(ipath, template)
        flat ← template.COPY()
        for all child ∈ this.value do
            flat.UPDATE(child.FLATTEN(ipath))
        return flat
class TREE
    function TREE(kdx, data)
        this.count ← 0
        this.depth ← 0
        this.kbucket ← SET()
        this.tree ← this.ROOT(kdx, data)
        this.ibucket ← this.tree.IBUCKET(this.depth)
        this.template ← MAP()
        for all kpath ∈ this.kbucket do
            this.template.UPDATE(MAP(kpath, null))
    function FLATTEN
        if this.count ⩾ LENGTH(this.ibucket) then
            this.count ← 0
            return null
        ipath ← this.ibucket.AT(this.count)
        this.count ← this.count + 1
        return this.tree.FLATTEN(ipath, this.template)
```

4 EVALUATION

To assess the efficacy of the index bucketing algorithm, we evaluate the performance measurements across two prominent question-answering datasets: QuAC (*QuAC, Question Answering in Context. https://quac.ai/*) and NewsQA (*NewsQA: A Machine Comprehension Dataset. https://www.microsoft.com/en-us/research/publication/newsqa-machine-comprehension-dataset/*). These datasets vary in file size: 74 MB and 151 MB respectively. Both datasets come with a myriad of restructuring challenges described below.

- QuAC dataset requires that the background attribute be prepended to each paragraph's context attribute, and data with "CANNOTANSWER" questions and questions without answers need to be filtered out (Fig. 3).

```
{"text": "Miami ... contributed to this report.",

"type": "train",
 "questions": [{
 "isQuestionBad": 0.0,
 "consensus": {
 "s": 15,
 "e": 32
 },
 "validatedAnswers": [{
 "count": 2,
 "s": 15,
 "e": 32
 }],
 "answers": [{
 "sourcerAnswers": [{
 "s": 15,
 "e": 32
 }]
 }],
 "q": "Who reportedly suffers a seizure?",
 "isAnswerAbsent": 0.0
 }],
 "storyId":
"./cnn/stories/6ebb8ab29b94430fa68f0e256c7703d9a41
f8bff.story"}…
```

Figure 3: QuAC question answering dataset structure.

- NewsQA dataset requires data extraction from start and end attributes, into a new answer attribute containing the indicated substring found in the text context, and data with "isQuestionBad" questions ne ed to be filtered out (Fig. 4).

```
{"text": "Miami ... contributed to this report.",
 "type": "train",
 "questions": [{
 "isQuestionBad": 0.0,
 "consensus": {
 "s": 15,
 "e": 32
 },
 "validatedAnswers": [{
 "count": 2,
 "s": 15,
 "e": 32
 }],
 "answers": [{
 "sourcerAnswers": [{
 "s": 15,
 "e": 32
 }]
 }],
 "q": "Who reportedly suffers a seizure?",
 "isAnswerAbsent": 0.0
 }],
 "storyId":
"./cnn/stories/6ebb8ab29b94430fa68f0e256c7703d9a41
f8bff.story"}…
```

Figure 4: NewsQA question answering dataset structure.

The index bucketing algorithm was juxtaposed against two alternative flattening implementations: one leveraging the Pandas Python API and another employing a basic solution that combines recursive and iterative techniques. Summarized in 0, Pandas Python is used as a benchmark for comparison, as it offers a competitive set of methods to flatten nested data collections, such as filling missing values, normalizing dictionaries into new columns, and exploding lists into new records. The basic implementation, on the other hand, serves to demonstrate the worst-case effects of each challenge. Evaluations span various subsets of each dataset incrementally from a Fibonacci-based sequence in the range of 0.1% to 100% to gauge scalability. Each subset underwent evaluations of the observed total time of initialization and execution runtimes. The average runtimes across the evaluations were recorded to ensure more robust assessments.

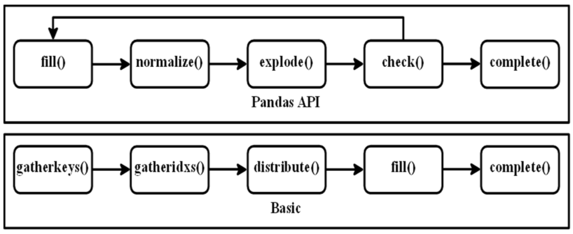


Figure 5: Pandas Python implementation & basic execution pipelines.

The ensuing graphs are organized by implementation and dataset, plotting subset size, measured in bytes, against runtime, measured in seconds. These evaluations were conducted on an Intel Core i7-8750H CPU, 32 GB RAM PC, clocking in at a base frequency of 2.20 GHz, and capable of reaching a maximum turbo frequency of 4.10 GHz. A stringent maximum time limit of thirty minutes was set to avoid prolonged executions, triggering a timeout exception if exceeded. Notably, the basic algorithm showcases an exponential growth pattern in total runtimes, vividly illustrating the cost escalations attributed to challenges that the index bucketing algorithm aims to address. Compared to Pandas Python implementation, our index bucketing framework shows a 24.7% faster total runtime with the QuAC dataset evaluations (0). With the NewsQA larger dataset, the Pandas Python encounters failures, which we suspect are attributed to duplicated data instances within the original dataset. While Pandas Python offers potential solutions to address these errors, implementing such remedies remains nontrivial to the best of our knowledge.

By preserving the original dataset structure, index bucketing eliminates the need for dataset reacquisition during subsequent executions. For instance, considering a scenario where the flattening process is repeated 100 times for each implementation, the index bucketing showcases substantial performance superiority. Although multiple iterations of flattening might not align with typical real-world scenarios, this comparison

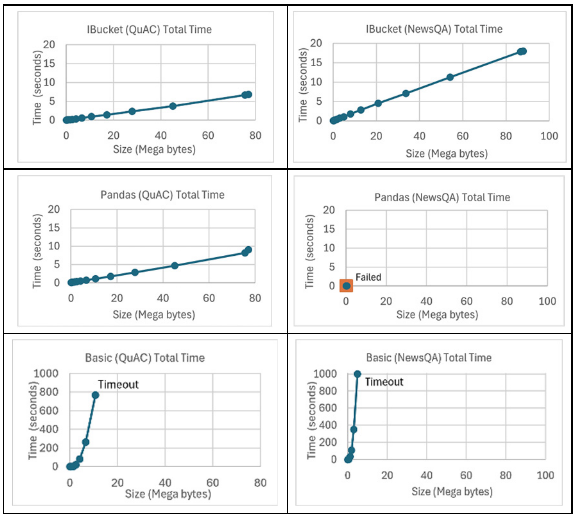


Figure 6: Total runtime evaluation.

demonstrates the index bucketing's efficiency in executing additional feature implementations beyond flattening. Tasks like conditional filtering or attribute selection can be executed notably more efficiently with index bucketing compared to other implementations. The performance results exemplify the enduring advantages of the index bucketing approach in handling repetitive operations and processing complex tasks.

5 RELATED WORK

We have discussed nearly related work on employing declarative querying APIs for integrating data query languages with host programming languages' data processing. Transforming nested queries into efficient forms using set-oriented operators has been investigated for decades in different contexts (Agrawal, 1988; Suciu, 1996). Work presented by (Ulrich, 2019) offers a review of query flattening and descriptions of query flattening in database theory. Obtaining flat outputs in the presence of collection queries was extended to multiset collections via normalization and conservative algorithms (Fegaras & Maier, 2000; Van den Bussche, 2001). Several applications of nested data models build on this calculus (Fegaras & Noor, 2018; Ricciotti & Cheney, 2021).

Another closely related work proposes a framework that translates nested collection queries into a semantically equivalent sequence of queries, where outputs may then be nested and efficiently evaluated (Smith et al., 2021). The framework flattens nested queries by utilizing a series of preprocessing and post-processing algorithms referred to as query shredding and query stitching. This has exhibited effectiveness in addressing information loss, duplication explosion, and irregular schema within the confines of traditional relational database environments.

For resiliency against skewed distribution in query processing, (Rödiger et al., 2016)introduce a distributed join algorithm that detects skewness for relational data by using small approximate histograms and adapting the redistribution scheme to resolve load imbalances. Nonetheless, alleviating performance inefficiencies of flattening nested collections with skew problems remains an open question in the context of query processing (Smith et al., 2020). Our framework addresses the aforementioned challenges which also arise when manipulating these large nested data structures, and has shown the potential to extend its scope to the realm of query processing.

6 CONCLUSIONS

We introduce a novel framework, index bucketing, that aims to address the irregular schema, skewed distribution, information loss, and duplication explosion challenges in the manipulation of nested data structures. Our contributions can be summarized as the following. Employing proactive processes, computational overheads that impede performance are effectively offloaded during initialization, hence enabling a controllable solution for data duplication (Challenge *A*). Addressing skewed data distribution (Challenge *B*) before manipulating the nested structure. This is achieved by aggregating index paths into an index bucket, a mechanism facilitating efficient indexed-hashing access for nested data and ultimately producing flattened records. Addressing irregular schema (Challenge *C*) in the initialization process that includes constructing a flat template—a critical step ensuring every flattened record encompasses all absent keys filled with null values. The architecture of index bucketing, rooted in a platform-independent, tree-based algorithmic structure, aligns seamlessly with the original nested data, preserving its inherent structure and circumventing potential information loss (Challenge *D*). The work explores an intuitive framework for mitigating these challenges assessed on prominent question-answering datasets such as NewsQA and QuAC. Performance is compared against a competitive Pandas Python API implementation and a basic recursive, iterative implementation. Index bucketing compares favorably against these alternatives, exemplifying the enduring advantages of the ability of the framework algorithm to handle repetitive operations and process complex nested data structures. Comparing the performance of index bucketing against larger datasets is a limitation of this study. More insights can be gleaned from further evaluations expanding to other datasets and implementations. Future work will, in part, explore the implications of index bucketing to handle repetitive operations and process complex nested data structures.

REFERENCES

Agrawal, R. (1988). Alpha: an extension of relational algebra to express a class of recursive queries. *IEEE Transactions on Software Engineering*, *14*(7), 879-885. https://doi.org/10.1109/32.42731

Apache Flink. http://flink.apache.org/.

Apache Spark, http://spark.apache.org/.

Diestelkämper, R. (2021). *Explaining existing and missing results over nested data in big data analytics systems* http://dx.doi.org/10.18419/opus-12052.

Diestelkämper, R., Lee, S., Herschel, M., & Glavic, B. (2021). *To Not Miss the Forest for the Trees - A Holistic Approach for Explaining Missing Answers over Nested Data* Proceedings of the 2021 International Conference on Management of Data, Virtual Event, China. https://doi.org/10.1145/ 3448016.3457249.

Fegaras, L., & Maier, D. (2000). Optimizing object queries using an effective calculus. *ACM Trans. Database Syst.*, *25*(4), 457–516. https://doi.org/10.1145/3776 74.3 77676.

Fegaras, L., & Noor, M. H. (2018, 2-7 July 2018). Compile-Time Code Generation for Embedded Data-Intensive Query Languages. 2018 IEEE International Congress on Big Data (BigData Congress), doi: 10.1109/ BigDataCongress.2018.00008.

Grust, T., Rittinger, J., & Schreiber, T. (2010). Avalanche-safe LINQ compilation. *Proc. VLDB Endow.*, *3*(1–2), 162–172. https://doi.org/10.14778/ 1920841.1920866.

Kitsuregawa, M., & Ogawa, Y. (1990). Bucket Spreading Parallel Hash: A New, Robust, Parallel Hash Join Method for Data Skew in the Super Database Computer (SDC). *Vldb '90*, 210–221.

NewsQA: A Machine Comprehension Dataset. https:// www.microsoft.com/en-us/research/publication/news qa-machine-comprehension-dataset/.

Pandas Python, https://pandas.pydata.org/.

QuAC, Question Answering in Context. https://quac.ai/. https://quac.ai/.

Ricciotti, W., & Cheney, J. (2021). Query Lifting. *Programming Languages and Systems*, *12648*, 579 - 606.

Rödiger, W., Idicula, S., Kemper, A., & Neumann, T. (2016, 16-20 May 2016). Flow-Join: Adaptive skew handling for distributed joins over high-speed networks. 2016 IEEE 32nd International Conference on Data Engineering (ICDE), https://doi.org/10.1109/ ICDE.2016.7498324.

Samwel, B., Cieslewicz, J., Handy, B., Govig, J., Venetis, P., Yang, C., Peters, K., Shute, J., Tenedorio, D., Apte, H., Weigel, F., Wilhite, D., Yang, J., Xu, J., Li, J., Yuan, Z., Chasseur, C., Zeng, Q., Rae, I., Biyani, A., Harn, A., Xia, Y., Gubichev, A., El-Helw, A., Erling, O., Yan, Z., Yang, M., Wei, Y., Do, T., Zheng, C., Graefe, G., Sardashti, S., Aly, A. M., Agrawal, D., Gupta, A., & Venkataraman, S. (2018). F1 query: declarative querying at scale. *Proc. VLDB Endow.*, *11*(12), 1835–1848. https://doi.org/10.14778/32298 63.3229871.

Smith, J. (2021). *Declarative nested data transformations at scale and biomedical applications,* University of Oxford.

Smith, J., Benedikt, M., Moore, B., & Nikolic, M. (2021). TraNCE: transforming nested collections efficiently. *Proc. VLDB Endow.*, *14*(12), 2727–2730. https://doi.org/10.14778/3476311.3476330.

Smith, J., Benedikt, M., Nikolic, M., & Shaikhha, A. (2020). Scalable querying of nested data. *arXiv preprint arXiv:2011.06381*.

Suciu, D. (1996). *Parallel programming languages for collections,* University of Pennsylvania.

Ulrich, A. (2019). *Query Flattening and the Nested Data Parallelism Paradigm* Universität Tübingen].

Van den Bussche, J. (2001). Simulation of the nested relational algebra by the flat relational algebra, with an application to the complexity of evaluating powerset algebra expressions. *Theoretical Computer Science*, *254*(1-2), 363-377.

Utilizing Data Analysis for Optimized Determination of the Current Operational State of Heating Systems

Ahmed Qarqour[1,2,*], Sahil-Jai Arora[1,3,*][a], Gernot Heisenberg[2][b],
Markus Rabe[3,][c] and Tobias Kleinert[4][d]
[1]*Bosch Thermotechnik GmbH, Junkersstraße 20-24, 73243 Wernau (Neckar), Germany*
[2]*Department of Information Management, TH Cologne University, Claudiusstraße 1, 50678 Köln, Germany*
[3]*Department IT in Production and Logistics, TU Dortmund University, 44221 Dortmund, Germany*
[4]*Department of Information and Automation Systems for Process and Material Technology,*
RWTH Aachen University, Turmstraße 46, 52064 Aachen, Germany
{Ahmed.Qarqour, Sahil-Jai.Arora}@de.bosch.com

Keywords: Heating Systems, Time Series Analysis, Air-to-Water Heat Pump System, Knowledge Discovery in Databases, Random Forest Algorithm, Field Data, Data-Driven Analysis, Fault Prediction.

Abstract: In response to the pressing global challenge of climate change, the emphasis on sustainable energy technologies has escalated, spotlighting the critical role of heat pump systems as eco-friendly alternatives for heating and cooling. These systems stand at the forefront of efforts to reduce greenhouse gas emissions and improve energy efficiency. The advent of Internet of Things (IoT) technology has unlocked the potential for comprehensive data collection on the operational intricacies of heat pump systems in real-world settings, offering precious insights into their performance and guiding technological advancements. This paper introduces an analytical approach to optimize air-to-water heat pump systems using time series data from Bosch Home Comfort Group's systems. Utilizing Fayyad's data-driven analysis model and the Random Forest algorithm, the study tackles system behavior complexities. Characterized by interpretability crucial for application, it achieves a 97.6% fault detection accuracy. The method encounters difficulties in accurately predicting compressor control faults due to limited data quality and a lack of comprehensive system information. The findings highlight IoT's potential to enhance system efficiency and availability, but also point to the limitations of relying solely on data-driven models for fault prediction in field systems.

1 INTRODUCTION

In 2021, German households consumed about 670 terawatt-hours of energy, mainly for space heating, as per the Federal Environment Agency (Icha and Lauf, 2022). Heat pumps are crucial in this regard, known for their efficiency and ability to reduce utility costs and emissions by leveraging renewable energy (Chiang, 2001). However, realizing their full potential requires understanding their entire lifecycle, from production to user operation. Key stages of this lifecycle encompass product development, manufacturing, storage, transport, installation, operation, and maintenance. These stages primarily generate significant data during the development and operational phases (Wiedemann and Schnell, 2006).

The Internet of Things (IoT) has upgraded data collection, allowing for the extensive networking of devices and sensors with the data stored in the cloud (Zhang et al., 2010). Analyzing these data aims to optimize heating systems. The potential incorporation of suppliers and service providers into this analysis enhances system lifecycle understanding, supporting early fault detection and refining system requirements for future models (Wiedemann and Schnell, 2006).

Fault detection methods in systems are crucial for

[a] https://orcid.org/0000-0002-6877-1480
[b] https://orcid.org/0000-0002-1786-8485
[c] https://orcid.org/0000-0002-7190-9321
[d] https://orcid.org/0000-0001-7441-4431
*These Authors contributed equally to this work

Qarqour, A., Arora, S-J., Heisenberg, G., Rabe, M. and Kleinert, T.
Utilizing Data Analysis for Optimized Determination of the Current Operational State of Heating Systems.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 200-209
ISBN: 978-989-758-716-0; ISSN: 2184-3228

improving efficiency, availability, and customer satisfaction (Chiang, 2001). These methods include model-based, data-based, and hybrid approaches (Zhang and Jiang, 2008). Model-based methods simulate and diagnose the system behavior with mathematical precision, but demand thorough understanding and are complex (Venkatasubramanian et al., 2003). Data-based strategies, leveraging machine learning on historical data, suit complex systems, but need high data quality and substantial resources (Chen, 1999). Hybrid approaches combine the strengths of models and data to efficiently detect faults, providing a balanced solution for fault diagnosis in challenging systems (Yang and Rizzoni, 2016). Data-driven methodologies employ structured models for Knowledge Discovery in Databases (KDD), including the Fayyad KDD framework (Fayyad et al., 1996).

As depicted in Figure 1, this model contains crucial steps for knowledge extraction from databases, starting with the selection of relevant data, followed by its cleaning and formatting in the preprocessing phase. Then, the data are transformed into a format appropriate for mining, after which mining is conducted to discover patterns (Fayyad et al., 1996). These patterns are interpreted to determine their relevance, and finally the extracted insights are presented. This comprehensive process is essential for understanding and enhancing system performance (Garcia et al., 2015).

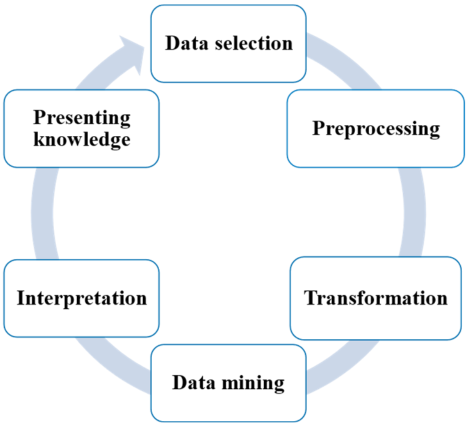


Figure 1: KDD process according to Fayyad.

2 RELATED WORK

2.1 Data-Based Approaches in Heating, Ventilation, and Air Conditioning Systems

With growing demand for efficient and reliable heating, ventilation, and air conditioning (HVAC) systems, the development and application of machine learning algorithms for fault detection and diagnosis (FDD) have become increasingly crucial (Li and O’Neill, 2018). Pioneering work by Gharsellaoui et al. (2020) leverages the Multiclass Support Vector Machine (SVM) algorithm to categorize data within smart buildings effectively. Concurrently, the approach of Ebrahimifakhar et al. (2020) introduces a statistical ML-based classification model using SVM to detect faults in rooftop units by analyzing and classifying data. Similarly, Bode et al. (2020) have developed an innovative FDD model that combines a big data framework with SVMs, aimed at identifying faulty operations in HVAC terminal units through the aggregation and evaluation of data from various sources. Complementing these efforts, Ren et al. (2020) have proposed a comprehensive FDD procedure that merges SVM with principal component analysis (PCA), designed to predict system behavior under new load conditions by extracting pivotal features from the dataset. This ensemble of models underscores the potential and reliability of machine learning in enhancing fault detection and diagnostic capabilities in HVAC systems. An extensive overview of FDD models within the realm of building technology, as detailed in the literature, highlights the eclectic range of approaches and techniques that form the foundation of this field (Li and O’Neill, 2018).

2.2 Key Challenges in Currently Applied Approaches

Heating systems are complex and impacted by diverse operating conditions. The need for interpretable models that can handle this complexity and be applied to different systems is critical. However, challenges arise with data-driven FDD methods developed based on black-box models such as artificial neural networks (ANN) and SVM, mainly due to their lack of interpretability (Yan et al., 2016). This limitation makes it difficult to understand the process of fault identification within these models. Moreover, the effectiveness of data-driven methods largely depends on the quality of the training data (Yang and Rizzoni, 2016). Insufficient data samples and errors in the training data can lead to incorrect classifications. Often the available training data do not cover the entire spectrum of system operation, which limits the model validity to certain conditions. Especially, very critical situations appear only rarely in reality, leading to a deficit of related sensor data. Without interpretability, evaluating model reliability and applicability becomes a challenge (Yan et al., 2016).

2.3 Methodological Contributions

This paper outlines an application-oriented methodology for heat systems employing the Random Forest algorithm for extracting knowledge from data. Central to this approach is its use of decision trees, distinguishing Random Forest by revealing causes of faults through key parameter identification and enhancing model transparency with decision tree visualizations, a clarity lacking in black-box models. Moreover, as an ensemble method, Random Forest reduces overfitting risks by aggregating multiple trees' predictions, ensuring applicability across varied operational conditions (Cutler et al., 2012). This adaptability is essential for analyzing and anticipating system faults, evaluating system performance through error rate analysis, and guiding potential enhancements. Emphasizing its computability, accuracy, and interpretability, this methodology underscores the direct applicability of Random Forest approaches over more complex techniques found in explainable artificial intelligence (XAI), such as Shapley values, ensuring the methodology's efficacy and practical relevance (Başağaoğlu et al., 2022). Thus, this approach provides a fault detection and prediction solution for heating systems in the field, making it particularly valuable for engineers and practitioners in the domain of heating systems.

3 DATA-DRIVEN METHODOLOGICAL FRAMEWORK

This paper presents a structured approach to analyze and explore air-to-water heat pump systems, with a focus 1 faults. Concurrently, the regression segment estimates the remaining time until a fault occurs. These models undergo testing to validate their accuracy in assessing the system status and forecasting faults.

The Model Interpretation stage offers a deep dive into the model's decision-making, elucidating how it identifies the system status and predicts faults. Expert knowledge validates the model's underlying logic.

4 APPLICATION OF METHODOLOGY: INTRODUCTION TO THE USE CASE STUDY

The methodology applied in this paper focuses on an air-to-water heat pump system with a fault in the compressor control. Such control faults arise from issues within the heat pump's control unit and can impact compressor performance. This may lead to inefficient operation of the heat pump, adversely affecting its heating and cooling capacity. Potential causes of these faults include high ambient temperatures around the compressor leading to sensor failures, poor wiring, or incorrect control settings. This analysis was conducted at the Bosch Home Comfort Group. The Heat Pump Development Department was responsible for providing the parameter data and fault information.

The primary objective of this study is to analyze the impact of the fault on the system and to identify the occurrence of the fault using system data. Additionally, the research explores the potential for predicting future fault occurrences. The system data, which describe the system's state, were collected through the bus system. The data analysis is based on time series data with a sampling frequency of 0.83 Hz, covering the period from February 1, 2021, to May 1, 2022. The findings contribute to enhance understanding of the field system's behavior. Based on these insights, strategies to optimize the efficiency and stability of the heat pump system can be developed, ensuring smooth operation in the future.

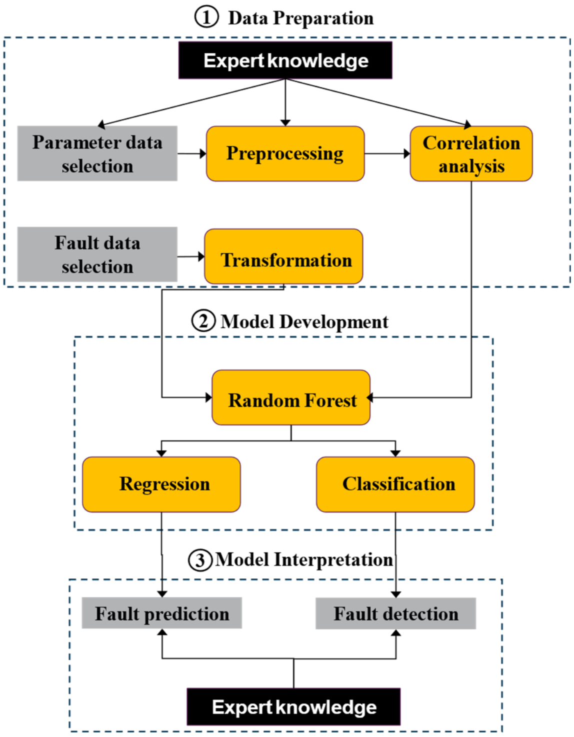


Figure 2: Stages of the methodology.

5 DATA PREPARATION

This stage is designed to achieve a structured and complete dataset. This section outlines how each step of the stage is executed for the investigated use case.

5.1 Data Selection

The data selection for analysis focuses on identifying critical parameters within the heat pump's bus system, which characterize the general state and specifically the control faults in the compressor. This selection is performed in close collaboration with experts in the heat pump development team at Bosch Home Comfort Group to ensure that the chosen data possess the necessary relevance and quality for the study. Verifying the availability and integrity of the data in the bus system is an essential part of this process. Finally, the resulting parameters considered central to the analysis are detailed as follows:

- **Power Setpoint:** Targeted electrical power consumption level for the heat pump, setting the desired performance level for optimal efficiency and meeting heating or cooling demands.
- **Actual Power:** Current electrical power consumption of the heat pump, used to assess energy efficiency and operational status.
- **Actual Compressor Speed:** Current speed at which the compressor is operating, indicating performance level and efficiency of the heat pump.
- **Air Temperature at the Evaporator:** Temperature of air entering the evaporator, helping to evaluate heat exchange efficiency and system load.
- **Temperature of the Compressor:** Current temperature of the compressor, used to monitor compressor health and prevent overheating.
- **Temperature of the Hot Gas:** Temperature of the gas after compression, before condensation, indicating the efficiency of the compression cycle.
- **Evaporator Return Temperature:** Temperature of the fluid returning to the evaporator, assisting in assessing heat absorption efficiency.
- **Outdoor Temperature:** Outdoor ambient temperature, used to adjust operations for optimal efficiency and performance.
- **Condenser Inlet Temperature:** Temperature of the fluid entering the condenser, providing insights into the condensing process efficiency.

5.2 Data Preprocessing

As long as the values of these parameters remain constant, the bus system does not report any values. However, when any value changes, the bus system communicates this change. In the dataset, this leads to empty cells between these two values, which need to be filled to complete the dataset. This is done using the zero-order hold principle, meaning empty cells between two known values are filled with the last known value until a new value is registered.

To detect outliers, data have been visualized using box plots. This decision was driven by the need for a straightforward and visually intuitive method, allowing experts to easily identify and assess unusual values as potential outliers. Box plots were chosen over other methods, because they clearly delineate the range of typical data, making deviations apparent. In the context of missing operational condition details, solely data-driven outlier detection proved to be unreliable (Xu et al., 2020). Instead, combining box plots with expert insights and system specifications enabled a more informed decision on whether values were outliers or relevant variations, ensuring a nuanced and accurate outlier elimination process.

5.3 Correlation Analysis

As mentioned in the previous section, the necessity of this step in the data preparation stage is caused by low data quality. Correlation analysis investigates the relationship between operational parameters to determine how accurately the data represent these physical interactions (Wilcox, 2001). This accurate representation is essential to deliver valid inputs to the model during the training phase. Therefore, the model is enabled to understand the system's status through the available training data and to produce reliable predictions about the system status**.** To achieve this, three sub-steps are involved: a) assessing data normality to select an appropriate correlation method, b) applying the chosen correlation to the dataset, and c) validating the correlation results against the parameters' physical relationships through expert knowledge.

The Shapiro-Wilk test (Ghasemi and Zahediasl, 2012) initially assessed for normal distribution revealed a non-normal distribution that necessitated the use of Spearman's correlation method (Wilcox,

2001) for the analysis. Experts reviewed the correlation coefficients to verify their physical relevance, ensuring that data faithfully represent the system's physical dynamics. This step illuminates crucial relationships between variables and affirms the data's pertinence to the studied physical phenomena.

5.4 Transformation

Fault information is encoded into binary values, with 0 indicating no fault and 1 indicating a fault occurrence, to serve as the target variable for training the Random Forest model. This conversion sets up a classification problem, allowing the model to learn fault detection from parameter data and target variables.

6 MODEL DEVELOPMENT

The Random Forest model is developed to analyze the relationship between various operational parameters and fault occurrences in the air-to-water heat pump system. It comprises two parts: (1) classification model that determines the system's status and (2) regression model predicting the time until a fault occurs. These models were implemented using the scikit-learn library in Python and developed within a Jupyter Notebook.

6.1 System Status Detection

Random Forest Classifier (RFC) employs decision trees on random data subsets, leveraging ensemble learning for accurate classifications while mitigating overfitting and assessing feature importance (Biau and Scornet, 2016). The steps of the model implementation are illustrated in Figure 3. To address the challenge of rare critical situations outlined in Section 2.2, particularly the infrequent occurrence of faults in the compressor control, a down sampling strategy has been implemented.

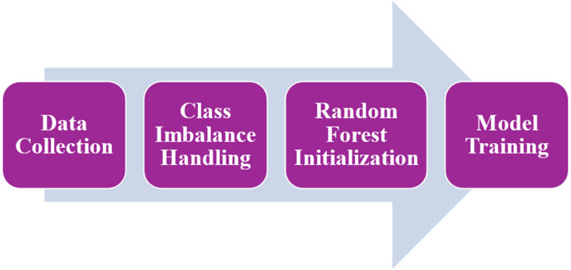


Figure 3: RFC implementation steps.

This method balances the dataset by reducing the number of non-faulty instances to equal the number of faulty instances, ensuring uniform representation. A RFC with three trees (n_estimators=3) and a random_state of 42 is chosen, targeting a balance of model complexity and computational efficiency. The decision to use three trees was based on performance evaluations against a validation set, where adding more trees resulted in only minimal improvements in accuracy, suggesting that further increases would not yield significant benefits. This choice reflects an optimization between simplicity and the ability to capture operational variability, with a random_state of 42 ensuring result reproducibility. Default parameter settings are maintained as detailed in (scikit-learn, 2024).

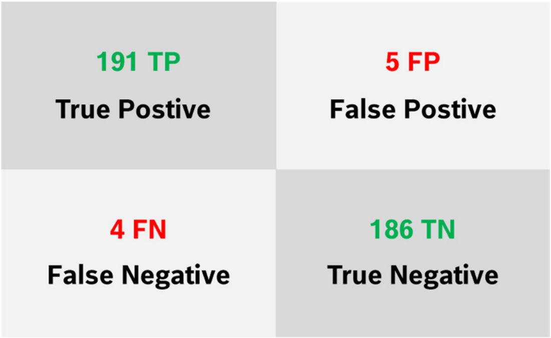
6.1.1 Evaluation of the Detection Model

The model was validated using test data to assess its reliability in predicting on unknown data, using a confusion matrix for evaluating accuracy and precision. Results are illustrated in Figure 4.

This analysis revealed 191 true positives, indicating non-faulty operation status were correctly identified, and 181 true negatives, which means fault status were accurately detected as such. Additionally, the model encountered four false negatives, representing overlooked fault status, and five false positives, where faults were incorrectly identified in non-faulty operation status. Achieving a high accuracy of 97.6% and a precision of 97.4%, the model demonstrates efficient fault detection and classification. Maintaining a low rate of false positives is crucial; they not only lead to unnecessary fault correction costs, but also could divert resources from actual issues, potentially leaving real faults undiagnosed. This emphasis on minimizing false positives is vital for operational efficiency and cost management. The results highlight the model's effective performance in accurately identifying the operation status, balancing accurate fault detection with the imperative to minimize false alarms.

6.1.2 Model Interpretation

The RFC algorithm addresses the challenge of lack of interpretability in data-driven FDD methods based on black-box models as mentioned in Section 2.2. It identifies key parameters through parameter importance calculation – facilitating an understanding of the classification processes – and enhancing transparency while validating the model’s outputs (Breiman, 2001). Through Python's scikit-learn library, feature importance is determined using



191 TP True Postive	5 FP False Postive
4 FN False Negative	186 TN True Negative

Figure 4: Confusion matrix of the test data.

the Mean Decrease in Gini (MDG) method, which assesses how a feature reduces impurity across the model's trees. MDG values range from 0 (no impact) to 1 (perfect prediction capability), where higher values indicate a stronger effect on model decisions (Biau and Scornet, 2016). This calculation considers the decrease in node impurity, weighted by the probability of reaching that node, averaged over all trees (Breiman, 2001). The key findings of the parameter importance are illustrated in Figure 5.

It indicates that specific parameters, such as condenser inlet and outlet temperatures, hot gas temperature, external temperature, compressor speed, and power setpoint are paramount in fault detection, demonstrating nearly equal importance. Conversely, parameters like evaporator air temperature, compressor temperature, evaporator return temperature, and current performance have a lower impact.

These insights emphasize the importance of temperature-related measurements in detecting the fault. Through the interpretability of the model, these insights into model parameters can be traced back to the faulty state of the system. The relevance of the parameters to the faulty state are confirmed by the experiential knowledge of experts.

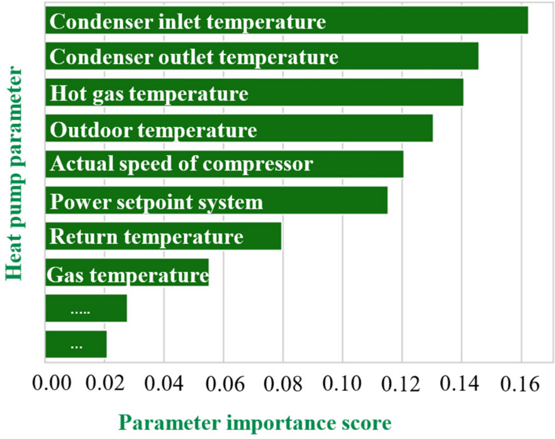


Figure 5: Visualization of the importance of parameters.

This validation not only enhances the development steps of the component to prevent the occurrence of such faults but also expands the knowledge of relevant factors that can lead to faults. In the future, this approach can also be applied to other types of faults to gain valuable insights.

6.2 System Status Prediction

This model aims to predict the remaining time until the next fault occurs, utilizing an ensemble of decision trees to make accurate predictions on continuous values by averaging the outputs of all trees in the forest. Similar to the classifier model, the Random Forest Regressor (RFR) applies ensemble learning, but focuses on estimating continuous outcomes. The implementation of the RFR mirrors that of the classifier model, as depicted in Figure 6.

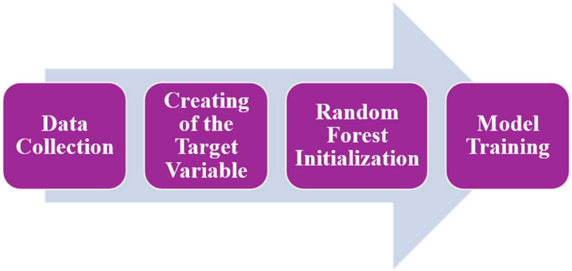


Figure 6: RFR implementation steps.

The process starts with data collection representing various operational conditions, followed by creating the target variable time until the next fault, which is hereafter referred to as "Time to Failure". This is achieved by reverse iterating through the data to calculate the time until the next fault for each data point, producing a list of minutes until the next fault. For this model, a RFR with ten trees (n_estimators=10) and a random_state of 42 was selected, balancing model complexity with computational efficiency. The choice of more trees for the RFR compared to the RFC reflects the increased complexity needed in regression models to capture data variability and nuances accurately (Corrales et al., 2018). With this optimized tree ensemble, the RFR can more accurately identify and predict underlying trends, enabling precise predictions for the time to failure. The default parameters are retained as described in (scikit-learn, 2024).

6.2.1 Evaluation of the Prediction Model

The model accuracy was validated using test data to assess its reliability in predicting on unknown data using the Mean Absolute Error (MAE). The test

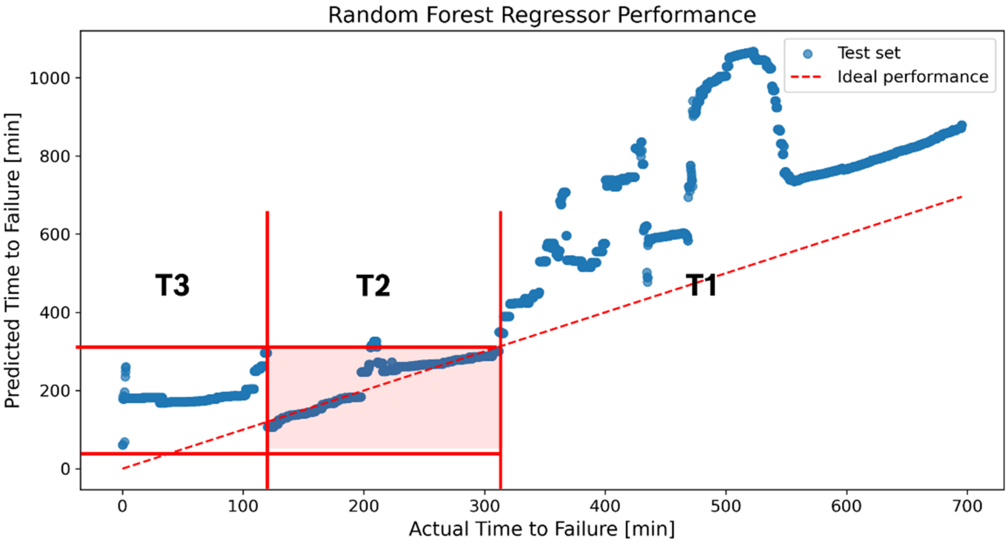


Figure 7: Accuracy of the model with test data.

dataset contains a fault scenario. The results were visualized in Figure 7, where the X-axis represents the actual values and the Y-axis the predicted values of "Time to Failure". Ideal model performance is achieved when data points closely align along the ideal performance line, aiming for an MAE value of 0, indicating precise alignment between predictions and actual events.

The accuracy evaluation of the model reveals three key insights that provide a nuanced view of the model's performance across different periods before a fault event.

- Phase T1: Actual Time to Failure > 320 minutes
- Phase T2: 320 minutes ≥ Actual Time to Failure ≥ 120 minutes
- Phase T3: 120 minutes > Actual Time to Failure > 0 minutes

Phase T1 describes long-term predictions, starting from 320 minutes before the fault occurrence. In this phase, it was observed that the model appears incapable of detecting reliable indicators of an impending fault, resulting in a large discrepancy between predicted and actual values. This limitation highlights the challenges in predicting faults over an extended period. Phase T2 describes mid-term predictions, between 120 and 320 minutes before the fault occurrence. In contrast to Phase T1, the model demonstrates considerably better performance with a MAE of 18.6 minutes. During this critical period, the model effectively analyzes and interprets operational conditions and potential signs of an impending fault, indicating its capability to utilize relevant information for fault prediction. Phase T3 involves short-term predictions made 0 to 120 minutes before a fault occurs. In this phase, the model's accuracy decreases, primarily due to a significant deviation of data points from the ideal line. This reduction in accuracy can be attributed to insufficient information density in the parameters, leading to unreliable predictions.

However, the application of the model to other faulty scenarios has revealed significant limitations, primarily due to the limited availability of faulty data and limited understanding of the underlying causes.

This problem is closely linked to the challenge of data quality and availability, as discussed in Section 2.2. The lack of comprehensive data sets significantly impairs the model's ability to predict under different operating conditions. In addition, the complexity of the heat pump system combined with a limited data set further reduces the model’s prediction accuracy. This is compounded by uncertain causes of failure such as wiring or software issues, which are discussed in more detail in Section 4. Ultimately, these challenges emphasize the urgent need for improved data quality and a deeper understanding of failure mechanisms.

6.2.2 Model Interpretation

In the RFR model, evaluating parameter significance is the key to decoding its predictive logic. This process identifies the extent to which various features impact the model's ability to predict the timing of a fault. Understanding the critical features enhances the insight into the model's operational dynamics. Differing from the RFC model, the RFR model assesses the feature importance via the Mean Decrease in Impurity (MDI). MDI reflects how each feature's variance reduction, averaged across all trees, contributes to the model's accuracy. This method highlights the influence of specific features on enhancing the model's precision by reducing prediction variance through data segmentation.

This analysis reveals that the outdoor temperature and air temperature at the evaporator exert the strongest influence on prediction accuracy, with a combined importance of 50%. Additionally, the condenser exit temperature and the power setpoint also make significant contributions to the forecast, both with importance of 14%. These four parameters collectively account for 78% of the predictive influence.

These results emphasize two major results: (1) temperature-related measurements and the power setpoint in the context of precise fault prediction are crucial and (2) there is a need for the extension of the knowledge about the selection of relevant parameters for fault monitoring and the definition of the time period in which a fault can be predicted. Despite the limited number of fault cases in the system history, these findings are valuable for future research and help in the selection of time periods and relevant parameters in the model training to reduce model complexity.

7 DISCUSSION OF RESULTS

The research findings, which were discussed with engineering experts from the heat pump department at Bosch Home Comfort Group, focus on four key questions:

- How does interpretability clarify causality between system parameters and faults while supporting model scalability?
- Which benefits does a system status detection model offer?
- How does the parameter significance derived from the classification model affect error detection logic and contribute to the optimization of the regression model for error prediction?
- How could more diverse data improve fault prediction, and what are the challenges?

Regarding the first aspect (interpretability), discussions with the experts in the heat pump department emphasize the importance of interpretability for scaling the model to systems with similar data deficiencies. As explained in Section 6.1.2, the interpretability of the model enables the exact quantification of the meaning of the parameters. This improves the understanding of how each feature affects the predictions of the model. This insight is crucial for accurate adjustments when applying the model to new systems. This ensures the effectiveness of the model in different operating environments. This detailed interpretative analysis also helps to adapt the model and standardize fault detection practices across different environments.

Regarding the second aspect (benefits of a detection model), experts highlight the significant benefits of a system status detection model, especially for systems that do not capture fault data. Such a model enables an understanding of the system's behavior in operation, identification of common faults, and efficient resource planning, directly contributing to the optimization of the system design.

Concerning the third aspect (parameter significance), discussions with experts emphasize the importance of specific parameters, such as condenser inlet and outlet temperatures, hot gas temperature, and outdoor temperature, identified in Section 6.1 as crucial for detecting faults within the compressor control. Expert opinions indicate that future research could significantly enhance prediction accuracy by redesigning the error detection logic to reflect parameter relevance and optimizing the online monitoring of these parameters. Achieving this improvement also involves intensified collaboration with service companies to obtain detailed fault information, including causes of occurrences. This collaboration forms the foundation for a more efficient predictive control system, aimed at reducing downtime and improving overall system performance.

The last aspect discussed with the experts involves analyzing the predictive model’s capability to determine the precise time phase when a fault can be anticipated within the system**.** The model – under the constraints of current assumptions and data rarity – identifies early symptoms of errors occurring between 120 and 320 minutes. This preliminary insight is crucial as it suggests that expanding our dataset with a broader range of failure cases could potentially reduce the need for extensive training data and help avoid overfitting. Enhancing the dataset in this manner would improve the model’s accuracy and its applicability to similar systems.

8 SUMMARY AND FUTURE WORK

This paper explores the potential of time series analysis of sensor data from heating systems in operation for detecting and predicting errors, a critical area complicated by the significant distance between users and manufacturers. A procedure based on Fayyad's model was implemented and applied to an air-to-water heat pump system to identify and forecast specific control faults in the compressor.

A RFC model was developed to recognize system status and assess the impact of parameter weights on fault detection. This model successfully determined the status of the systems, achieving a detection accuracy of 97.6% and a precision of 97.4%. A key challenge was the limited dataset, which complicated the expert validation and underscored the necessity for a larger data foundation. The analysis underscored the significance of certain parameters, particularly temperature readings, in fault detection. Experts validated these findings, emphasizing the need for ongoing adjustment of weight factors.

The limited availability of fault data and the lack of system information restricts the effectiveness of the RFR model. This limitation stems from the system's lifecycle; after sale, third-party service and maintenance companies oversee installation and upkeep, while manufacturers conduct field monitoring for a brief period. As a result, failure data collection is primarily limited to this monitoring phase, thus affecting the model's ability to predict accurately.

Future research directions, inspired by this work, will explore the potential of Random Forest models to analyze more extensive datasets with increased error instances and assess other machine learning algorithms for error detection and prediction in heat pump systems. An optimized dataset, including detailed parameter and fault information, is crucial for developing models that accurately reflect system reliability and behavior. Additionally, future studies should explore the reliability of specific system components and their impact on overall system reliability. Future investigations should incorporate not only existing data but also laboratory results, simulations, and physical models. The integration of physics-based models will be explored to establish causal relationships between system parameters and fault occurrences, thereby enhancing the model's ability to predict and diagnose faults with higher accuracy. This approach is expected to improve the overall effectiveness of the system, contributing to a deeper understanding of system dynamics, and advancing control strategies for heating systems.

REFERENCES

Arora, S.-J., & Rabe, M. (2023). Predictive maintenance: Assessment of potentials for residential heating systems. *International Journal of Computer Integrated Manufacturing*, 1--25. https://doi.org/10.1080/0951192X.2023.2204471.

Başağaoğlu, H., Chakraborty, D., Lago, C. D., Gutierrez, L., Şahinli, M. A., Giacomoni, M., Furl, C., Mirchi, A., Moriasi, D., & Şengör, S. S. (2022). A review on interpretable and explainable artificial intelligence in hydroclimatic applications. *Water, 14*(8), 1230. https://doi.org/10.3390/w14081230.

Biau, G., & Scornet, E. (2016). A random forest guided tour. *TEST, 25*(1), 197--227. https://doi.org/10.1007/s11749-016-0481-7.

Bode, G., Thul, S., Baranski, M., & Müller, D. (2020). Real-world application of machine-learning-based fault detection trained with experimental data. *Energy, 198*, 323. https://doi.org/10.1016/j.energy.2020.117323.

Breiman, L. (2001). Random forests in machine learning. *Springer, New York, NY*, 5--32. https://doi.org/10.1023/A:1010933404324.

Chen, J. (2013). Model-based fault diagnosis in dynamic systems using identification techniques. *Springer, London, United Kingdom*. ISBN: 978-1-4471-3829-7. https://doi.org/10.1007/978-1-4471-3829-7.

Chiang, L. H. (2001). Fault detection and diagnosis in industrial systems. *Springer, London, United Kingdom*. https://doi.org/10.1088/0957-0233/12/10/706.

Corrales, D., Corrales, J., & Ledezma, A. (2018). How to address the data quality issues in regression models: A guided process for data cleaning. *Symmetry, 10*(4), 99. https://doi.org/10.3390/sym10040099.

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). Random forests in Ensemble Machine Learning. *Springer, New York, NY*, 157--175. https://doi.org/10.1007/978-1-4419-9326-7_5.

Dey, M., Rana, S. P., & Dudley, S. (2020). Smart building creation in large scale HVAC environments through automated fault detection and diagnosis. *Future Generation Computer Systems, 108*, 950--966. https://doi.org/10.1016/j.future.2018.02.019.

Ebrahimifakhar, A., Kabirikopaei, A., & Yuill, D. (2020). Data-driven fault detection and diagnosis for packaged rooftop units using statistical machine learning classification methods. *Energy and Buildings, 225*, 318. https://doi.org/10.1016/j.enbuild.2020.110318.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM*, *39*, 27--34. https://doi.org/10.1145/240455.240464.

García, S., Luengo, J., & Herrera, F. (2015). Data preprocessing in data mining. *Springer International Publishing, Cham, Switzerland*, 19--38. https://doi.org/10.1007/978-3-319-10247-4.

Gharsellaoui, S., Mansouri, M., Trabelsi, M., Harkat, M.-F., Refaat, S. S., & Messaoud, H. (2020). Interval-valued features based machine learning technique for fault detection and diagnosis of uncertain HVAC systems. *IEEE Access, 8*, 892--902. https://doi.org/10.1109/ACCESS.2020.3019365.

Ghasemi, A., & Zahediasl, S. (2012). Normality tests for statistical analysis: a guide for non-statisticians. *International Journal of Endocrinology and Metabolism, 10*(2), 486--489. https://doi.org/10.5812/ijem.3505.

Icha, P., & Lauf, T. (2022). Entwicklung der spezifischen Treibhausgas-Emissionen des deutschen Strommix in den Jahren 1990–2021. Retrieved February 12, 2024, https://www.umweltbundesamt.de/sites/default/files/medien/1410/publikationen/2022-04-13_cc_15-2022_strommix_2022_fin_bf.pdf.s

Li, Y., & O'Neill, Z. (2018). A critical review of fault modeling of HVAC systems in buildings. *Building Simulation*, *11*(5), 953--975. https://doi.org/10.1007/s12273-018-0458-4.

scikit-learn. (2024).

sklearn.ensemble.RandomForestClassifier. Retrieved April 23, 2024,

https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., & Kavuri, S. N. (2003). A review of process fault detection and diagnosis: Part I: Quantitative model-based methods. *Computers & Chemical Engineering, 27*(3), 293--311. https://doi.org/10.1016/S0098-1354(02)00160-6.

Wiedemann, B., & Schnell, G. (2006). Bus systems in automation and process technology. *Vieweg+Teubner*, 151–344. https://doi.org/10.1007/978-3-8348-9108-2_4.

Wilcox, R. (2001). Fundamentals of modern statistical methods. *Springer, New York, NY*, 67--91. https://doi.org/10.1007/978-1-4757-3522-2.

Xu, X., Lei, Y., & Li, Z. (2020). An incorrect data detection method for big data cleaning of machinery condition monitoring. *IEEE Transactions on Industrial Electronics, 67*, 326--336. https://doi.org/10.1109/TIE.2019.2903774.

Yan, R., Ma, Z., Zhao, Y., & Kokogiannakis, G. (2016). A decision tree based data-driven diagnostic strategy for air handling units. *Energy and Buildings*, *133*, 37--45. https://doi.org/10.1016/j.enbuild.2016.09.039.

Yang, R., & Rizzoni, G. (2016). Comparison of model-based vs. data-driven methods for fault detection and isolation in engine idle speed control system. *In Proc. of PHM Conference*, *8*(1), Oct. 2016. https://doi.org/10.36001/phmconf.2016.v8i1.2502.

Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications, 1*(1), 7--18. https://doi.org/10.1007/s13174-010-0007-6.

Zhang, Y., & Jiang, J. (2008). Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control, 32*(2), 229--252. https://doi.org/10.1016/j.arcontrol.2008.03.008.

A Core Technology Discovery Method Based on Hypernetwork

Chen Wenjie[a]
National Science Library (CHENGDU), Chinese Academy of Sciences, Qunxian South Street, Chengdu, China
chenwj@clas.ac.cn

Keywords: Identification of Core Technologies, Technology Co-Occurrence Hypernetwork, Hypernetwork Embedding, Centrality Measurement.

Abstract: Identifying and analyzing the core technologies in a specific technical field can comprehensively understand the research status and development trends in that field, providing reference and suggestions for the research and development of key and disruptive technologies. This article introduces the technique of representing the multiple co-occurrence relationships between entities using hypernetwork structure, and uses hypernetwork embedding technology to automatically generate technology node vectors that integrate structural and attribute features. Through fuzzy clustering, technology clusters are obtained, and measurement indicators such as local neutrality, semi local centrality, and global centrality based on hypernetwork structure are constructed to identify the core technology nodes in each technology cluster. Taking the field of carbon capture, utilization, and storage technology as an example, the effectiveness and scientificity of the method proposed in this article were verified. The results showed that chemical absorption, membrane separation, solid adsorption, and low-temperature separation are the core technologies in this field, which helps China to allocate resources reasonably, increase research and development efforts in core technology, and gain competitive advantages.

1 INTRODUCTION

Core technology refers to the basic, progressiveness and dominant technology in a specific technical field, which has a significant impact on other technologies and is characterized by long cycle, complexity and high investment. How to eliminate the low value technology foam from the vast number of patented technologies and accurately identify the high value core technology has become an urgent problem.

The technology co-occurrence network is a typical relational network that characterizes the evolution and integration process and innovation mechanism of technology in scientific research activities, with technology entities as nodes, technology co-occurrence relationships as edges, and co-occurrence frequency as edge weights. It has been widely used in research such as technology identification, technology opportunity discovery, and technology layout. The technology co-occurrence network can intuitively reflect the co-occurrence frequency (node degree) of technology entities in patents, but cannot characterize whether a patent is composed of two or more technologies. This is because each edge of the network can only connect two nodes, and when modeling co-occurrence relationships, higher-order multivariate relationships are flattened and mapped to lower order binary relationships, resulting in a lack of multivariate co-occurrence information. As shown in Figure 1, a hypernetwork is an extension of a regular network, where each hyperedge can contain any number of nodes, allowing for a more accurate characterization of complex co-occurrence relationships between technical entities. This is more in line with the diverse, complex, and clustered characteristics of co-occurrence relationship structures.

In summary, this article proposes a core technology identification method based on technology co-occurrence hypernetworks. Firstly, attempt to introduce hyper network structures into traditional technology co-occurrence networks to characterize the complex and diverse co-occurrence relationships between technological entities; Then, the structural and attribute features of the co-occurrence network are modeled using a hypernetwork embedding model to overcome the

[a] https://orcid.org/0000-0001-7515-3624

Wenjie, C.
A Core Technology Discovery Method Based on Hypernetwork.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 210-217
ISBN: 978-989-758-716-0; ISSN: 2184-3228

shortcomings of traditional manual construction of high-order features, and different types of technology clusters are obtained through fuzzy clustering; Finally, the centrality index based on ordinary network structure is improved to be suitable for weighted hypernetworks, measuring the importance of nodes in each technology cluster from three dimensions: local centrality, semi local centrality, and global centrality to identify core technologies.

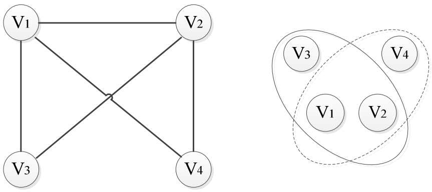


Figure 1: Technology Co-occurrence Hypernetwork.

2 RELATED WORK

The methods for discovering core technologies can be roughly divided into qualitative methods represented by expert analysis, quantitative methods represented by network analysis or text analysis, and a combination of qualitative and quantitative methods.

The identification method based on relational networks is an important research direction in quantitative analysis methods. This type of method utilizes relationships such as referencing, similarity, and co-occurrence between technical entities to construct a technical relationship network, and then identifies the core technologies in the network through machine learning models, complex network analysis, and the construction of an evaluation index system based on the network's structure, attributes, temporal and thematic characteristics (Liu, 2022). In reference based methods, (Kajikawa, 2022) constructed a citation network in the field of energy research, clustering nodes based on the topology of the network, and tracking emerging energy technologies by calculating the citation characteristics and average publication time of paper nodes in each cluster. (Cho, 2008) constructed a patent citation network for Taiwan Province from 1997 to 2008, based on structural hole theory to identify core technologies in the network. (Qi, 2020)combined main path analysis and small world network characteristics to identify core technology nodes in patent citation networks. This type of method is widely used, but there is a lag problem, that is, it takes a long time to reflect the citation of scientific and technological literature, and the identified technology tends to be closer to hot technologies rather than core technologies.

In the similarity based approach, (Kong, 2021) trained the BERT model to obtain patent vectors and constructed a patent similarity matrix, and used the DNN model to identify key technologies from identifying off patents. (Song, 2018) constructed a similarity network using the coupling relationship between patents, using the clustered peripheral nodes as candidate technologies and identifying core technologies using two indicators: technology and market. (Wu, 2023) first extracted core technical themes from patent texts and calculated the similarity between technical themes. Then, they constructed a technical topic similarity network, obtained technical module clustering through K-means algorithm, and finally used TRIZ theory nine screen method to determine cutting-edge technologies. This type of method can accurately measure the semantic similarity between technologies, but its interpretability and computational efficiency are poor, making it impossible to prove the existence of core technologies in highly similar technology clusters. In the co-occurrence based approach, (Huang, 2019) extracted technical terms from patent literature to construct a co-occurrence network, and used a link prediction model to predict the dynamic network of technology, identifying core technical themes from both influence and novelty. (Luo, 2017) constructed a patent co-occurrence network using patent subclasses as technology nodes, obtained candidate technology nodes using k-kernel analysis, and identified core technologies from candidate technologies using evidence reasoning. Chen Yuxin et al. constructed a technology and scientific knowledge co-occurrence network based on patent citation data, and used the core edge theory to predict disruptive technologies from three aspects: foundational, influential, and abrupt. (Dotsika, 2017) predicted core technologies through co word network analysis and visualization methods. (Dou, 2023) constructed a relationship network using the influence of journals and the co-occurrence frequency of keywords, generated anonymous random walk sequences for the network, and then trained vector sequences using the word2vec model. Finally, the similarity between sequences was used to characterize the similarity of the evolution of the technology to identify reversal techniques. This type of method lacks the disclosure of the attribute characteristics of technical entities and the strength characteristics of relationships between entities, and usually selects some high co-occurrence frequency technologies as core technologies.

3 CORE TECHNOLOGY DISCOVERY BASED ON TECHNOLOGY CO-OCCURRENCE HYPERNETWORKS

3.1 Construction and Symbolic Definition of Technology Co-Occurrence Hypernetwork

In the construction of technology relationship networks, the definition, concept, and connotation of technology nodes are diverse. They can be entities of different granularities such as scientific literature, themes, and keywords, as well as different types of entities such as technical documents, technical terms, patent classification numbers, inventors, and institutions. These entities are linked into a network structure through relationships such as citation, co-occurrence, and similarity. Therefore, how to choose appropriate entities and relationships to characterize the technical relationship network is a primary issue. Due to the possibility of multiple different forms of text expression in the same technology, the core technology discovery method based on text content features needs to deal with issues such as noise and ambiguity, and the effectiveness of recognition results is limited by the accuracy of entity extraction and semantic understanding ability. IPC (International Patent Classification Number) can effectively classify and organize technical fields, and is a symbol of patent creativity and novelty. Each patent is limited and described by several IPC's for its purpose or purpose, making it a good representation object for technical entities. IPC co-occurrence reveals the correlation between technologies, reflects the process of technology integration and evolution, and a comprehensive analysis of it can support research on core technology discovery in the field. Therefore, this article takes the IPC of a patent as a technical node, the multivariate co-occurrence relationship formed by multiple IPCs as a hyperedge, and the corresponding patent information as an attribute of the hyperedge, constructs a technical co-occurrence hypernetwork, and generates the corresponding structural feature matrix and node feature matrix. The hypernetwork constructed in this way can better describe the interrelationships between technological entities, patents, and technology patents, which is more in line with the characteristics of multi-agent, aggregation, and mutual nesting in co-occurrence structures. For example, this hypernetwork can simultaneously characterize the ternary co-occurrence relationship (v_x, v_y, v_z) and binary co-occurrence relationship (v_x, v_y) have limitations for ordinary networks.

The formal definition of a technology co-occurrence hypernetwork is HN=(V, E)， Among them, the technical node set $V = \{v_1, v_2, \ldots, v_m\}$， Hyperedge set $E = \{e_1, e_2, \ldots, e_n\}$, m, and n are the number of nodes and edges, respectively. The weight setting of hyperedges and nodes has a significant impact on the structural characteristics of technical co-occurrence hypernetworks. Here, the contribution of a single patent to the hyperedge weight is set to 1, and the weight of hyperedge e_j is as follows:

$$w(e_j) = |D(e_j)| \qquad (1)$$

$w(e_j)$characterizes the co-occurrence strength of technical nodes, $D(e_j)$is the patent set corresponding to the hyperedge e_j, where the IPC of each patent is exactly the same. A hyperedge contains several technical nodes, each of which contributes differently to the hyperedge, and it is necessary to allocate weights to the technical nodes reasonably. The main IPC of a patent typically provides invention information representing technological innovation, while the secondary IPC provides additional information about the patent. From the perspective of knowledge spillover, the technical knowledge produced by the main IPC flows towards the sub IPC, forming a knowledge transfer relationship, and the main IPC should have a higher weight. Therefore, the weight of node v_i on the superedge e_j is defined as:

$$w(v_i, e_j) = \sum_{d \in D(e_j)} w(v_i, d) \qquad (2)$$

$$w(v_i, d) = \begin{cases} \alpha, & \text{if } v_i \; is\ the\ main\ IPC\ of\ patent\ d \\ \frac{1-\alpha}{n(d)-1}, & \text{if } v_i \text{ is the sub IPC of patent d} \end{cases}$$

Among them, $w(v_i, d)$ The weight of node v_i in patent d, α is a weight parameter, $n(d)$ is the number of IPC in patent d.

The structural feature matrix of the hypernetwork *HN* includes: correlation matrix, hyperedge weight matrix, node degree matrix, and hyperedge degree matrix. The correlation matrix $\boldsymbol{H}$ is an $m \times n$ real matrix, and its elements are defined as follows:

$$h(i,j) = \begin{cases} 1, v_i \in e_j \\ 0, \ v_i \notin e_j \end{cases} \qquad (3)$$

If the technical node v_i is associated with the hyperedge e_j, then the corresponding element value in matrix $\boldsymbol{H}$ is 1, otherwise it is 0. The hyperedge weight matrix is a diagonal matrix defined as follows:

$$\boldsymbol{W_e} = \mathrm{diag}(w(e_1), \dots, w(e_n)) \quad (4)$$

The node degree matrix D_v and the hyperedge degree matrix D_e are both diagonal matrices, defined as:

$$\boldsymbol{D_v} = \mathrm{diag}(d(v_1), \dots, d(v_m)) \quad (5)$$

$$\boldsymbol{D_e} = \mathrm{diag}(\mathrm{d}(e_1), \dots, d(e_n)) \quad (6)$$

$$d(v_i) = \sum_{j=1}^{n} h(v_i, e_j) w(e_j) \quad (7)$$

Among them, $d(v_i)$ is the degree of node v_i, which represents the sum of weights of hyperedges associated with v_i and reflects the activity of the technology. The larger the value, the more patents the corresponding technology appears, and the greater its influence on other technologies. $d(e_i)$ is the degree of the hyperedge e_i, which refers to the number of hyperedges that share a common technical node with that hyperedge. The higher the value, the higher the degree of connectivity within the hypernetwork.

The node feature matrix N_f of the hypernetwork HN is trained by a deep autoencoder DAE, as follows:

$$N_f = \mathrm{DAE}(onehot(v_1), \dots, onehot(v_m))$$

Among them, $onehot(v_i)$ is the unique hot code generated by the IPC corresponding to node v_i. IPC reflects the inherent characteristics of technology and determines the technical topics involved in patents through a 5-layer structure of parts, categories, subcategories, main groups, and groups. Therefore, $onehot(v_i)$ is also composed of 5 separate hot codes spliced together, with each part encoding a length equal to the total number of corresponding hierarchical structure categories. The dimension of the vector corresponding to $onehot(v_i)$ is too large and very sparse (most elements are 0). Directly using it as a node feature vector will affect the flexibility and computability of vector representation. Further use of DAE is needed to convert it into a low dimensional dense vector.

3.2 Technology Cluster Analysis

Technology clustering aims to assign technology nodes to different clusters, where each cluster structure can be viewed as a technology cluster. The strong correlation between technology nodes within a technology cluster has similar technical characteristics, while the weak correlation between different clusters reflects the diversity of technology. Through technical clustering analysis, not only can we grasp the correlation between various technologies, but we can also reflect the evolution dynamics of technologies, more accurately perceive core technologies, and guide R&D personnel to adjust technical strategies in a timely manner. Traditional node clustering or community partitioning algorithms are only applicable to common structures such as citation networks and co-occurrence networks, and cannot handle higher-order structures such as hypernetworks and hypergraphs. And the hyper network embedding technology can automatically convert high-order network structure data into low dimensional computable feature vectors, avoiding manual construction of high-order relational features, thereby more efficiently supporting downstream tasks such as node clustering, classification, and link prediction. Therefore, this article uses the hyper network embedding model HGNN (Hypergraph Neural Networks) to map the technical co-occurrence hypernetwork to a low dimensional vector space, achieving vectorized representation of technical nodes. Then, the FCM (Fuzzy C-Means) algorithm is used to complete fuzzy clustering of node vectors, and the technical clustering results are obtained.

HGNN is a spectral domain based convolutional neural network model that can model high-order multivariate co-occurrence relationships between technical nodes. It takes the structural feature matrix and node feature matrix of the technical co-occurrence hypernetwork as inputs, and combines the two features into a technical node vector through multiple hypernetwork convolutional layers, as shown in Figure 2. The corresponding convolution operator is defined as:

$$X' = D_v^{-\frac{1}{2}} H W_e D_e^{-1} H^T D_v^{-\frac{1}{2}} X\theta \quad (8)$$

HGNN aggregates technical node information onto hyperedges through the calculation of the $D_e^{-1} H^T D_v^{-\frac{1}{2}} X\theta$ part, and then aggregates hyperedge information onto relevant nodes using the $D_v^{-\frac{1}{2}} H W_e$ calculation. Among them, θ is the convolutional layer parameter, X is the input node feature matrix, X' is the node feature matrix output after convolution calculation.

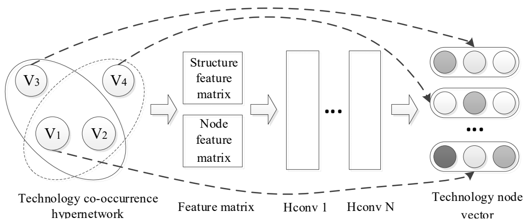


Figure 2: Vectorization of hypernetwork node.

In traditional clustering algorithms, a technology node can only belong to one technology cluster, while fuzzy clustering can calculate the probability distribution of technology nodes belonging to different clusters, more accurately modeling the complex membership relationships between nodes and clusters. FCM calculates the membership matrix $\boldsymbol{U} = \left(u_{i,c}\right)_{\mathrm{m}\times\mathrm{k}}$ of the technical node vector by minimizing the weighted Euclidean distance. The objective function for achieving the closest distance between technology nodes in the same cluster and the larger distance between nodes in different clusters is:

$$\mathrm{J} = \sum_{i=1}^{m} \sum_{c=1}^{k} u_{i,c}^{a} d_{i,c}^{2} \tag{9}$$

Among them, $u_{i,c}$ is the probability that technology node v_i belongs to technology cluster c, k is the number of technology clusters; The sum of the membership degrees of each technology node v_i to k clusters is 1, that is, $\sum_{c=1}^{k} u_{i,c} = 1$; a is the fuzzy coefficient, $d_{i,c}$ is the Euclidean distance from the node vector to the cluster center.

3.3 Core Technology Discovery

Core technology is a dominant and irreplaceable technology in the field of technology, which plays an important supporting role in other technologies. Currently, core technology discovery methods based on technology relationship networks typically rely solely on node path analysis, centrality measurement, structural holes, and community structure information to identify key nodes from the network as core technologies, resulting in relatively one-sided identification results. This article evaluates the core level of technology from the perspective of a hypernetwork. After conducting technical clustering analysis based on the structural and node characteristics of the hypernetwork, the most suitable nodes for core technical features are excavated from different technology clusters. For each technology cluster, a combination of local, semi local, and global centrality indicators is used to identify core technology nodes. Local centrality indicators are used to identify technology nodes with strong irreplaceability, semi local centrality indicators are used to detect technology nodes with significant influence on the domain technology, and global centrality indicators are used to discover technology nodes that dominate the entire network. The measurement indicators for core technology nodes are defined as follows:

$$\mathrm{CT}(v_i) = EWM\left(\mathrm{LC}(v_i),\ \mathrm{SLC}(v_i),\ \mathrm{GC}(v_i)\right) \tag{10}$$

Among them, EWM (the entropy weight method) represents the entropy weight method, which measures the internal differences of various indicator data and determines the weight value through information entropy.

The local centrality index evaluates node centrality by using the degree of overlap of node neighborhood structures. It is believed that the more neighbors the target node has and the lower the topological similarity between neighboring nodes, the stronger the irreplaceability of the target node in network function and structure, and the higher the local importance of the target node. Taking into account the structural similarity information of nodes in the hypernetwork and the weight information of their hyperedges, the local centrality index is defined as follows:

$$\mathrm{LC}(v_i) = \sum_{v_b, v_c \in N(v_i)} 1 - sim(v_b, v_c) \tag{11}$$

$$sim(v_b, v_c) = \sum_{v_u \in N(v_b) \cap N(v_c)} \frac{w(v_b, v_u) + w(v_u, v_c)}{d(v_u)} \tag{12}$$

Among them, $sim(v_b, v_c)$ is the similarity in the adjacency structure between node v_b and node v_c, with a higher value indicating a higher degree of structural overlap between the two nodes. $N(v_x)$ is the set of neighboring nodes of node v_x, $w\left(v_x, v_y\right)$ represents the sum of weights of the hyperedges corresponding to node v_x and node v_y, calculated as follows:

$$w\left(v_x, v_y\right) = \sum_{e \in E(v_x, v_y)} w(e) * \left(w(v_x, e) + w\left(v_y, e\right)\right) \tag{13}$$

Hyperedge set $E\left(v_x, v_y\right) = \{e | e \in E, v_x \in e, v_y \in e\}$, Each hyperedge e in the set contains two nodes, v_x and v_y.

$$\mathrm{SLC}(v_i) = \frac{l(v_i)}{\sum_{j=1}^{m} l(v_j)} \tag{14}$$

$$l(v_i) = \sum_{p \in path(v_i, N)} \sum_{e \in p} w(e) \tag{15}$$

Among them, $path(v_i, t)$ represents a path with a step size of N starting from node v_i, E is a hyperedge in path p.

The betweenness centrality index considers the global properties of ordinary networks, characterizes the control power of nodes over network traffic in the shortest path, and its value is the proportion of target nodes passing through the shortest path among all nodes. Extend the betweenness centrality index to weighted hypernetworks and construct a global centrality index, defined as follows:

$$\mathrm{GC}(v_i) = \sum_{j=1}^{m} \sum_{k=1}^{n} \frac{g_{jk}(v_i)}{g_{jk}},\ j \neq k \neq i \tag{16}$$

Among them, a hyperpath is a sequence of hyperedges composed of multiple hyperedges, g_{jk} is the number of shortest hyperpaths between node v_j and node v_k, $g_{jk}(v_i)$ is the number of shortest paths between node j and node k passing through node v_i.

4 EXPERIMENT

4.1 Technology Co-Occurrence Hypernetwork Construction

In this experiment, the global high-quality patent database Incopat was used as the data source, and keywords such as "CCUS", "Carbon Capture", "Carbon Utilization", and "Carbon Storage" were used to search for patent literature in the CCUS field published from January 2004 to September 2023. After removing duplicate and invalid literature, a total of 4476 patents were obtained. After obtaining the CCUS domain patent set, IPC fields were extracted from each patent and divided into primary IPC and secondary IPC. A technical co-occurrence hypernetwork was constructed using the multivariate co-occurrence relationship between IPC, resulting in a total of 3641 technical nodes and 2778 hyperedges. The corresponding visual graphs for some nodes are shown in Figure 3. In order to better evaluate the effectiveness of the method, two sub networks of different sizes, hypernet-1 and hypernet-10, were extracted from the hypernetwork based on the number of patent citations, as shown in Table 1. Among them, Hypernet k indicates that the patent corresponding to each hyperedge in the hypernetwork has been referenced by at least k other patents. The ratio of the number of relational nodes is used to measure the density of a hypernetwork. The larger the value, the denser the network, and vice versa, the sparser the network. The structural feature matrix of subnet hypernet-k is directly generated using the hypergraph deep learning library DHG, while the node feature matrix is trained by a five layer deep autoencoder. The number of neurons in each layer is set to 1000, 100, 15, 100, and 1000, with a total of 100 iterations of training.

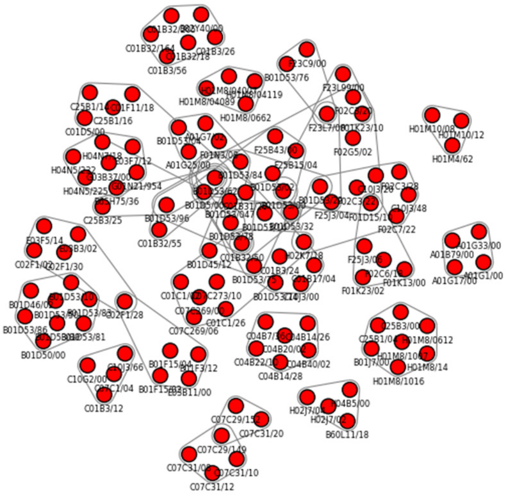


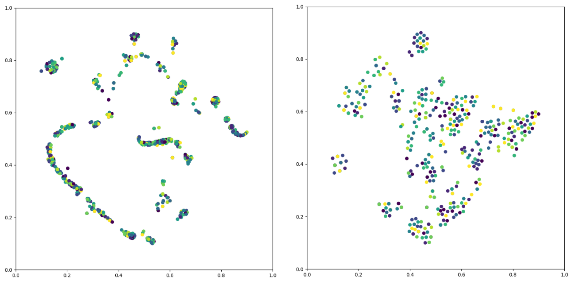
Figure 3: Technology co-occurrence hypernetwork visualization graph.

Table 1: Sub-network.

Hypernetwork	hypernet-1	hypernet-10
Number of patents	1379	228
Number of technical nodes	1506	440
Number of binary relationships	211	47
Number of multiple relationships	713	125
Relationship and node quantity ratio	0.61	0.39

4.2 Technology Clustering

When training the technical node vectors, a hypernetwork embedding model HGNN is constructed using the DHG library. The optimal multivariate relationship structure, model architecture, and training parameters are automatically searched through the Optuna library. The number of hidden layers, hidden layer dimension, weight attenuation, and learning rate are set to 5, 15, 0.0001, and 0.008, respectively. To verify the effectiveness of HGNN, t-SNE[37] was used to map the technical node vector to a two-dimensional Euclidean space. Nodes with similar local structural features are closer in this space, as shown in Figure 4.



hypernet-1 hypernet-10

Figure 6: Visualization of technical node vectors.

4.3 Core Technology Discovery and Analysis

To verify the effectiveness of the method proposed in this article, our experiment takes network efficiency and the rate of decrease in maximum connectivity coefficient as evaluation indicators, and compares the core technology discovery index CT based on hypernetworks and the recognition index degree centrality, weighted centrality and structural holes based on ordinary network structures Comparative analysis will be conducted using k-kernel analysis, degree centrality, clustering coefficients, etc. Among them, The CT index removes technology nodes with centrality values of Top-5 from each technology cluster in hypernet-k, and calculates the decline rate of two evaluation indicators. Based on the ordinary network structure, the indicators need to first convert the hypernetwork-k to the ordinary network structure before calculating the descent rate. The specific process is as follows: 1) For the hyperedge of the binary relationship $(\mathrm{IPC}_1, \mathrm{IPC}_2)$, Directly construct its two nodes as edges of a regular network. (2) For the hyperedges of n-ary co-occurrence relationships $(\mathrm{IPC}_1, \mathrm{IPC}_2, \dots, \mathrm{IPC}_\mathrm{n})$, If n>2, combine the technical nodes in pairs and decompose them into C_n^2 ordinary edges; If n<2, discard the hyperedge. (3) Sort nodes based on indicator values, remove the same number of nodes as the CT indicator, and calculate the descent rate. The network efficiency and the decrease rate of maximum connectivity coefficient of the network are calculated as follows:

$$\nabla \mathrm{E} = \frac{\mathrm{E_{before}} - \mathrm{E_{after}}}{\mathrm{E_{before}}} \tag{17}$$

$$\mathrm{E}_1 = \frac{1}{m(m-1)} \sum_{i,j \in V; i \neq j} \frac{1}{d_{ij}} \tag{18}$$

$$\mathrm{E}_2 = \frac{l}{m} \tag{19}$$

Among them, E_1 represents network efficiency, and its value is the average of the reciprocal sum of the shortest paths between nodes, E_2 is the maximum connectivity coefficient. M is the number of nodes, d_{ij} is the shortest distance between node i and node j, L is the number of nodes in the largest connected subgraph. The larger the ∇E, the greater the decrease in network efficiency and maximum connectivity coefficient, indicating that the deleted nodes have higher importance and influence. The network efficiency of hypernet-k and the decrease rate of maximum connectivity coefficient are shown in Table 2.

Table 2: Network efficiency and connectivity coefficient decline rate.

Centrality indicators	Network efficiency decline rate		Maximum connectivity coefficient decrease rate	
	hypernet-1	hypernet-10	hypernet-1	hypernet-10
Degree centrality	0.21	0.15	0.35	0.34
Convergence factor	0.24	0.17	0.34	0.37
Weighted centrality	0.34	0.23	0.51	0.48
K-kernel analysis+degree centrality	0.38	0.25	0.57	0.51
CT	0.41	0.27	0.63	0.54

5 CONCLUSIONS

In response to the limitation that traditional technology co-occurrence networks can only describe binary co-occurrence relationships, this paper extends ordinary networks to a hypernetwork structure to model complex multivariate co-occurrence relationships between entities in a more flexible pattern. To more accurately perceive different types of core technologies, the HGNN model is used to fuse the structural features and node attribute features of the hypernetwork to generate technical node vectors, and FCM fuzzy clustering is used to obtain technical clusters for different topics. Due to the tendency of traditional centrality metrics to fall into local optima and not be applicable to hypernetwork structures, we extend metrics based on ordinary network structures to hypernetwork structures and identify core technology nodes in technology clusters from three dimensions: local, semi local, and global.

REFERENCES

Liu p p, Wang l. (2022). Research Progress of Emerging Technology Identification from the Perspective of Relational Network. Library and Information Service.

Kajikawa Y, Yoshikawa J, Takeda Y, et al. (2022). Tracking emerging technologies in energy research: Toward a roadmap for sustainable energy Technological Forecasting & Social Change.

Cho T S, Shih H Y. (2008). Patent citation network analysis of core and emerging technologies in Taiwan: 1997–2008. Scientometrics.

Qi Y, Tang H, Shi J G. (2020). Research on Core Technology Identification Based on Small World Network Characteristics: A Case Study of Graphene. Journal of Intelligence.

Kong D J, Dong F, Chen Z J. (2021). Prediction of Emerging Technologies from the Perspective of Outlier Patents--Based on Bert Model and Deep Neural Networks. Library and Information Service.

Song K, Kim K, Lee S, et al. (2018). Identifying promising technologies using patents: A retrospective feature analysis and a prospective needs analysis on outlier patents. Technological forecasting and social change.

Wu C, Wang H Q, Wang S S. (2023). Research on the Identification and Prediction Methods of Frontier Technologies——Based on the Patent Topic Similarity Network and Technology Evolution. Forum on Science and Technology in China.

Huang L, Zhu Y H, Zhang Y. (2019). Research on Identification of Emerging Topics Based on Link Prediction with Weighted Networks. Journal of the China Society for Scientific and Technical Information.

Luo J L, Li M J, Jiang J, et al. (2017). On Recognition of the Core Technology Using Evidential Reasoning. Journal of Intelligence.

Dotsika F, Watkins A. (2017). Identifying potentially disruptive trends by means of keyword network analysis. Technological Forecasting & Social Change.

Dou Y X, Kai Q, Wang J M. (2023). Potential Disruptive Technology Identification Method Based on Graph Representation Learning. Journal of the China Society for Scientific and Technical Information.

Modelling of an Untrustworthiness of Fraudulent Websites Using Machine Learning Algorithms

Kristína Machová[a] and Martin Kaňuch
Department of Cybernetics and Artificial Intelligence, Technical University of Košice, Letná 9, Košice, Slovakia
kristina.machova@tuke.sk, martin.kanuch@student.tuke.sk

Keywords: Untrustworthy Content, Fraudulent Websites, Detection Models, Machine Learning, Neural Networks.

Abstract: This paper focuses on learning models that can detect fraudulent websites accurately enough to help users avoid becoming a victim of fraud. Both classical machine learning methods and neural network learning were used for modelling. Attributes were extracted from the content and the structure of fraudulent websites, as well as attributes derived from the way of their using, to generate the detection models. The best model was used in an application in the form of a Google Chrome browser extension. The application may be beneficial in the future for new users and older people who are more prone to believe scammers. By focusing on key factors such as URL syntax, hostname legitimacy, and other special attributes, the app can help prevent financial loss and protect individuals and businesses from online fraud.

1 INTRODUCTION

The Internet gives us many advantages, but there are also disadvantages and threats to this wonderful tool. Hacker attacks, stolen personal data and whitewashed accounts are often mentioned in the media. Many people still fall into the trap of scammers even though there is a huge effort by the authorities to stop these scams. What do these scams look like? Can they be stopped? How can such scams be avoided and how can internet users be alerted using a smart app? This article offers answers to these questions and possible solutions.

Phishing attacks have become a major concern in the digital age, posing significant threats to users' online security. Detecting and preventing phishing websites is crucial to protect individuals and organizations from falling victim to cybercrimes.

The Australian Government's website (*scamwatch.gov.au*) shows how the number of reports of fraud and the amount of money lost has evolved. In 2019, Australians lost more than $142 million to scammers. In 2020 it was $175 million in 2021 - $323 million and in the first 3 months of 2022 alone people lost $167 million to scams, more than in the whole of 2019. As we can see, this problem is only getting worse.

Large companies are also falling victim to internet fraud. For example, if a hacker gains access to an employee's email of a company and sends a phishing page that looks like a company login sent to the employee from a supervisor, the hacker can gain access to sensitive company information and the company can suffer major financial losses. Or the fraudster will spread a link to a page that looks like a legitimate news site and inform the public about a false event, which in turn will negatively affect stock growth (Ciampaglia, 2018).

The results of our research into the ability to model fraudulent behaviour on the web have been used to develop an application that can detect that a user is on a fraudulent website and inform the user if the application service is activated. We think that this application will be especially helpful for new Internet users and older people, who are most often victims of such scams and are the most vulnerable.

2 FRAUDULENT WEB SITES

The Internet has always been relatively secure in the realm of websites run by large companies, well-known brands and other technology giants. Payment gateways on online stores are very well secured with

[a] https://orcid.org/0000-0002-7741-4039

Machová, K. and Kaňuch, M.
Modelling of an Untrustworthiness of Fraudulent Websites Using Machine Learning Algorithms.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 218-225
ISBN: 978-989-758-716-0; ISSN: 2184-3228

two/three and multi-factor authentication. But sometimes it happens that we get to a website that pretends to be trustworthy but a fraudster has created an exact copy of the site. We want to log in to our account but logging in doesn't work and our password and email have just been sent to the scammer. This method of creating a fraudulent site is called spoofing in English. Nowadays, DDoS attacks became a real problem. The number of DDoS attacks in 2021 has been recorded as high as 9.75 million (Vermer, 2021). Although DDoS attacks are more frequent, modern servers can handle them more easily than in the past.

In 2021, hackers managed to obtain just one single password to the system of the US oil pipeline company Colonial Pipeline (Turton, 2021). The hackers gained access to the system after an employee entered the password to a fraudulent website posing as the company's VPN. The hackers then locked down the entire system using ransomware and demanded a ransom of 75 bitcoins ($4.4 million at the time).

In July 2020, Twitter employees were the target of a phishing attack, and hackers managed to gain access to the accounts of many celebrities as Elon Musk, Bill Gates, and shared the message that if you send Bitcoin to a certain Bitcoin address, your deposit will be doubled (Leswing, 2021). The scam was also shared by hacked accounts of well-known financiers such as Mike Bloomberg and Warren Buffet. This was an example of a scam called a Ponzi scheme.

Factors such as page load time, SSL protocol and contact details play an important role in identifying a fraudulent site (Fedorko, 2020). If the loading time of a web page is longer than 5 seconds, it causes a decrease in the credibility of the page. According to the latest rules, all websites should have SSL. It is the "https:" at the beginning of the URL. The presence of SSL increases the credibility of the website. Also, visibly accessible contact information - phone number, e-mail address, brief information about the company, for example, physical address, ID number, etc. increase the credibility of the website.

2.1 Related Works

Several studies have focused on what fraudulent sites have in common and how big the differences are between phishing sites, fraudulent payment gateways, fraudulent online stores and sites that pretend to be legitimate news organizations. For example, one of the common features that fraudulent sites have in common are invalid certificates and many buttons with broken links (Fedorko, 2020). In this study a descriptive statistic, multiple linear regression and structural equation modelling were used.

Other research, which worked with a dataset of phishing websites (Hannousse, 2021), discusses an importance of the syntax of URLs, i.e., how many special characters are in a link, how long the link is, how many times the www subdomain is in the link, whether the link contains the name of a globally known brand, and also whether the domain is registered at all and if so what is its age. These are features that we can more easily extract and preprocess for machine learning models. In this study, following machine learning were used for detection models training: logistic regression, random forest and support vector machines. The best performing model was learned using random forests method.

In 2013, research was conducted where 2046 participants decide whether or not the website displayed is trustworthy on a scale of 1-5. Those participants who very frequently ranked websites with the number 5 or the number 1 were often the most wrong in their decisions (Rafalak, 2014). In the study, descriptive statistics were used for estimated psychological traits levels. The results of this research are helpful in designing a method to detect fraudulent websites.

Nowadays, more and more user-generated content is hosted on web servers that belong to a small group of giant technology companies. This trend is leading to a centralized web with many problems. These could be addressed by decentralizing the web, which has the potential to ensure that the end-user always knows that the website they are currently on is from a legitimate source or not. A study (Kim, 2021) proposes a blockchain-based way of operating such a decentralized web.

Another way to prevent phishing and password leaks is by using blockchain encryption of messages and communications in companies between company servers when logging into the system. If an employee sends a login key or password to a corporate system, the blockchain ensures through a stored hash that only the target corporate server can read the content of the message - i.e. the password (Cai, 2017). In this case, it cannot happen that the content of the message - the password to the system, can be read by a hacker who sent a phishing website to the employee.

The study (Rutherford, 2022) demonstrates that the machine learning approach is viable with validation accuracy ranging from 49 to 86%. The support vector machine was able to predict whether a cadet would be compromised upon receipt of a phishing attack with a 55% accuracy while a recall score was 71%. On the other hand, logistic regression model had the highest 86% accuracy while maintaining a recall score only of 16%.

In the paper (Aljabri, 2022), in addition to the classic machine learning methods, it was also used a deep learning. For classification performance in identifying phishing websites, random forest algorithm achieved the highest accuracy.

The article (Alnemari, 2023) presents experiments with phishing detection models learned using artificial neural networks, support vector machines, decision trees, and random forest techniques. Their results show that the model based on the random forest technique is the most accurate.

The analysis of related works in the field showed, that blockchain encryption and descriptive statistic are often used. From machine learning methods mainly logistic regression, random forest and support vector machines were used. So we have decided to use logistic regression from statistical methods of machine learning and random forests as very successful method of ensemble learning, and two other methods of ensemble learning - gradient boosting and ADABoost. We also experimented with neural networks, as they have recently been successful in solving a wide range of problems.

3 USED METHODS

3.1 Classic Machine Learning

We have used one of methods of regression analysis namely logistic regression (LR) as the classic statistic machine learning method suitable for detection models generation on numerical, non-text data. The LR is a technique to estimate parameters of a logistic model. Logistic model is a model where linear combinations of independent variables are transformed using a specific type of logistic function, mostly a sigmoid function (Brownlee, 2023).

We also tested approaches based on ensemble learning such as boosting (gradient boosting - GB and XGBoost – extreme boosting) and random forest (RF). The ensemble learning is based on the idea of combination of several weak prediction models into one stronger model for final decision.

The GB provide this by iterative minimizing the loss function. The algorithm generates a set of decision trees where each tree is trained to predict the difference between the predicted and true values (i.e., residual values) of the previous tree. The algorithm then calculates the residues of the predictions and trains a new decision tree to predict these residues. This process is repeated several times. The final prediction is obtained by voting or averaging the results of particular trees. One of the benefits of GB regression is its ability to handle a wide variety of data types. In addition, it is known for its high accuracy and robustness, as well as its ability to process high-dimensional data. However, it also has some limitations. One is that it can be a computationally complex, especially for large datasets. Another limitation is that the model can be sensitive to the choice of hyper parameters such as learning speed and the number of trees in the file (Ke, 2017).

XGBoost (Extreme Gradient Boosting) is designed to improve the performance of traditional gradient boosting algorithms using a combination of regularization and parallel processing techniques. Regularization techniques such as L1 and L2 are used to prevent overtraining and improve generalization performance. XGBoost also uses a technique to trim trees, further reducing overtraining and improving model efficiency (Chen, 2016).

RF tries to minimize the variance by creating more decision trees in different parts of the same training data. Individual trees are de-correlated using a random selection of a subset of attributes. The method achieves the final classification by voting or averaging the results of particular trees. RF method is used mainly in cases where a limited amount of data is available, which significantly reduces memory requirements when generating many trees (Donges, 2024).

3.2 Neural Networks

We also experimented with very successful methods for generating of artificial neural networks, namely LSTM (long-short term memory), CNN (convolutional neural network) and MLP (multi-layer perceptron).

LSTM is the most known recurrent neural network, which can re-store information a longer time and that is why they can process longer sequence of inputs. LSTM networks are composed of repeating modules (LSTM blocks) in the form of a chain. The basis of LSTM is a horizontal line through which a vector passes between individual blocks. There are three gates (input, forget and output gate) in individual cells. These gates are used to remove or add information to the state of the block. Information passes through these gates, which are composed of neurons with a sigmoidal activation function. Depending on the value of the output on these neurons, certain amount of information passes through it (0 means that no information passes through the gate and 1 means that everything passes through the gate) (Ralf, 2019).

The basic building block of the CNN network is a convolutional layer that applies a set of filters to the input and extracts properties from it. Filters are learned by training as weights and other parameters. Filters are usually small in size to capture local patterns in the data. The output from the convolutional layer is then transmitted through a nonlinear activation function such as ReLU (Rectified Linear Unit) to insert nonlinearity into the network. In addition to convolutional layers, CNN networks typically include post-layer pooling, which reduces dimensionality and helps prevent overfitting. The most common type is max pooling, which selects the maximum value in a certain local region of the property map. The last layers of the CNN network are usually fully interconnected layers. The output from the last layer is used for prediction (He, 2016).

MLP consists of multiple layers of interconnected neurons, each performing a nonlinear transformation at its input. The basic building block of MLP is the perceptron, which receives a set of input values and produces a single output value. The output of a perceptron is determined by the weighted sum of its inputs that passes through a nonlinear activation function such as a sigmoid or ReLU function. MLPs are powerful models that can learn complex nonlinear input-output relationships (Goodfellow, 2016).

4 MODELS TRAINING AND TESTING

4.1 Dataset Description

We sourced data from (Hannousse, 2021). Dataset is also available to download from (Kaggle, 2024). This dataset contains 11 429 URL addresses, and has 87 attributes and information about its legitimacy; 50% of URLs were legitimate and 50% of URLs were linking to phishing websites. Dataset contains 3 types of attributes: attributes extracted from syntax of URL, attributes extracted from source code of a website and attributes that were queried using APIs. Dataset was split to 2 datasets. We also created 3rd dataset, which contained URLs with 34 extracted attributes. These URLs were of a browsing history of a simulated user. All datasets were loaded and adjusted in python programming environment – Spyder Anaconda using library *pandas*.

Dataset 1 was full dataset with all 87 attributes. This dataset was used to test the suitability of a diverse group of models from simple ones that use one function for classification to more complex ones that use network of functions for classification.

Dataset 2 was reduced to 34 attributes. The rest 53 attributes were removed from the dataset 2 because we couldn’t reliably extract their values from websites other than those in the kaggle dataset, so the final model learned also from those 53 attributes would not be sufficiently general while using for phishing detection on a random website. The methodology of filtering was based on indication of missing values during the extraction process. The reason for those extraction errors (missing values of attributes) could be that many of phishing websites are after some period blocked by internet provider, blocked by firewall or antivirus software or simply no longer exist. The chosen 34 attributes are detailed in the Appendix. We expected that after attributes reduction the performance of models could be negatively impacted but extraction of attributes values was much faster (from 30 seconds to less than one second).

Dataset 3 was used to test the functionality of the application. It contained 100 URLs of a simulated user, every URL contained 34 attributes (same as in dataset 2) and information of legitimacy of an URL.

4.2 Methodology

We chose a diverse type of models from the simplest ones to models based on learning an artificial neural networks. We have also used techniques of ensemble learning as random forest, gradient boosting, and XGBoost. All these different types of methods were chosen to see how well they can classify target attribute – phishing webpage. The above-mentioned models were trained at first on Dataset 1 (first round). Then three best methods were used for training on Dataset 2 with reduced number of attributes (second round) and only one best method was used for training on Dataset 3 (final round). The process is illustrated in Figure 1.

4.3 First Round of Training

Models were trained in python programming environment – Spyder Anaconda using library *scikit-learn*. Dataset 1 with 87 attributes was split into train and test dataset (ratio 80:20). In both sets the ratio of legitimate and phishing URLs was 50:50. Following models were trained and tested on this dataset: LR, GB, XGBoost, RF, LSTM, CNN, and MLP. The number of models (for example trees in RF) was set on 100. The same n_estimators=100 was set for GB and XGBoost. The hyper-parameters of the various NNs are presented in Table 3 and Table 4.

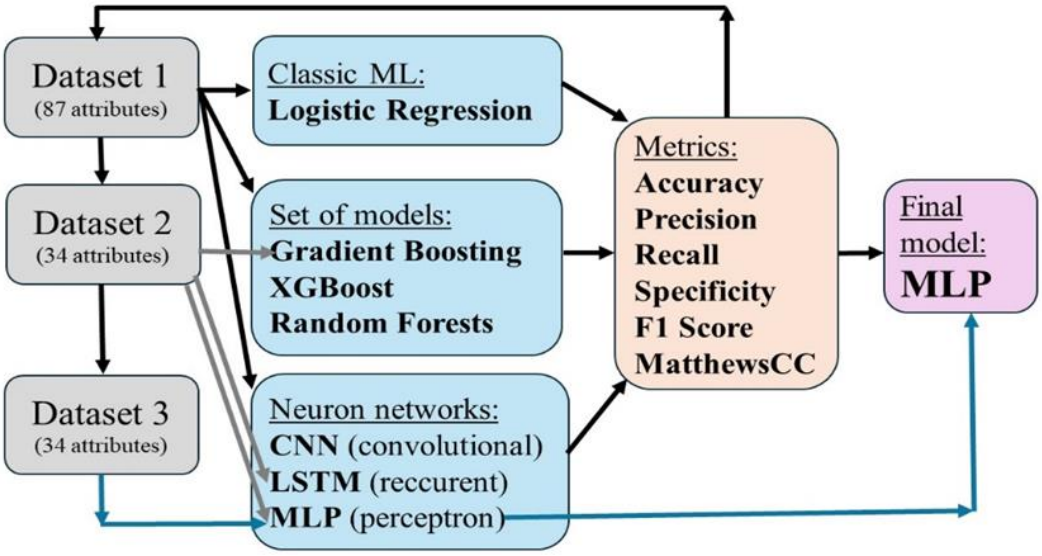


Figure 1: The methodology of models training.

After testing we evaluated the accuracy of every model and for next training (second round) chose only those, that achieved accuracy of more than 95%. We chose accuracy as metric to compare models because accuracy is most basic metric that evaluates general performance of models. This metric is also suitable for a balanced dataset.

Models that achieved expected accuracy were: GB, LSTM and MLP Neural Network. Other Metrics such as precision, recall, sensitivity, F1-score, and Matthew’s correlation coefficient were calculated for the tested models and are shown on Table 1 and 2. The mentioned three best methods were used for training on Dataset 2 with reduced number of attributes. The reason for this was that we wanted to see how much the accuracy of these models drops when we train them on less complete and complex data, but which are more suitable for use in real-world conditions, since it is not necessary to extract all 87 attributes values considered at the beginning of training process. In recognition of the phishing pages, we need to extract for them all values of all attributes, which were used in final model training.

Table 1: The results of experiments on Dataset 1 using LR, GB, XGB (XGBoost) and RF. The shortcut Matthews CC is Matthew’s Correlation Coefficient.

Method	**LR**	**GB**	**XGB**	**RF**
Accuracy	0.783	**0.952**	0.949	0.925
Precision	0.823	0.956	0.953	0.921
Recall	0.760	0.953	0.949	0.934
Specificity	0.810	0.951	0.948	0.915
F1 Score	0.790	0.954	0.951	0.927
Matthews CC	0.568	0.904	0.897	0.849

In Table1 and Table 2, there are bolded the results in accuracy of three best models. In Table 1 it is GB – gradient boosting, and in Table 2 they are LSTM – long-short term memory, and MLP - multi-layer perceptron.

Table 2: The results of experiments on Dataset 1 using CNN, LSTM, and MLP.

Method	**CNN**	**LSTM**	**MLP**
Accuracy	0.942	**0.954**	**0.963**
Precision	0.951	0.954	0.952
Recall	0.939	0.958	0.974
Specificity	0.946	0.950	0.953
F1 Score	0.945	0.956	0.963
Matthews CC	0.884	0.908	0.927

4.4 Second Round of Training

We determined that going forward, we would only train models on Dataset 2 that exceeded accuracy=0.95 in the first round on Dataset 1, and thus GB, LSTM and MLP were selected.

We used the scikit-learn library to train the **GB** model. In the first step of testing, normalized data entered the learning process. We defined the model as follows:

- model=GradientBoostingClassifier (n_estimators=100,
- learning_rate=0.1,
- max_depth=3).

We then trained and tested the model with the functions:

- model.fit(X_train, y_train)
- tested y_pred=model.predict(X_test).

Finally, we displayed the confusion matrix with the function cm=confusion_matrix(y_test, y_pred).

The LSTM model was trained using the keras library. In the first step, it was necessary to change the shape of the input normalized data to make it suitable for the LSTM model. This reshaping was done with the following functions:

- X_train=np.reshape(X_train,(X_train.shape [0],1,X_train.shape[1]));
- X_test=np.reshape(X_test,(X_test.shape[0], 1,X_test.shape[1])).

We defined the model architecture with the following functions (see Table 3).

Table 3: The architecture of the LSTM model.

Method	**LSTM**
model	"sequential()"
model_add (LSTM)	128 neurons
input_shape	1
return_sequencies	False
dense	64, activation="relu"
dense	1, activation="sigmoid"
optimizer	Adam, lr=0.001
model_compile	loss="binary_crossentropy"
optimizer_metrics	"accuracy"

For training the MLP model, we also used the scikit-learn library. In the first step of training, normalized data entered the learning process. We defined the model as follows (see Table 4).

Table 4: The architecture of the MLP model.

Method	**MLP**
model	"MLPClassifier"
solver	"adam"
alpha	0,01
hidden_layer_sizes	100, 100, 100, 100, 100
max_iter	100
random_state	44

The results of testing of all the models trained on Dataset 2 are shown in Table 5.

Table 5: The results of experiments on Dataset 2 using GB, LSTM and MLP.

Method	**GB**	**LSTM**	**MLP**
Accuracy	0.925	0.933	**0.948**
Precision	0.927	0.940	0.935
Recall	0.929	0.932	0.960
Specificity	0.920	0.933	0.935
F1 Score	0.928	0.936	0.947
Matthews CC	0.850	0.865	0.895

The predicted reduction in accuracy in the second round of training GB model was confirmed but the reduction in accuracy was minimal from 95.22% to 92.5%. The reduction in accuracy of the LSTM model in the training on Dataset 2 was also confirmed but the reduction in accuracy was also minimal from 95.40% to 93.25%, and similarly for MLP model was observed the reduction of accuracy, but the reduction was the lowest - only 1.57%. The final best MLP model has been retrained on Dataset 3 and used in our application for phishing detection.

5 PHISHING DETECTION APPLICATION

It is important for the proper functioning of the application to ensure smooth operation. The application cannot slow down the page load time and at the same time it must evaluate the page fast enough to inform the user about the threat. The threat information should not block the website but rather alert the user with a pop-up window, as an extension for Google Chrome, not as a new browser window.

If a website is flagged as fraudulent, because posing as a real news service or because this is a satirical site such as babylonbee.com or theonion.com, should the app block access to such sites? We think not. We just want to inform the user that what they are reading may not be true or the website is not a legitimate news service. So, we would like to stick to the principle of freedom of expression and just warn the user.

This work introduces a functional web application (Google Chrome extension) that can detect fake/phishing/spoofing websites and is intended to help people not to be fooled and robbed. Our application consists of two parts.

The first part – *Python script* - is launched automatically when browser is launched, user does not interact with this part at all. Python script is coded in a form of server that awaits input from browser – from extension using *REST – get* method. When script gets the URL sent from Chrome extension, MLP model trained on dataset 2 extracts all 34 attributes from URL and URL is evaluated. If URL is evaluated as phishing, number 1 is sent to extension, if URL is evaluated as legitimate, number 0 is sent to extension.

The second part – *Google Chrome extension* - is part that user will see and can interact with it. The application is situated on the top right corner of browser for good visibility, as shown in Figure 2.

The extension automatically sends newly opened URLs to Python script using *REST* method *put*. While waiting for response, extension changes its colour to yellow and text changes to "LOAD…" indicating that extension is waiting for response (shown in Figure 2). If 5 seconds passes and no response follows, extension changes its colour to orange and text changes to "ERR" indicating error message (shown in left part of Figure 3). If extension receives number 0 in 5 seconds time, extension changes its colour to green and text changes to "tick" symbol (shown in the middle part of Figure 3). If extension receives number 1 (python script evaluated URL as phishing site), the extension changes its colour to red and text changes to "X" symbol (shown in right part of Figure 3).

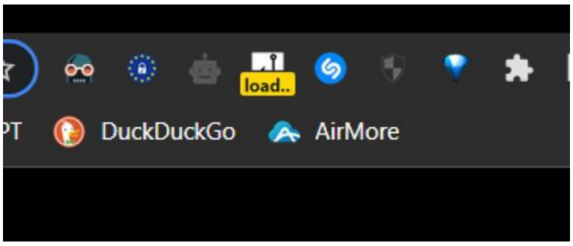


Figure 2: The illustration of our application as Google Chrome extension.

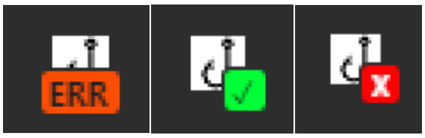


Figure 3: The illustration of different responses of the application.

6 CONCLUSIONS

Our study demonstrates the effectiveness of machine learning models in detecting phishing websites. We have trained and tested all selected models on the entire Dataset 1 - 87 attributes and expected accuracy of at least 95%. The models that achieved the required accuracy in the first round were MLP, LSTM and GB. In the second round of training, we used Dataset 2 reduced to 34 attributes (attributes that we can reliably extract from various websites outside the dataset). We re-trained and tested the models that advanced to this second round. We expected an accuracy of at least 90% and this was achieved for all three models in the second round. The highest accuracy was achieved by the Multilayer Perceptron (MLP) model at 94.75%. We used this model in developing a web application for real-time phishing detection. This solution can enhance online security and provide users with instant alerts regarding the legitimacy and trustworthiness of visited websites. This approach offers a proactive solution to combat phishing attacks and reduce the risk of cybercrimes.

ACKNOWLEDGEMENTS

This work was supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic, and the Slovak Academy of Sciences under no. 1/0685/21 and by the Slovak Research and Development Agency under the Contract no. APVV-22-0414.

REFERENCES

Aljabri, M., Mirza, S. (2022) Phishing Attacks Detection using Machine Learning and Deep Learning Models. 7th International Conference on Data Science and Machine Learning Applications (CDMA), Riyadh, Saudi Arabia, 2022, ps. 175-180.

Alnemari, S., Alshammari, M. (2023) Detecting Phishing Domains Using Machine Learning. Applied Sciences, Vol. 13, no. 8, 2023, ps. 4649-4649, ISSN 2076-3417.

Brownlee, J. (2023) Logistic regression for machine learning [online] 2023, Accessible [August 5, 2024].

Cai, C., Yuan, X., Wang, C. (2017) Towards trustworthy and private keyword search in encrypted decentralized storage. In IEEE International Conference on Communications (ICC17), 2017, ps. 1-7, DOI: 10.1109/ICC.2017.7996810.

Chen, T., Guestrin, C. (2016) Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, ps. 785-794, ISBN 978-1-4503-4232-2.

Ciampaglia, G. et al. (2018). Research Challenges of Digital Misinformation: Toward a Trustworthy Web. AI Magazine [online]. Vol. 39, no. 1,2018, ps. 65-74, ISSN 0738-4602.

Donges, N. (2024) Random forest: A complete guide for machine learning [online] 2024, Accessible [August 5, 2024].

Fedorko, I., Gburová, J. (2020). Selected Factors of Untrustworthiness during web pages using. [online]. 2020, ISSN 2453-756X.

Goodfellow, I., Bengio, Y., Courville, A. (2016) Deep learning. MIT press, 2016, captions 6-9, ISBN 9780262035613.

Hannousse, A. Yahiouche, S. (2021). Web page phishing detection. [online]. V3. Mendeley Data, 2021, DOI: 10.17632/c2gw7fy2j4.3.

He, K., Zhang, X., Ren, S., Sun, J. (2016) Deep residual learning for image recognition. In Proceedings of the

IEEE conference on computer vision and pattern recognition, 2016, ps. 770-778, DOI: 10.1109/CVPR.2016.90.

Kaggle (2024) Kaggle datasets [online] Accessible [May 25, 2024] <[https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset]>.

Ke, G. et al. (2017) Lightgbm: A highly efficient gradient boosting decision tree". In Advances in neural information processing systems, Vol. 30, no.1., 2017, ps. 1-9.

Kim, G.-H. (2021). Blockchain for the Trustworthy Decentralized Web Architecture. International Journal of Internet, Broadcasting and Communication [online]. Vol. 13, no. 1, ps. 26–36, 2021. ISSN 2233-7857.

Leswing, K. (2021). Hackers appear to target Twitter accounts of Elon Musk, Bill Gates, others in digital currency scam, CNBC [online] 2021, Accessible [May 25, 2024] <https://www.cnbc.com/2020/07/15/hackers-appear-to-target-twitter-accounts-of-elon-musk-bill-gates-others-in-digital-currency-scam.html>.

Rafalak, M., Abramczuk, K., Wierzbicky, A. (2014) Incredible: is (almost) all web content trustworthy? Analysis of psychological factors related to website credibility evaluation. In Proceedings of the 23rd International Conference on World Wide Web (WWW '14 Companion), New York, Association for Computing Machinery, 2014, ps. 1117-1122, ISBN 9781450327459.

Ralf, C., Rothstein-Morris, E. (2019) Understanding LSTM [online] 2019, Accessible [May 25, 2024] https://arxiv.org/abs/1909.09586.

Rutherford, S., Lin, K., Blaine, R.W. (2022) Predicting Phishing Vulnerabilities Using Machine Learning. IEEE SoutheastCon Conference, IEEE345 E 47TH ST, NEW YORK, NY 10017 USA, 2022, ps. 779-786.

Turton, W., Mehrotra, K. (2021). Hackers Breached Colonial Pipeline Using Compromised Password, Bloomberg [online] 2021-06-04, Accessible [May 25, 2024] https://www.bloomberg.com/news/articles/2021-06-04/hackers-breached-colonial-pipeline-using-compromised-password.

Vermer, B. (2021). Cybercriminals launched 9.75 million DDoS attacks in 2021. (IN)SECURE Magazine, Vol.70, ps. 54-55, 2021.

APPENDIX

The attributes in the Dataset 2 were following:

- ***length_url*** - is the length of the URL obtained by the function *len(url)*
- ***length_hostname*** is the length of the hostname obtained by *len(urlparse(url).netloc)*
- ***ip*** detects if the URL is in IP shape
- ***nb_dots*** detects the number of "." characters in URL
- ***nb_hyphens*** detects the number of "-" in URL
- ***nb_at*** detects the number of "@" in URL
- ***nb_qm*** detects the number of "?" in URL
- ***nb_and*** detects the number of "&" in URL
- ***nb_or*** detects the number of "|" in URL
- ***nb_eq*** detects the number of "=" in URL
- ***nb_underscore*** detects the number of "_" in URL
- ***nb_tilde*** detects whether the character " ~ " is present in the URL with the function *url.count('~')>0*
- ***nb_percent*** detects the number of "%" in URL
- **nb_slash** detects the number of "/" in URL
- ***nb_star*** detects the number of "*" in URL
- ***nb_colon*** detects the number of " : " in URL
- ***nb_comma*** detects the number of " , " in URL
- **nb_semicolumn** detects the number of " ; " in URL
- ***nb_dollar*** detects the number of "$" in URL
- ***nb_space*** detects the number of spaces in the URL
- ***nb_www*** detects the number of strings "www" in the URL with the function *url.count('www')*
- ***nb_com*** detects the number of "com" strings in the URL with the *url.count('com')* function
- ***nb_slash*** detects the number of "//" in URL
- ***https_token*** detects whether the string "https" is present in the URL
- ***ratio_digits_url*** detects the ratio between the number of digits and the number of other non-numeric characters in the URL
- ***abnormal_subdomain*** detects the abnormal shape of the subdomain "www" with the function *re.search('(http[s]?://(w[w]?|\d))([w]?(\d|-)))',url)*
- ***nb_subdomains*** counts the number of subdomains in the URL with the function *len(re.findall("\.",url))*
- ***prefix_suffix*** finds out if there are prefixes/suffixes in the URL with the function *re.findall(r "https?://[^\-]+-[^\-]+/",url)*
- ***shortening_service*** detects if the URL is shortened by services such as tinyurl, bit.ly, bit.do, etc.
- ***phish_hints*** detects if there are strings in the link that may point to a phishing link. Strings such as: "wp", "login", "css", "plugins", etc.
- ***domain_in_brand*** detects if there is a name string from the tag list in the URL. Tags like: "Pepsi", "Adidas", "Adobe", "Amazon", "Google", etc.
- ***website registered in WHOIS database,***
- ***domain_registration_length***
- ***websites PageRank***

Comparative Analysis of Topic Modelling Approaches on Student Feedback

Faiz Hayat[1][a], Safwan Shatnawi[2][b] and Ella Haig[1][c]
[1]*School of Computing, University of Portsmouth, Buckingham Building, Lion Terrace, Portsmouth PO1 3HE, U.K.*
[2]*Education Practice, International Research and Exchange Board, Amman, Jordan*
faiz.hayat@port.ac.uk, sshatnawi@irex.org, ella.haig@port.ac.uk

Keywords: Topic Modelling, BERT, LDA, LSA, NMF, Education.

Abstract: Topic modelling, a type of clustering for textual data, is a popular method to extract themes from text. Methods such as Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) and Non-negative Matrix Factorization (NMF) have been successfully used across a wide range of applications. Large Language Models, such as BERT, have led to significant improvements in machine learning tasks for textual data in general, as well as topic modelling, in particular. In this paper, we compare the performance of a BERT-based topic modelling approach with LDA, LSA and NMF on textual feedback from students about their mental health and remote learning experience during the COVID-19 pandemic. While all methods lead to coherent and distinct topics, the BERT-based approach and NMF are able to identify more fine-grained topics. Moreover, while NMF resulted in more detailed topics about the students' mental health-related experiences, the BERT-based approach produced more detailed topics about the students' experiences with remote learning.

1 INTRODUCTION

Machine learning tasks are typically divided into supervised and unsupervised learning (Berry et al., 2019). For textual data, one of the most used unsupervised methods is topic modelling, which is a type of clustering that extracts topics or themes from text (Zhao et al., 2021).

Three of the most popular methods for topic modelling are Latent Dirichlet Allocation (LDA) (Blei et al., 2003), Latent Semantic Analysis (LSA) (Deerwester et al., 1990) and Non-negative Matrix Factorization (NMF) (Lee and Seung, 1999). Since the arrival of Large Language Models (LLMs) in 2017 (Vaswani et al., 2017), pre-trained deep learning models like BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) have shown impressive results for unsupervised learning across many applications (e.g., (Abuzayed and Al-Khalifa, 2021; Egger and Yu, 2022; Sharifian-Attar et al., 2022)). Compared with other topic modelling approaches, BERT-based models have the following two key advantages: (1) because they were trained on large amounts of data, they have the capacity to encode complex semantic relationships, and (2) the ability to capture both left and right contexts, which accounts for the term "bidirectional" (Devlin et al., 2019).

In this paper, we compare a BERT-based topic modeling approach with LDA, LSA, and NMF to identify relevant topics from student feedback on their COVID-19 pandemic experience, focusing on mental health and remote learning.

The main contribution of the paper is a comparative analysis of topic modeling using LLMs like BERT against traditional methods. Few studies have explored this comparison, leaving the superiority of newer approaches uncertain. We investigate whether BERT provides an advantage over traditional methods in analyzing student feedback on pandemic experiences. Our study compares topics identified using BERT-based modeling with NMF, LDA, and LSA.

The rest of the paper is structured as follows: Section 2 reviews background and related work, Section 3 details the experimental setup, Section 4 presents the results, Section 5 compares methods and discusses findings and Section 6 concludes with future research directions.

[a] https://orcid.org/0000-0003-0249-4617
[b] https://orcid.org/0000-0002-5063-1295
[c] https://orcid.org/0000-0002-5617-1779

Hayat, F., Shatnawi, S. and Haig, E.
Comparative Analysis of Topic Modelling Approaches on Student Feedback.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 226-233
ISBN: 978-989-758-716-0; ISSN: 2184-3228

2 RELATED WORK

This section provides an overview of LDA, LSA, and NMF, reviews research on student feedback, and discusses evaluation approaches for topic modeling.

LDA is a probabilistic generative model widely used for topic modeling in natural language processing (NLP) (Blei et al., 2003). It assumes that each document in a corpus is a mixture of topics, and each topic is a distribution over words. LDA aims to uncover latent topics from a collection of documents by iteratively assigning words to topics and adjusting assignments to maximize the likelihood of the data.

LSA (Deerwester et al., 1990) is a technique used for dimensionality reduction and semantic analysis of textual data. It employs Singular Value Decomposition (SVD) to identify latent semantic structure in a corpus by capturing relationships between terms and documents, representing them in a lower-dimensional space for easier detection of semantic similarities.

NMF (Lee and Seung, 1999) is a dimensionality reduction technique widely used in natural language processing (NLP) and other fields. It decomposes a non-negative matrix into two lower-dimensional matrices, representing topics and document-topic distributions. NMF is applied to tasks such as topic modeling, document clustering, and feature extraction.

BERT (Grootendorst, 2022) is a pre-trained deep learning model developed by Google for natural language processing tasks. It excels in capturing contextual information bidirectionally, enabling it to understand the meaning of words in context more effectively than previous models. BERT has revolutionized NLP tasks by leveraging large-scale pre-training on vast text data and fine-tuning for specific downstream tasks, e.g., (Ding et al., 2023; Malladi et al., 2023).

Topic modelling has been used to analyse student feedback in many studies, e.g., (Buenano-Fernandez et al., 2020; Hujala et al., 2020; Sun and Yan, 2023). There have also been several studies investigating student experiences during the COVID-19 pandemic (e.g., (Oliveira et al., 2021; Stevanović et al., 2021; Waheeb et al., 2022). Many studies also employ BERT-based models with educational-related data (e.g., (Bai and Stede, 2023; Cochran et al., 2023; Sung et al., 2019)). However, to our knowledge, only one study has used BERT-based topic modeling on student feedback (Masala et al., 2021), and none have focused on students' COVID-19 experiences through open-text responses.

We are aware that other studies (e.g., (Müller et al., 2023; Wang et al., 2020; Xu et al., 2022)) have used BERT-based topic modeling to examine COVID-19 experiences in the general population, but they focus on social media data, not student responses from open-ended questionnaires. Therefore, these studies are not directly relevant to our research.

The one study we found using a BERT-based topic modeling technique (Masala et al., 2021) concentrated on examining student textual feedback at the course level. The researchers developed a tool that analyzed large volumes of student feedback, producing clusters of similar contexts and recurring keywords for each course. The processing pipeline involved extracting general evaluations, restoring diacritics using RoBERT (a Romanian BERT model), and performing keyword extraction with KeyBERT (fine-tuned for the Romanian language). To capture the context around these keywords, they utilized two methods: extracting sentences containing the keywords and using dependency tree traversal to gather related context. The extracted contexts were then grouped using K-Means clustering applied to BERT-generated embeddings.

In contrast, our study applies multiple topic modeling techniques to analyze survey responses related to mental health and remote learning during the COVID-19 pandemic. We explore several algorithms, including BERT-based embeddings, Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA), and Non-negative Matrix Factorization (NMF). Additionally, our study incorporates dimensionality reduction using UMAP and clustering with HDBSCAN to discover underlying topics in the survey data.

We now turn our attention to the evaluation of topic modeling techniques. In supervised learning, techniques are evaluated by comparing the predictions against a known ground truth, but in unsupervised learning, such ground truth is often absent, making evaluation challenging without human judgment. Although various metrics are used to evaluate topic modeling and clustering methods, performance can vary widely across techniques and data types (Doogan and Buntine, 2021; Harrando et al., 2021), and the validity of fully automated evaluations without human judgment has been questioned (Hoyle et al., 2021).

Some clustering/topic modelling techniques require as input the number of clusters/topics, while for others, the 'optimal' number emerges from the data. For the former, metrics like coherence scores (Abdelrazek et al., 2023; O'Callaghan et al., 2015), can help determine the optimal number of topics, but these also need human judgment (Doogan and Buntine, 2021). In our research, we combined coherence scores with human evaluation.

While the usefulness of BERT-based approaches for topic modelling has been shown for different types of education-related data, there has only been one study using a BERT-based approach on student feedback from open-ended questions and this study did not include a comparison with other topic modelling approaches. Our study contributes to a better understanding of the usefulness of BERT by providing the first comparative study for this type of data.

3 EXPERIMENTAL SETUP

In this section, we describe the data collection and preprocessing, as well as the process for topic modelling for each of the four investigated approaches.

3.1 Data Collection

Data collection for this research involved conducting a survey among students at a UK university in 2022. The aim was to assess the influence of the COVID-19 pandemic on students. The questionnaire included four open-ended prompts designed to clarify the particular difficulties students encountered regarding their mental well-being and remote learning during the pandemic: 'What challenges or issues regarding mental health did you face during the pandemic? What aspects, if any, did you struggle with?'; 'Please share any other comments/ opinions/ solutions about your mental health during the pandemic.'; 'What challenges or issues regarding remote learning did you face during the pandemic? What aspects, if any, did you struggle with?' and 'Please share any other comments/ opinions/ solutions about remote learning during the pandemic.'

Ethical approval was obtained from the university's Ethics Committee before distribution. The survey was distributed using email lists specific to each faculty, reaching out to a diverse group of students from different academic disciplines such as social sciences, humanities, business and law, and technology. The involvement in the survey was voluntary and respondents remained anonymous.

Responses from 340 participants included 696 submissions from the open-ended questions: 375 on mental health and 321 on remote learning. The sample size for topic modeling consisted of all 696 textual responses. We made this decision due to the prevalence of short answers and many students responding selectively to some questions and not others.

3.2 Data Preprocessing

Data preprocessing was conducted to prepare the textual data for analysis. Specifically, this process included the elimination of stop words, such as "the," "is," and "and", which are common words that provide little value in understanding the underlying themes of the text. Special characters and numbers not contributing to semantic analysis were also filtered out to refine the dataset and improve the quality of information fed into the topic modeling algorithm.

3.3 Topic Modeling Algorithms

Four topic modeling algorithms were utilized: a BERT approach described below, Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) and Non-negative Matrix Factorization (NMF). The implementation was carried out using Google Colab, a cloud-based environment integrated with Python.

The BERT-Based Topic Modelling Approach. The following steps were applied: 1) Obtaining document embeddings by utilizing the 'paraphrase-MiniLM-L6-v2' pre-trained model; 2) UMAP (Uniform Manifold Approximation and Projection) (McInnes et al., 2018) was used to reduce the dimensionality of the embeddings, improving visualization and clustering; 3) Performing clustering using the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) (McInnes et al., 2017) algorithm to generate the topics; 4) Using visualizations to analyze the 10 most frequent words per topic, response distribution across topics, and the dendrogram from the clustering algorithm; 5) Conducting qualitative analysis to validate topics by examining responses assigned to each topic.

Latent Dirichlet Allocation (LDA). The following steps were applied: 1) Using the Gensim library, a dictionary and a document-term matrix were created to represent the term frequency; 2) Applied LDA to the document-term matrix to infer the underlying topics and their word distributions; 3) Analyzing the resulting topics by examining the most probable words associated with each topic; 4) Conducting qualitative analysis to validate the topics by reviewing documents assigned to each topic.

LSA (Latent Semantic Analysis). The following steps were applied to derive the topics using LSA: 1) Creating a term-document matrix representing the frequency of terms in documents. 2) Applying SVD to the term-document matrix to decompose it into three matrices: a term-concept matrix, a diagonal matrix of singular values, and a concept-document matrix. 3) Analyzing the resulting concept vectors to identify latent semantic topics. 4) Conducting qualitative analysis to validate the topics by reviewing documents associated with each concept.

NMF (Non-Negative Matrix Factorization). The following steps were applied: 1) Vectorizing the preprocessed text data into a term-document matrix, where each row represents a document and each column represents a term. 2) Applying NMF to factorize the term-document matrix into two matrices representing topics and document-topic distributions. 3) Analyzing the resulting topics by examining the most prominent terms associated with each topic. 4) Conducting qualitative analysis to validate the topics by reviewing documents assigned to each topic.

4 RESULTS

As the BERT-based approach uses the HDBSCAN algorithm, the optimal number of topics emerges from the data; in our case, this was 13. LDA, LSA and NMF require a number of topics as an input. For these methods, to identify the optimal number of topics, as mentioned in Section 2, we chose the coherence score (Abdelrazek et al., 2023), which aggregates the coherence of each topic, measured as the semantic similarity between top words in the topic, in combination with human judgment. The highest coherence scores were obtained for 13 topics with LDA, 12 topics with LSA, and 16 topics with NMF, and our qualitative evaluation showed that for each method the topics were relevant and distinct from each other.

We conducted a deeper qualitative assessment of topics from all four algorithms and found the BERT-based approach and NMF yielded the most interesting results. Due to space constraints, we present detailed results for these methods and summarize LDA and LSA results for comparison in the next section.

The topics that resulted from the BERT-based approach are presented in Table 1. We grouped the topics into themes, analyzed in the following paragraphs.

As anticipated, we see that the subjects are arranged in relation to the two elements—mental health and remote learning—that were highlighted in the open-ended questions. Out of the thirteen topics, three (0 and 4-5) are related to mental health, two are related to both (1 and 6) and eight topics (2-3 and 7-12) are related to distant learning.

The application of the BERT-based modeling approach to mental health allows differentiation between several aspects, including anxiety (Topic 0), social isolation and loneliness (Topic 4), and the generic impact of the epidemic on mental health (Topic 5).

It is interesting to note that a more comprehensive picture of remote learning emerges, covering a wide range of topics, from the more general ones like the university experience in general (Topic 3) and the impact of the pandemic on the university experience (Topic 8), to the more specialised ones like concentration problems (Topic 2), internet connectivity (Topic 7), virtual communication (Topic 9), lecture formats (Topic 10), the experience of remote learning across various modules and courses (Topic 11), and the value of in-person communication (Topic 12).

Aspects of both remote learning and mental health are included in Topics 1 and 6. In Topic 1, motivation is discussed as a practical requirement for participating in remote learning, as well as a crucial component of mental health. The only positive topic is Topic 6, which describes the methods respondents use to preserve their mental health and academic motivation.

Table 1 displayed the number of textual instances per topic in the second column, with a relatively large variation. Topic 12 (face-to-face communication) has the fewest instances (14), while Topic 5 (the pandemic's effects on mental health) has the highest (88).

There are parallels between Topics 5 and 8, which discuss how the pandemic has affected mental health (Topic 5) and remote learning (Topic 8), respectively. Topics 10 and 11 share commonalities as well, as they both deal with challenges related to remote learning. The variations between the two topics highlight experiences related to lectures and teaching sessions in Topic 10 and broader experiences related to remote learning at the module or course level in Topic 11.

As mentioned in Section 3.1, responses to mental health (375) outnumbered those to distant learning (321). The fact that there are three topics about mental health and eight about remote learning suggests that while there are more different experiences with remote learning, there is a greater homogeneity of experiences with mental health. This further demonstrates the capacity of the BERT-based approach to discern between elements with subtle variations.

Table 2 presents the topics resulting from applying NMF. Similar to the BERT-based approach, the topics cover mental health, remote learning, or both aspects.

In terms of mental health, specific issues such as anxiety, eating disorders and depression are covered in Topic 1, dealing with uncertainty in Topic 3, and the generic impact of the pandemic on mental health in Topic 8. Topic 10 is also more generic, covering emotional well-being aspects, while Topic 11 is more specifically about social isolation challenges.

The topics covering remote learning aspects vary from more generic, about distance learning and the use of online tools (Topics 2, 6, and 9), to more specific issues such as motivation to study (Topic 12) and difficulties in grasping learning content (Topic 13).

Several topics cover both mental health and remote learning aspects: time management (Topic 0), motivational issues (Topics 4 and 15), lack of social interaction in remote learning (Topic 5), the impact of the pandemic on physical health (Topic 7), and the

Table 1: Topics extracted with the BERT-based approach; (Docs refers to the number of documents/responses for each topic).

No.	Docs	Topic Name	Topic Description	Keywords
0	34	Anxiety and Depression	Increased anxiety and depression disorders, leading to heightened awareness and impacts on mental health.	anxiety, depression, disorders, increased, eating, panic, depressed, still, health, aware
1	34	Motivation	Struggle to maintain motivation, resulting in challenges in staying focused and productive.	motivation, motivated, stay, keep, staying, lack, work, hard, struggle
2	26	Easily Distracted	Experience difficulty concentrating due to various distractions, affecting productivity and focus.	distracted, easily, concentrate, focused, couldnt, distractions, attention, skip, work, focus
3	19	University Experience	Mixed experiences during university, including success, failure, and uncertainty	university, year, felt, experience, well, think, failed, cheated, second, uni
4	74	Loneliness and Friendship	Loneliness and lack of social contact affecting mental well-being and interaction.	friends, loneliness, lonely, social, see, depression, lack, contact, isolation, able
5	88	Pandemic and Mental Health	Heightened awareness of mental health issues during the pandemic, affecting individuals and communities globally.	pandemic, health, mental, people, social, covid, anxiety, family, made, measures
6	19	Daily Routine	Recognizing the value of a consistent, positive daily routine for better mental health.	daily, good, routine, mental, health, work, home, day, weekly, sleep
7	16	Internet Connection Issues	Frustration and challenges from unreliable internet, affecting academic and personal tasks.	internet, connection, bad, unreliable, issues, lesson, poor, exams, could, found
8	27	Remote Learning	Adapting to remote learning challenges, including online lectures and assignments.	pandemic, remote, learning, working, lectures, really, time, away, home, im
9	19	Zoom Calls	Adjusting to the challenges and discomfort associated with online video calls, especially in educational and professional settings.	zoom, camera, calls, people, would, anyone, lessons, interacting, comfortable, answer
10	50	Online Lectures	Facing challenges with online lectures, including slower learning and engagement issues.	lectures, lecturers, questions, online, without, lecture, students, felt, slower
11	52	Remote Learning Experience	Reflecting on remote learning, its benefits, and drawbacks compared to traditional methods.	learning, remote, time, modules, learn, teachers, students, like, lectures, course
12	14	Face-to-Face Learning	Emphasizing the importance of face-to-face interaction in learning environments for effective communication and understanding.	face, union, learning, lower, guidance, lecturers, communication, important, facetoface, seeing

need for support during studies (Topic 14).

The distribution of responses per topic, unlike BERT-based approaches, NMF has a more balanced range, with the smallest topic having 25 responses (Topic 3) and the largest 60 (Topic 14). Eleven of the sixteen topics have between 40 and 50 responses.

5 COMPARISON AND DISCUSSION

To compare the four algorithms, we selected four themes that cover all the topics produced across all four solutions: remote learning and challenges, mental health and challenges, social issues and loneliness, and motivation and physical health. The topic distribution by theme is shown in Table 3, and Fig. 1 illustrates the theme proportions for each algorithm.

The BERT-based approach allocates the highest percentage (61.54%) of its thematic content to Remote Learning and Challenges, indicating its strong emphasis on analyzing issues related to remote education. Conversely, it allocates smaller proportions to Mental Health and Challenges (23.08%), Social Issues and Loneliness (7.69%), and Motivation and Physical Health Issues (7.69%), suggesting a relatively narrower focus on these domains.

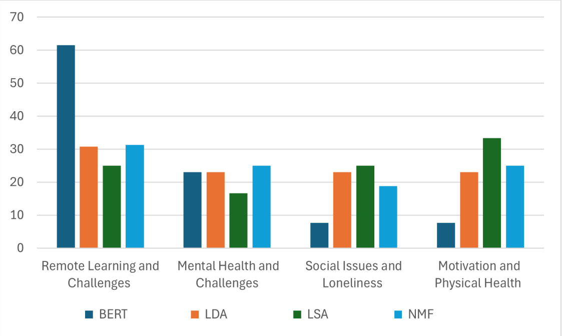


Figure 1: Comparitive Analysis based on Themes.

LDA (Latent Dirichlet Allocation) has the highest percentage to Remote Learning and Challenges (30.77%), with smaller proportions for Mental Health and Challenges, Social Issues and Loneliness, and Motivation and Physical Health, each at (23.08%).

Table 2: Topics extracted using NMF.

No.	Docs	Topic Name	Topic Description	Keywords
0	59	Time Management Struggles	Focuses on time management challenges worsened by pandemic-related work-life disruptions	struggled, focus, working, time, work, helped, day, home, pandemic, lot
1	30	Mental Health Challenges	Addresses severe mental health issues, including heightened levels of depression and anxiety due to societal pressures.	severe, reached, leaving, eating, society, disorder, house, increased, depression, anxiety
2	50	Remote Learning Preferences	Highlights difficulties adapting to remote learning and a preference for traditional face-to-face interactions.	difficulty, pace, better, lecture, preferred, tutor, prefer, remote, face, learning
3	25	Coping with Uncertainty	Discusses struggles with coping mechanisms during times of uncertainty, leading to feelings of loneliness and boredom.	uncertainty, email, change, coping, boring, struggling, extremely, covid, help, loneliness
4	49	Motivation Struggles	Focuses on maintaining motivation for completing coursework, with challenges in maintaining consistent effort.	module, getting, far, complete, week, whilst, went, struggled, motivation, work
5	40	Lack of Social Interaction	Explores the absence of social interactions in learning environments, leading to feelings of disconnection.	teacher, medium, unable, aspect, make, seeing, talking, people, interaction, social
6	40	Challenges with Distance Learning	Addresses difficulties in maintaining engagement and interaction in distance learning settings.	contact, distance, interaction, long, learning, student, issue, lecture, online, lack
7	45	Impact on Physical Health	Examines how disrupted routines and less exercise affected health during the pandemic.	low, exercise, issue, daily, pandemic, struggle, routine, good, mental, health
8	43	Impact on Mental Health	Examines worsening mental health from isolation, academic stress, and future uncertainty.	worse, life, depression, parent, job, caused, worried, stress, isolation, feel
9	45	Online Learning Experience	Evaluate online learning tools like Zoom, highlighting effectiveness and engagement issues.	use, useful, attention, session, zoom, know, people, lecture, online, class
10	50	Emotional Well-being	Addresses emotional challenges during university, such as stress, depression, and loneliness.	quite, stressed, teaching, university, feel, year, depressed, lonely, like, felt
11	40	Social Isolation Challenges	Explores challenges in maintaining social connections with family and friends due to prolonged social isolation.	future, kept, knowing, member, socialise, difficult, person, able, family, friend
12	35	Study Motivation	Discusses maintaining study motivation and focus amid distractions and coursework demands.	skill, studying, lesson, money, course, focused, stay, staying, motivated, hard
13	45	Understanding Course Material	Explores difficulties in grasping course material, especially under the distractions and pressures of lockdowns.	happened, understanding, lockdown, grade, thing, losing, understand, study, assignment, time
14	60	Academic and Financial Challenges	Addresses challenges in academics and financial stability, highlighting the need for institutional and peer support.	needed, course, poor, socialising, people, financial, harder, really, lecturer, support
15	40	Exam Preparation Challenges	Discusses challenges in preparing for exams due to distractions and unreliable internet connections.	exam, concentrate, difficult, bad, especially, learn, distracted, connection, easily, internet

In contrast, LSA (Latent Semantic Analysis) produces a unique thematic distribution in comparison with the other methods. It assigns a substantially higher percentage (33.34%) to Physical Health Issues and Motivation, indicating a strong emphasis on these two areas. There is no difference between Social issues and Loneliness (25%) and Remote Learning and Challenges(25%). It does, however, give Mental Health and Challenges a lower percentage (16.67%).

NMF allocated the highest percentage (31.25%) to Remote Learning and Challenges, equal percentages to Mental Health and Challenges, and Motivation and Physical Health (23.08%), and the lowest percentage (18.75%) to Social Issues and Loneliness.

Overall, NMF and LDA have the most balanced distributions across the four themes and can capture

Table 3: Algorithmic Topics Distribution.

Algorithm	Remote Learning and Challenges	Mental Health and Challenges	Social Issues and Loneliness	Motivation and Physical Health
BERT	Topic 2, 3, 7, 8, 9, 10, 11, 12	Topic 0, 5, 6	Topic 4	Topic 1
LDA	Topic 2, 8, 9, 12	Topic 0, 4, 6	Topic 3, 5, 10	Topic 1, 7, 11
LSA	Topic 3, 5, 7	Topic 0, 8	Topic 1, 2, 9	Topic 4, 6, 10, 11
NMF	Topic 0, 2, 6, 9, 13	Topic 1, 3, 8, 10	Topic 5, 11, 14	Topic 4, 7, 12, 15

at a good level of detail several distinct aspects. The BERT-based approach, on the other hand, has a more unbalanced distribution across the four themes but can capture more fine-grained issues related to remote learning. In particular, three topics identified by the BERT-based approach were not identified as separate topics by any of the other algorithms: internet connection issues (Topic 7), online calls (Topic 9), and face-to-face learning (Topic 12). By volume of responses, these are also among the smallest topics, with 16, 19 and 14 responses, respectively. From this point of view, the BERT-based approach may be better when a more fine-grained picture would be of interest.

For all approaches, we applied data preprocessing, as outlined in Section 3.2. There is very little empirical evidence concerning the use of textual data preprocessing when pre-trained LLMs are used. We applied the BERT-based approach with no preprocessing as well as the preprocessing mentioned in Section 3.2 and found more coherent results when using preprocessing, hence, we reported the results with preprocessing. This aligns with the view that preprocessing should still be considered for LLMs expressed in a recent review of text preprocessing (Chai, 2023).

6 CONCLUSION

This paper presents a comparative study using four topic modeling methods: BERT, LDA, LSA, and NMF, on student feedback in textual format about the mental health and remote learning students' experiences during the COVID-19 pandemic.

This study sought to determine the effectiveness of the BERT topic model compared to traditional approaches like NMF, LDA, and LSA. The results indicated that BERT provided deeper insights into remote learning challenges during the pandemic. While traditional methods produced similar results in mental health, social issues, isolation, and motivation, BERT showed clear advantages in topic understanding.

Our study found that all methods produced coherent topics covering various aspects, but BERT and NMF generated more interesting topics than LDA and LSA. NMF had a balanced response distribution, while BERT exhibited significant variation.

The two primary limitations of our study are the sample size and the post-epidemic data collection, which may have influenced students' recollections. We gathered 696 textual instances from 340 participants. Despite this small sample, all algorithms produced coherent topics.

Among the four algorithms, the BERT-based approach was least affected by the small sample size due to its extensive pre-training, which may explain its ability to capture more nuanced topics. Our research highlights the potential of BERT-based topic modeling for educational data. In the future we will explore alternative BERT models, like DeBERTa (He et al., 2020), known for its effectiveness in textual emotion recognition (Boitel et al., 2023), to capture more emotionally nuanced experiences.

REFERENCES

Abdelrazek, A., Eid, Y., Gawish, E., Medhat, W., and Hassan, A. (2023). Topic modeling algorithms and applications: A survey. *Information Systems*, 112:102131.

Abuzayed, A. and Al-Khalifa, H. (2021). BERT for arabic topic modeling: An experimental study on BERTopic technique. *Procedia computer science*, 189:191–194.

Bai, X. and Stede, M. (2023). A survey of current machine learning approaches to student free-text evaluation for intelligent tutoring. *International Journal of Artificial Intelligence in Education*, 33(4):992–1030.

Berry, M. W., Mohamed, A., and Yap, B. W. (2019). *Supervised and unsupervised learning for data science*. Springer.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022.

Boitel, E., Mohasseb, A., and Haig, E. (2023). A comparative analysis of GPT-3 and BERT models for text-based emotion recognition: performance, efficiency, and robustness. In *UK Workshop on Computational Intelligence*, pages 567–579. Springer.

Buenano-Fernandez, D., Gonzalez, M., Gil, D., and Luján-Mora, S. (2020). Text mining of open-ended questions in self-assessment of university teachers: An LDA topic modeling approach. *IEEE Access*, 8:35318–35330.

Chai, C. P. (2023). Comparison of text preprocessing methods. *Natural Language Engineering*, 29(3):509–553.

Cochran, K., Cohn, C., Hastings, P., Tomuro, N., and Hughes, S. (2023). Using BERT to identify causal structure in students' scientific explanations. *Artificial Intelligence in Education*, pages 1–39.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *The Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 4171–4186.

Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., Hu, S., Chen, Y., Chan, C.-M., Chen, W., et al. (2023). Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.

Doogan, C. and Buntine, W. (2021). Topic Model or Topic Twaddle? Re-evaluating Semantic Interpretability Measures. In *North American Association for Computational Linguistics 2021*, pages 3824–3848.

Egger, R. and Yu, J. (2022). A topic modelling comparison between LDA, NMF, Top2Vec, and BERTopic to demystify twitter posts. *Frontiers in Sociology*, 7:886498.

Grootendorst, M. (2022). BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794*.

Harrando, I., Lisena, P., and Troncy, R. (2021). Apples to apples: A systematic evaluation of topic models. In *The International Conference on Recent Advances in Natural Language Processing*, pages 483–493.

He, P., Liu, X., Gao, J., and Chen, W. (2020). DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Hoyle, A., Goel, P., Hian-Cheong, A., Peskov, D., Boyd-Graber, J., and Resnik, P. (2021). Is Automated Topic Model Evaluation Broken?: The Incoherence of Coherence. *Advances in neural information processing systems*, 34:2018–2033.

Hujala, M., Knutas, A., Hynninen, T., and Arminen, H. (2020). Improving the quality of teaching by utilising written student feedback: A streamlined process. *Computers & Education*, 157:103965.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Malladi, S., Gao, T., Nichani, E., Damian, A., Lee, J. D., Chen, D., and Arora, S. (2023). Fine-tuning language models with just forward passes. In *Advances in Neural Information Processing Systems*, volume 36, pages 53038–53075.

Masala, M., Ruseti, S., Dascalu, M., and Dobre, C. (2021). Extracting and clustering main ideas from student feedback using language models. In *International Conference on Artificial Intelligence in Education*, pages 282–292. Springer.

McInnes, L., Healy, J., Astels, S., et al. (2017). hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205.

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Müller, M., Salathé, M., and Kummervold, P. E. (2023). COVID-Twitter-BERT: A natural language processing model to analyse COVID-19 content on Twitter. *Frontiers in artificial intelligence*, 6:1023281.

Oliveira, G., Grenha Teixeira, J., Torres, A., and Morais, C. (2021). An exploratory study on the emergency remote education experience of higher education students and teachers during the COVID-19 pandemic. *British Journal of Educational Technology*, 52(4):1357–1376.

O'Callaghan, D., Greene, D., Carthy, J., and Cunningham, P. (2015). An analysis of the coherence of descriptors in topic modeling. *Expert Systems with Applications*, 42(13):5645–5657.

Sharifian-Attar, V., De, S., Jabbari, S., Li, J., Moss, H., and Johnson, J. (2022). Analysing longitudinal social science questionnaires: Topic modelling with BERT-based embeddings. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 5558–5567.

Stevanović, A., Božić, R., and Radović, S. (2021). Higher education students' experiences and opinion about distance learning during the Covid-19 pandemic. *Journal of Computer Assisted Learning*, 37(6):1682–1693.

Sun, J. and Yan, L. (2023). Using topic modeling to understand comments in student evaluations of teaching. *Discover Education*, 2:1–12.

Sung, C., Dhamecha, T. I., and Mukhi, N. (2019). Improving short answer grading using transformer-based pretraining. In *International Conference on Artificial Intelligence in Education*, pages 469–481.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Waheeb, S. A., Khan, N. A., and Shang, X. (2022). Topic modelling and sentiment analysis of online education in the COVID-19 era using social networks-based datasets. *Electronics*, 11(5):715.

Wang, T., Lu, K., Chow, K. P., and Zhu, Q. (2020). COVID-19 sensing: negative sentiment analysis on social media in china via BERT model. *IEEE Access*, 8:138162–138169.

Xu, W. W., Tshimula, J. M., Dubé, É., Graham, J. E., Greyson, D., MacDonald, N. E., and Meyer, S. B. (2022). Unmasking the Twitter discourses on masks during the COVID-19 pandemic: User cluster–based BERT topic modeling approach. *JMIR Infodemiology*, 2(2):e41198.

Zhao, H., Phung, D., Huynh, V., Jin, Y., Du, L., and Buntine, W. (2021). Topic modelling meets deep neural networks: a survey. In *The AAAI International Joint Conference on Artificial Intelligence 2021*, pages 4713–4720.

Reducing the Transformer Architecture to a Minimum

Bernhard Bermeitinger[1,*][a], Tomas Hrycej[2,*], Massimo Pavone[2,**], Julianus Kath[2,**]
and Siegfried Handschuh[2,*][b]

[1] *Institute of Computer Science in Vorarlberg, University of St. Gallen (HSG), Dornbirn, Austria*

[2] *Institute of Computer Science, University of St.Gallen (HSG), St. Gallen, Switzerland*

[*] *{firstname.lastname}@unisg.ch*, [**] *{firstname.lastname}@student.unisg.ch*

Keywords: Attention Mechanism, Transformers, Computer Vision, Model Reduction, Deep Neural Networks.

Abstract: Transformers are a widespread and successful model architecture, particularly in Natural Language Processing (NLP) and Computer Vision (CV). The essential innovation of this architecture is the Attention Mechanism, which solves the problem of extracting relevant context information from long sequences in NLP and realistic scenes in CV. A classical neural network component, a Multi-Layer Perceptron (MLP), complements the attention mechanism. Its necessity is frequently justified by its capability of modeling nonlinear relationships. However, the attention mechanism itself is nonlinear through its internal use of similarity measures. A possible hypothesis is that this nonlinearity is sufficient for modeling typical application problems. As the MLPs usually contain the most trainable parameters of the whole model, their omission would substantially reduce the parameter set size. Further components can also be reorganized to reduce the number of parameters. Under some conditions, query and key matrices can be collapsed into a single matrix of the same size. The same is true about value and projection matrices, which can also be omitted without eliminating the substance of the attention mechanism. Initially, the similarity measure was defined asymmetrically, with peculiar properties such as that a token is possibly dissimilar to itself. A possible symmetric definition requires only half of the parameters. All these parameter savings make sense only if the representational performance of the architecture is not significantly reduced. A comprehensive empirical proof for all important domains would be a huge task. We have laid the groundwork by testing widespread CV benchmarks: MNIST, CIFAR-10, and, with restrictions, ImageNet. The tests have shown that simplified transformer architectures (a) without MLP, (b) with collapsed matrices, and (c) symmetric similarity matrices exhibit similar performance as the original architecture, saving up to 90 % of parameters without hurting the classification performance.

1 INTRODUCTION

Recently, *Large Language Models* (LLMs) have shown impressive performance in producing complex text answers to given questions. Their outstanding feature is the massive size of parameter sets (up to billions). The rapidly growing parameter number has limited the possibility of developing such models (as well as objectively investigating their properties) to companies and institutions capable of making considerable investments in computing the model's parameters.

This is why it is of great interest to attempt to find more efficient configurations with fewer parameters without performance loss. A computing model with an excellent success record is based on the transformer architecture (Vaswani et al., 2017). Their success is due to an excellent ability to capture contextual information. Initially developed for language processing, transformers have also been successfully used in Computer Vision (CV). The analogy to language processing is the following: the semantics of individual words are determined by other words in the word sequence. Frequently, the basic units are not words but tokens (e.g., *n*-grams consisting of *n* consecutive letters). Since the *Vision Transformer* (Dosovitskiy et al., 2021), in an image, the tokens are represented by *patches* — typically square regions of pixels in the image. Other patches can influence or disambiguate a patch's conceptual meaning. For example, the environment in which an individual object is embedded in the image may disambiguate the identification of a specific bird or mushroom species.

The fundamental concept of the transformer is that

[a] https://orcid.org/0000-0002-2524-1850

[b] https://orcid.org/0000-0002-6195-9034

Bermeitinger, B., Hrycej, T., Pavone, M., Kath, J. and Handschuh, S.
Reducing the Transformer Architecture to a Minimum.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 234-241
ISBN: 978-989-758-716-0; ISSN: 2184-3228

of *attention* (Bahdanau et al., 2016). It is based on the insight that a particular token's semantics are influenced by its close relationships with other tokens. The tokens are encoded as real-valued vectors in a high-dimensional space (frequently around 1,000 dimensions or more). These vectors are called *embeddings*. The algebraic similarity between the embedding vectors measures the semantic proximity between the tokens. This similarity measure is the vector product or the cosine angle between the vectors. The weighting of tokens by such similarity measure is called attention, which, in analogy to human attention, focuses on relevant concepts. From the computational point of view, a transformer is a structure consisting of

- an algorithm for consideration of token context, the *attention mechanism*, and
- a *Multi-Layer Perceptron* (MLP) for nonlinear transformation of intermediary data.

Multi-Head Attention. For every transformer in the stack, the following processing is done by the attention mechanism (*multi-head attention* or *MHA*). The input of a training sample in the stack's s-th Transformer (out of their total number S) is a sequence of input vectors x_{si}. This sequence is transformed into an equally long sequence of output embeddings z_{si}. Each of them is, for given weights, a formally linear transformation

$$
\begin{aligned}
z_{si} &= \left(\sum_{j=1}^{i} a_{sij} x_{sj} W_s^V \right) W_s^O \\
&= \left(\sum_{j=1}^{i} a_{sij} x_{sj} \right) W_s^V W_s^O
\end{aligned}
\tag{1}
$$

i.e., a weighted average of input embeddings x_{si}, linearly transformed by matrix $W_s^V W_s^O$. The weight vectors $a_{si} = [a_{si1}, a_{si2}, \ldots, a_{sii}]$ are computed as

$$a_{si} = \text{Softmax}\left(s_{si}\right) \tag{2}$$

The vector argument of the Softmax() function measures the similarity between a present token x_Q, "the query" and another token x_K, "the key".

$$s_{sij} = x_{si} W_s^Q W_s^{KT} x_{sj}^T \tag{3}$$

This form of attention mechanism is referred to as *single-head*. A popular variant consists of an extension to multiple heads indexed by h:

$$z_{si} = \sum_{h=1}^{H} \left(\sum_{j=1}^{i} a_{shij} x_{sj} \right) W_{sh}^V W_{sh}^O \tag{4}$$

Each head has its separate matrices W_h^Q, W_h^K, W_h^V, and W_h^O. The weights are also computed separately as

$$a_{shi} = \text{Softmax}\left(s_{shi}\right) \tag{5}$$

and

$$s_{shij} = x_{si} W_{sh}^Q W_{sh}^{KT} x_{sj}^T \tag{6}$$

Multi-Layer Perceptron. The second component is a standard MLP with a single hidden layer, applied to each intermediary embedding z_{si}:

$$
\begin{aligned}
h_{si} &= f\left(z_{si} W_s^{(1)} + b_s^{(1)} \right) \\
y_{si} &= h_{si} W_s^{(2)} + b_s^{(2)}
\end{aligned}
\tag{7}
$$

with $f()$ being a nonlinear function, usually the *Gaussian Error Linear Unit (GELU)* (Hendrycks and Gimpel, 2023), weight matrices $W_s^{(1)}$ and $W_s^{(2)}$ as well as bias vectors $b_s^{(1)}$ and $b_s^{(2)}$.

(He and Hofmann, 2024) have investigated the possibilities of simplifying the transformer architecture. Their focus has been increasing the signal throughput through the network. The proposed changes primarily consist of modifying or omitting shortcut connections and normalizing layers. In addition, they have addressed the possibility of omitting matrices W^V and W^O. The last idea has also been implemented in our modifications proposed in Section 3.

Our focus is different: we intend to substantially reduce trainable parameters to accelerate the training and improve convergence.

2 TRANSFORMER WITHOUT THE MLP

The MLP requires the majority of the parameters to be fitted. This is justified by the argument that the MLP is the vehicle for implementing nonlinear mappings.

However, it can be argued that the first component, the attention mechanism, can also capture nonlinearities. It is the variable weights that make the mapping nonlinear. The argument of the Softmax() function is already a quadratic function of input tokens, and the function itself is nonlinear. Even if the Softmax() were linear, the multiplication of input tokens by the weights a_{sij} (which are quadratic in these tokens) would result in a cubic function of input tokens. The nonlinearity of Softmax() makes this mapping only more nonlinear.

So, a stack of S transformers is a chain of S at least cubic functions of the input, resulting in a function of polynomial order of at least $3S$. This makes clear that subsequent processing by an MLP is not the only nonlinear element of the processing. The extent of the task's nonlinearity cannot be assessed in advance. Still, the hypothesis that a reduced transformer without an MLP may cover the nonlinearity needs for some tasks is justified and can be validated by appropriate tests.

Without the MLPs, the transformer architecture can be described in more explicit terms. This is particularly the case if a single-head option is pursued.

3 SINGLE-HEAD CONFIGURATION

Although the matrices W_s^Q, W_s^K, W_s^V, and W_s^O can theoretically map the embedding vector to an arbitrary vector width, it is common to keep this width constant throughout the model, referring to the *model width N*. Then, in the case of a single head, these matrices are square. With square matrices, it is evident that $W_s^V W_s^O$ can be collapsed to a single matrix W_s^{VO}, and, analogically, $W_s^Q W_s^{KT}$ to W_s^{QK}. This saves 50 % of the attention module's parameters, from $4SN^2$ to $2SN^2$.

Concatenating the transformer-encoder layers without MLP leads to the following recursion:

$$\begin{aligned} y_{1i} &= \left(\sum_{j=1}^{i} a_{1ij} x_{1j} \right) W_1^{VO} \\ y_{2i} &= \left(\sum_{j=1}^{i} a_{2ij} y_{1j} \right) W_2^{VO} \\ &= \left(\sum_{k=1}^{i} a_{2ik} \left(\sum_{j=1}^{k} a_{1kj} x_{1j} \right) W_1^{VO} \right) W_2^{VO} \\ &= W_1^{VO} W_2^{VO} \sum_{k=1}^{i} a_{2ik} \sum_{j=1}^{k} a_{1kj} x_{1j} \\ & \ldots \end{aligned} \tag{8}$$

When stacking the attention modules, the matrices W_s^{VO} concatenate to their product over $s = 1, \ldots, S$. Then, they collapse into a single matrix

$$W^{VO} = \prod_{s=1}^{S} W_s^{VO} \tag{9}$$

Since every sum $\sum_{j=1}^{i} a_{sij}$ is equal to unity (as a result of the softmax operation), every successive transformer layer performs a weighted mean of stacked inputs x_{1j}.

The total number of parameters with S matrices W_s^{QK} and a single matrix W^{VO} is $(S+1)N^2$, only slightly more than 25 % of the original size without MLP. So far, all this is possible without losing any expressive power of the single-head transformer without MLP — only obsolete parameters are deleted.

In many NLP applications, the output of the last transformer of the stack is expected to produce an embedding of a word or a language token. These output embeddings can be expected to come from the space spanned by the input words or tokens. From this viewpoint, it may appear questionable to transform the input embeddings by matrices W_s^{VO} and to re-transform them back into the word embeddings. Then, it may be worth attempting to delete the value transformations. This has also been the proposal of (He and Hofmann, 2024), resulting in a simple weighted mean

$$z_{si} = \sum_{j=1}^{i} a_{sij} x_{sj} \tag{10}$$

The output embedding z_{Si} is a convex combination of input embeddings x_{1i}. In other words, it is a member of the convex set spanned by x_{1i}.

This concept has been implemented in the Keras framework by setting the matrices W_s^V and W_s^O to unit matrices. Collapsing $W_s^Q W_s^{KT}$ to W_s^{QK} has been reached by setting the matrix W^K to a unit matrix. The newly defined matrix W_s^{QK} replaces matrix W_s^Q.

4 MULTI-HEAD CONFIGURATION

The relationships of Section 3 are valid wherever the matrices W_{sh}^V, W_{sh}^O, W_{sh}^Q, and W_{sh}^K are square. This may also apply to multiple heads. However, it is usual to commit to a reduced dimension per head. With $H > 1$ heads, it is common to map the embedding vector to a narrower vector of width N/H, assumed to be integer.

In such cases, the matrices W_{sh}^V, W_{sh}^O, W_{sh}^Q, and W_{sh}^K are not square but of dimension $(N, N/H)$. Collapsing $W_{sh}^Q W_{sh}^{KT}$ to W_{sh}^{QK} is then no longer efficient since W_{sh}^{QK} is of dimension (N,N) and has thus N^2 parameters while W_{sh}^Q and W_{sh}^K together have $2N^2/H$, which is a smaller or equal number for $H > 1$.

Moreover, it is impossible to equivalently concatenate the value/projection matrices W_{sh}^{VO} to a unique product because of varying index h along various paths through the heads.

Nevertheless, omitting the W_{sh}^{VO} at all would have the same justification as for single-head configuration: the output embedding z_{Si} would become a convex combination of input embeddings x_{1i}, which can be expected to correspond to a meaningful word or token.

5 SYMMETRY OF SIMILARITY

The expression Eq. (3) measures the similarity between queries and keys. The general concept of characterizing similarity between vectors by their product is symmetric: a is equally similar to b as is b to a.

However, the similarity between a key and a query evaluated with the help of $x_{si}W^Q_{sh}W^{KT}_{sh}x^T_{sj}$ is asymmetric. This is because the matrices W^Q_{sh} and W^K_{sh} are potentially different.

This asymmetry leads to different similarities between x_{si} and x_{sj} in the roles of key and query: x_{si} is not as similar to x_{sj} as is x_{sj} to x_{si}. The vector x_{si} is also not the most similar to itself. The matrix product $W^Q_{sh}W^{KT}_{sh}$ is generally not positive definite, so it is not even guaranteed that the similarity of x_{si} to itself is positive.

The asymmetry can be deliberate and justified from some viewpoints. It is not a matter of course that the roles of queries and keys are symmetric. However, some of the mentioned properties can make its use harmful.

The symmetry can be guaranteed by simply setting $W^Q_{sh} = W^K_{sh}$. Then, half of the parameters dedicated to the query and key matrices can be economized. In the single-head case, the same effect is reached by a symmetric matrix W^{QK}_s, with identical parameters mirrored over the diagonal, i.e., $w^{QK}_{sij} = w^{QK}_{sji}$. Another possibility is to parameterize a lower triangular matrix T^{QK}_s and to multiply it by its transpose, getting

$$W^{QK}_s = T^{QK}_s T^{QKT}_s \tag{11}$$

This amounts to the well-known *Cholesky decomposition* (Cholesky, 1924) of a symmetric matrix.

With both methods, the number of parameters is $\frac{N(N+1)}{2}$ instead of N^2, or even $2N^2$ of the original version without collapsing W^Q and W^K.

The symmetry is implemented by reusing W^Q_{sh} as W^K_{sh}, omitting the use of W^K_{sh} at all.

6 SETUP OF COMPUTING EXPERIMENTS

The benchmarks for the evaluation have been chosen from the CV domain. They are medium-sized problems that can be run for a sufficient number of experiments. This would not be possible with large models such as those used in language processing.

For the experiments, two well-known image classification datasets MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky, 2009) were used. MNIST contains grayscale images of handwritten digits (0–9) while CIFAR-10 contains color images of exclusively ten different mundane objects like “horse”, “ship”, or “dog”. They contain 60,000 (MNIST) and 50,000 (CIFAR-10) training examples. Their respective preconfigured test split of each 10,000 examples are used as validation sets. While CIFAR-10 is evenly distributed among all classes, MNIST can be considered almost equally distributed.

An important criterion is that the training set size is sufficient for good generalization. The training size (as related to the number of model parameters) must be large enough for the model not to be underdetermined so that we can fairly assess the models’ performances. As a criterion for this, the overdetermination ratio of each benchmark candidate has been evaluated (Hrycej et al., 2023):

$$Q = \frac{KM}{P} \tag{12}$$

with K being the number of training examples, M being the output vector length (usually equal to the number of classes), and P being the number of trainable model parameters.

This formula justifies itself by ensuring that the numerator KM equals the number of constraints to be satisfied (the reference values for all training examples). This number must be larger than the number of trainable parameters for the system to be sufficiently determined. (Otherwise, there is an infinite number of solutions, most of which do not generalize.) This is equivalent to the requirement for the overdetermination ratio Q to be larger than unity.

The losses and accuracies in Table 1 show that the performance with 12 encoders is not superior to that with 6 encoders. The parameter set sizes with 12 encoders have been 563,242 with MLP and 198,100 without MLP. This is substantially more than 287,686 and 101,470, respectively, with 6 encoders. Consequently, the latter variant has been adopted as a baseline.

6.1 Results for MNIST

Following the arguments of Sections 2 to 5, the following reduced transformer variants have been tested:

- with and without an MLP in each transformer-encoder,
- with 1 and 4 heads,
- with the original matrix configuration as well matrix pair W^Q and W^K collapsed into one matrix, W^V and W^O omitted (one head variants only), and

Table 1: Results of 16 experiments on the two datasets MNIST and CIFAR-10 with 6 or 12 consecutive transformer encoders and 1 or 4 attention heads per encoder layer either with the default MLP inside each encoder layer or skipping it entirely. The loss and accuracy for the training and validation sets are reported after each model is trained for exactly 500 epochs.

Dataset	#Encs-#Heads	MLP?	Q	Train loss	Val. loss	Train. acc. [%]	Val. acc. [%]
MNIST	6-1	yes	2.15	0.0067	0.0747	99.78	98.38
	6-1	no	6.46	0.0277	0.1023	99.07	97.49
	6-4	yes	2.09	0.0018	0.0739	99.95	98.26
	6-4	no	5.91	0.0021	0.0912	99.92	98.29
	12-1	yes	1.08	0.0052	0.0652	99.81	98.71
	12-1	no	3.29	0.0117	0.0970	99.62	97.94
	12-4	yes	1.08	0.0025	0.0656	99.92	98.70
	12-4	no	3.29	0.0026	0.1002	99.93	98.10
CIFAR-10	6-1	yes	1.74	0.1533	2.2418	94.63	60.24
	6-1	no	4.93	0.9341	1.3590	66.16	55.30
	6-4	yes	1.74	0.1109	2.4033	96.01	60.46
	6-4	no	4.92	0.5621	1.6984	80.82	52.37
	12-1	yes	0.89	2.3026	2.3026	9.82	10.00
	12-1	no	2.52	0.5604	1.7219	79.48	54.06
	12-4	yes	0.89	0.0632	2.6379	97.92	58.02
	12-4	no	2.52	0.1787	2.3200	93.59	55.60

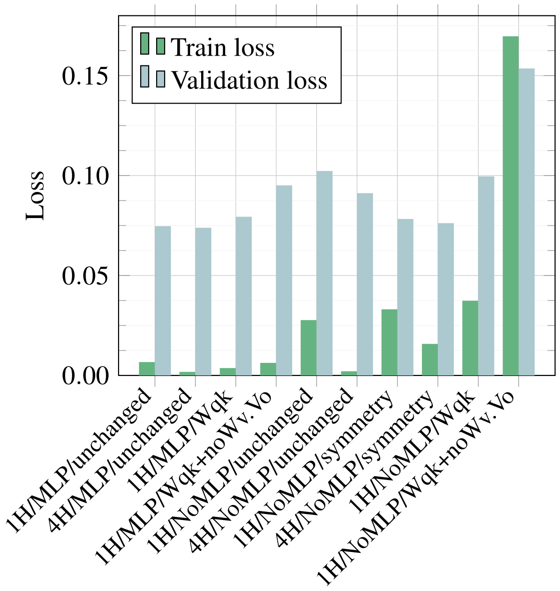


Figure 1: Training and validation losses attained by various reduced transformer-encoders with six encoder layers on MNIST.

- with asymmetric and symmetric similarity measures.

The variants depicted refer to the matrix options:

- *unchanged* corresponds to the original attention module matrix variety;
- *Wqk* variants use a single matrix for the product $W^Q W^{KT}$; these variants are only available for a single attention head, and their similarity measure is asymmetric as in the original version;
- *noWv.Vo* denotes omitting the value matrices W^V as well as the projection matrices W^O; also, these variants imply a single attention head and asymmetric similarity measurement;
- *symmetric* variants are committed to symmetric similarity measures; W^V and W^O are left untouched.

The performances of the individual variants are given in Table 2. For better comparability, the losses are additionally depicted in Fig. 1.

The following observations can be made:

- The original variants with MLPs perform better than those without MLPs on the training set.
- By contrast, their advance disappears on the validation set, particularly if the symmetric similarity metrics are used.
- The variant with asymmetric similarity without MLP is inferior to the analogical one with symmetric similarity.
- The minimum variant with query and key matrices W^Q, W^K collapsed to $W^{QK} = W^Q W^{KT}$ and additionally omitted value and projection matrices show a higher loss than other variants. This may be due to its dramatically reduced parameter number, which may lead to an insufficient capacity to capture nonlinearities.

As MNIST is a relatively easy benchmark, the accuracy results are very close to each other. The parameter numbers are substantially different. The symmetric variant without MLP has only about 25 % of the parameter number of the original, full variant

Table 2: Loss and accuracy for different variants of transformer-encoder modifications on MNIST: 1 or 4 heads, with or without the MLP, with a single W_{qk} matrix, no value and projection matrices, or a symmetric similarity measurement.

# Heads	MLP?	Modification	# Parameters	Q	Train loss	Val. loss	Train. acc. [%]	Val. acc. [%]
1	yes	unchanged	279,106	2.15	0.0067	0.0747	99.78	98.38
4	yes	unchanged	287,746	2.09	0.0018	0.0739	99.95	98.26
1	yes	Wqk	257,506	2.33	0.0037	0.0794	99.89	98.43
1	yes	Wqk+noWv,Vo	212,866	2.82	0.0063	0.0951	99.78	98.27
1	no	unchanged	92,890	6.46	0.0277	0.1023	99.07	97.49
4	no	unchanged	101,530	5.91	0.0021	0.0912	99.92	98.29
1	no	symmetry	69,910	8.58	0.0331	0.0783	98.85	97.80
4	no	symmetry	69,910	8.58	0.0158	0.0762	99.46	98.24
1	no	Wqk	70,570	8.50	0.0374	0.0996	98.70	97.60
1	no	Wqk+noWv,Vo	26,650	22.51	0.1697	0.1536	94.82	95.32

with MLP. The variant with collapsed matrices has about 33 % of the original parameters. The parameters include, in addition to the attention modules of all transformer-encoders, the embedding matrix reducing the image patch to the embedding vector.

The number of parameters has a strong effect on the generalization capability of the model. This can be quantified with the help of the overdetermination ratio from Eq. (12) in column Q of Table 2. The loss gap between the training and validation sets is the largest for the original version with Q close to unity while it shrinks towards the symmetric version without MLPs.

6.2 Results for CIFAR-10

The variants tested are analogical to those for MNIST. The losses and accuracies attained after 500 epochs are given in Table 3, the losses additionally in Fig. 2.

The result characteristics are similar to those for MNIST but more distinct:

- The original variant with MLP reaches the best training set loss but the worst validation set loss.
- Compared to the original variant, the reduced variants without MLP and with symmetric similarity are superior in generalization.
- This also applies to the variant with collapsed key and query matrices.
- Even the minimum variant with all considered matrix reductions (except for symmetry), whose parameter count is only a tenth of the original version with MLP, shows a better validation set performance than the original variant with all matrices and MLP.

The measured accuracies are roughly consistent with the losses on the training set. On the validation set, some of them follow, paradoxically, a different ranking. However, the fact that the loss, not the accuracy, is explicitly trained justifies the arguments via loss rather than accuracy.

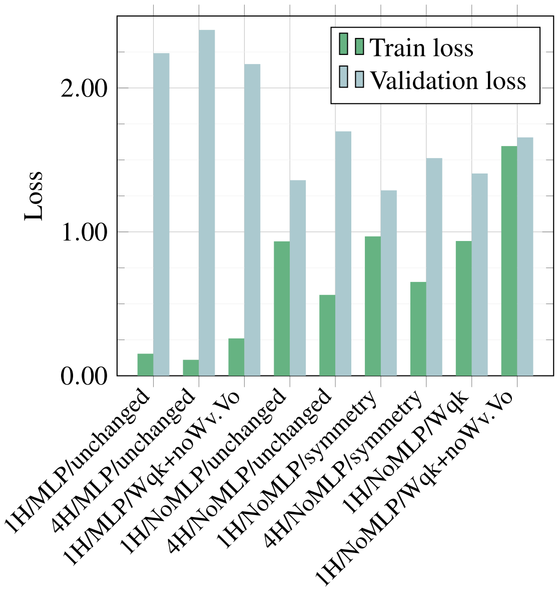


Figure 2: Training and validation losses attained by various reduced transformer-encoders with six encoder layers on CIFAR-10.

6.3 Trials with ImageNet

Several trials on the ImageNet dataset (Russakovsky et al., 2015) have been conducted to support the hypotheses with a larger benchmark. Unfortunately, the baseline run with the original transformer architecture, including MLP, has not been successful. In all trials, *Adam* failed to find a substantial improvement in the initial parameter state. By contrast, without MLP, it has been converging at least to a state with a moderate classification performance. This is why we cannot present a serious study on ImageNet. It can only be concluded that discarding MLP is helpful

Table 3: Loss and accuracy for different variants of transformer-encoder modifications on CIFAR-10: 1 or 4 heads, with or without MLP, with a single W_{qk} matrix, no value and projection matrices, or a symmetric similarity measurement.

# Heads	MLP?	Modification	# Parameters	Q	Train loss	Val. loss	Train. acc. [%]	Val. acc. [%]
1	yes	unchanged	287,686	1.74	0.1533	2.2418	94.63	60.24
4	yes	unchanged	287,746	1.74	0.1109	2.4033	96.01	60.46
1	yes	Wqk+noWv,Vo	221,446	2.26	0.2597	2.1659	90.53	54.98
1	no	unchanged	101,470	4.93	0.9341	1.3590	66.16	55.30
4	no	unchanged	101,530	4.92	0.5621	1.6984	80.82	52.37
1	no	symmetry	78,490	6.37	0.9686	1.2885	64.80	55.85
4	no	symmetry	78,490	6.37	0.6521	1.5125	76.10	55.52
1	no	Wqk	79,150	6.32	0.9364	1.4057	66.03	53.70
1	no	Wqk+noWv,Vo	35,230	14.19	1.5961	1.6565	40.52	39.17

for convergence. The proof that this variant's performance is acceptable is still pending, and further work will be required to provide it.

7 CONCLUSIONS AND LIMITATIONS

The experiments presented have shown limited utility of some parameter-extensive components of the transformer architecture. In particular, the following findings can be formulated:

- The MLP component is frequently presented as necessary for capturing nonlinearities in the modeled relationship. However, the inherent nonlinearity of the similarity measures seems powerful enough in many practical cases.
- While the classification performance without the MLPs is not significantly inferior to that with MLPs, a substantial benefit is saving the parameters. With model size N, the attention mechanism requires $4N^2$ parameters in the form of matrices W^Q, $W^K W^V$, and W^O. The size of the MLP is usually chosen as an integer multiple of h of the model size. Then, the MLP consists of weights and biases of two layers, with a total of $hN(N+1)+N(hN+1) = 2hN^2 + hN + N \approx 2hN^2$. If the multiple is $h = 4$, MLP has double the number of parameters as the attention mechanism. Consequently, omitting MLP reduces the parameters to 33 % of the original size.
- Symmetric similarity measures tend to perform better than asymmetric ones, with 50 % fewer query and key matrix parameters. This improvement may be reached by excluding undesirable freedoms, such as a token being dissimilar to itself. The parameter reduction can be expected to constrain the search for the optimum fit fruitfully.
- Collapsing the value and the key matrix into one is another possibility of reducing the parameter set of these matrices by 50 %.
- Omitting the value matrix W^V and the projection matrix W^O reduces the parameters of the whole attention module by 50 %. This variant has also been proposed by (He and Hofmann, 2024), with the observation of no significant performance loss in NLP benchmarks.
- Both preceding reductions amount to a reduction to 25 % of the original attention module size.
- In our experiments, the variants with the collapsed query/key matrices, omitted value, and projection matrices are slightly inferior for MNIST but equal for CIFAR-10. These minimum variants have less than 10 % of parameters compared with the classical transformers, including MLP. Compared to the architecture with 12 encoders, it is as little as 5 %.

The savings in computing time have been proportional to the savings in parameter numbers.

Our research has been limited to image processing benchmarks MNIST, CIFAR-10, and ImageNet. The experiments with the last benchmark have partially failed due to computing problems. Empirical evidence with the help of two medium-sized benchmarks and an incomplete test of a larger one is not satisfactory. This requests further research with more robust algorithms. There is considerable potential for second-order optimization methods such as the conjugate gradient algorithm of (Fletcher and Reeves, 1964), thoroughly described in (Press et al., 1992). This algorithm's convergence is excellent, but implementing the stopping rule in widespread packages seems to improve its ability to prevent early stops before reaching the minimum region.

Limitations to image processing suggest further extension. The proper domain of transformers is NLP. An obstacle to its investigation is the size of benchmark problems, so most published investigations consist of observing the performance of fine-tuning pre-trained models. To use pre-trained parameter sets, these fine-tuned models must be identical or almost identical to the pre-trained models. This makes the testing of different architectures difficult. A possibility is to use a large model used for pre-training as a *teacher* and a medium-sized model as *student*, mimicking its performance. This procedure, referred to as *knowledge distillation*, has been proposed by (Hinton et al., 2015) and used, e.g., by (Sun et al., 2019).

These will be important focuses soon.

REFERENCES

Bahdanau, D., Cho, K., and Bengio, Y. (2016). Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR*.

Cholesky, A.-L. (1924). Note Sur Une Méthode de Résolution des équations Normales Provenant de L'Application de la MéThode des Moindres Carrés a un Système D'équations Linéaires en Nombre Inférieur a Celui des Inconnues. — Application de la Méthode a la Résolution D'un Système Defini D'éQuations LinéAires. *Bulletin Géodésique*, 2(1):67–77.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, page 21, Vienna, Austria.

Fletcher, R. and Reeves, C. M. (1964). Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154.

He, B. and Hofmann, T. (2024). Simplifying Transformer Blocks. arXiv:2311.01906 [cs].

Hendrycks, D. and Gimpel, K. (2023). Gaussian Error Linear Units (GELUs). arXiv:1606.08415 [cs].

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the Knowledge in a Neural Network. arXiv:1503.02531 [cs, stat].

Hrycej, T., Bermeitinger, B., Cetto, M., and Handschuh, S. (2023). *Mathematical Foundations of Data Science*. Texts in Computer Science. Springer International Publishing, Cham.

Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images. Dataset, University of Toronto.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press, USA.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252.

Sun, S., Cheng, Y., Gan, Z., and Liu, J. (2019). Patient Knowledge Distillation for BERT Model Compression. arXiv:1908.09355 [cs].

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Efficient Neural Network Training via Subset Pretraining

Jan Spörer[1,*][a], Bernhard Bermeitinger[2,*][b], Tomas Hrycej[1,*], Niklas Limacher[1,**]
and Siegfried Handschuh[1,*][c]

[1] *Institute of Computer Science, University of St.Gallen (HSG), St.Gallen, Switzerland*
[2] *Institute of Computer Science in Vorarlberg, University of St.Gallen (HSG), Dornbirn, Austria*
*{firstname.lastname}@unisg.ch, **{firstname.lastname}@student.unisg.ch*

Keywords: Deep Neural Network, Convolutional Network, Computer Vision, Efficient Training, Resource Optimization, Training Strategies, Overdetermination Ratio, Stochastic Approximation Theory.

Abstract: In training neural networks, it is common practice to use partial gradients computed over batches, mostly very small subsets of the training set. This approach is motivated by the argument that such a partial gradient is close to the true one, with precision growing only with the square root of the batch size. A theoretical justification is with the help of stochastic approximation theory. However, the conditions for the validity of this theory are not satisfied in the usual learning rate schedules. Batch processing is also difficult to combine with efficient second-order optimization methods. This proposal is based on another hypothesis: the loss minimum of the training set can be expected to be well-approximated by the minima of its subsets. Such subset minima can be computed in a fraction of the time necessary for optimizing over the whole training set. This hypothesis has been tested with the help of the MNIST, CIFAR-10, and CIFAR-100 image classification benchmarks, optionally extended by training data augmentation. The experiments have confirmed that results equivalent to conventional training can be reached. In summary, even small subsets are representative if the overdetermination ratio for the given model parameter set sufficiently exceeds unity. The computing expense can be reduced to a tenth or less.

1 INTRODUCTION

Neural networks as forecasting models learn by fitting the model forecast to the desired reference output (e.g., reference class annotations) given in the data collection called the training set. The fitting algorithm changes model parameters in the loss function's descent direction, measuring its forecast deviation. This descent direction is determined using the loss function gradient.

The rapidly growing size of neural networks (such as those used for image or language processing) motivates striving for a maximum computing economy. One widespread approach is determining the loss function gradient from subsets of the training set, called batches (or mini-batches). Different batches are alternately used to cover the whole training set during the training.

There are some arguments supporting this procedure. (Goodfellow et al., 2016, Section 8.1.3) refers to the statistical fact that random standard deviation decreases with the square root of the number of samples. Consequently, the gradient elements computed from a fraction of $1/K$ training samples (with a given positive integer K) have a standard deviation equal to the factor $\sqrt{K}$ multiple of those computed over the whole training set, which seems to be a good deal.

Another frequent justification is with the help of stochastic approximation theory. The stochastic approximation principle applies when drawing training samples from a stationary population generated by a fixed (unknown) model. (Robbins and Monro, 1951) discovered this principle in the context of finding the root (i.e., the function argument for which the function is zero) of a function $g(x)$ that cannot be directly observed. What can be observed are randomly fluctuating values $h(x)$ whose mean value is equal to the value of the unobservable function, that is,

$$E\left[h(x)\right] = g(x) \tag{1}$$

The task is to fit an input/output mapping to data by gradient descent. For the parameter vector w of this mapping, the mean of the gradient $h(w)$ with respect to the loss function computed for a single train-

[a] https://orcid.org/0000-0002-9473-5029
[b] https://orcid.org/0000-0002-2524-1850
[c] https://orcid.org/0000-0002-6195-9034

Spörer, J., Bermeitinger, B., Hrycej, T., Limacher, N. and Handschuh, S.
Efficient Neural Network Training via Subset Pretraining.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 242-249
ISBN: 978-989-758-716-0; ISSN: 2184-3228

ing sample is expected to be equal to the gradient $g(w)$ over the whole data population. The local minimum of the loss function is where the gradient $g(w)$ (i.e., the mean value of $h(w)$) is zero. (Robbins and Monro, 1951) have proven that, under certain conditions, the root is found with probability one (but without a concrete time upper bound).

However, this approach has some shortcomings. For different batches, the gradient points in different directions. So, the descent for one batch can be an ascent for another. To cope with this, (Robbins and Monro, 1951) formulated the convergence conditions. They require that if the update rule for the parameter vector is

$$w_{t+1} = w_t - c_t h(w_t) \tag{2}$$

which corresponds to the usual gradient descent with step size c_t, the step size sequence c_t has to satisfy the following conditions:

$$\sum_{t=1}^{\infty} c_t = \infty \tag{3}$$

and

$$\sum_{t=1}^{\infty} c_t^2 < \infty \tag{4}$$

Condition Eq. (3) is necessary for the step not to vanish prematurely before reaching the optimum with sufficient precision. Condition Eq. (4) provides for decreasing step size. With a constant step size, the solution would infinitely fluctuate around the optimum. This is because, in the context of error minimization, its random instance $h(w)$ will not diminish for individual samples, although the gradient $g(w) = E[h(x)]$ will gradually vanish as it approaches the minimum. Finally, the gradients of individual samples will not vanish even if their mean over the training set is zero. At the minimum, $g(w) = 0$ will result from a balance between individual nonzero vectors $h(w)$ pointing to various directions.

2 SHORTCOMINGS OF THE BATCH ORIENTED APPROACH

The concept of gradient determination using a subset of the training set is mostly satisfactory. However, several deficiencies from theoretical viewpoints suggest an enhancement potential.

2.1 Violating the Conditions of the Stochastic Approximation

The conditions Eq. (3) and Eq. (4) for convergence of the stochastic approximation procedure to a global (or at least local) minimum result from the stochastic approximation theory. Unfortunately, they are almost always neglected in the neural network training practice. This may lead to a bad convergence (or even divergence). The common *Stochastic Gradient Descent* (SGD) with a fixed learning step size violates the stochastic approximation principles. However, even popular sophisticated algorithms do not satisfy the conditions. The widespread (and successful) *Adam* (Kingma and Ba, 2015) optimizer uses a weight consisting of the quotient of the exponential moving average derivative and the exponential moving average of the square of the derivative

$$\begin{aligned} w_{t+1,i} &= w_{t,i} - \frac{cm_{t,i}}{\sqrt{d_{t,i}}} \frac{\partial E\left(w_{t,i}\right)}{\partial w_{t,i}} \\ m_{t,i} &= \beta_1 d_{t-1,i} + (1-\beta_1) \frac{\partial E\left(w_{t-1,i}\right)}{\partial w_{t-1,i}} \\ d_{t,i} &= \beta_2 d_{t-1,i} + (1-\beta_2) \left(\frac{\partial E\left(w_{t-1,i}\right)}{\partial w_{t-1,i}}\right)^2 \end{aligned} \tag{5}$$

with metaparameters c, β_1, and β_2, network weights $w_{t,i}$, and the loss function E. β_1 is the decay factor of the exponential mean of the error derivative, β_2 is the decay factor of the square of the error derivative, and c is the step length scaling parameter. Their values have been set to the framework's sensible defaults $c = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ in the following computing experiments.

Normalizing the gradient components by the moving average of their square via $\sqrt{d_{t,i}}$ is the opposite of the decreasing step size required in the stochastic approximation theory. If the gradient becomes small (as expected at the proximity of the minimum), the normalization increases them. This may or may not be traded off by the average gradient $m_{t,i}$.

2.2 "Good" Approximation of Gradient Is not Sufficient

The quoted argument of (Goodfellow et al., 2016, Section 8.1.3) that the standard deviations of gradient components are decreasing with the square root $\sqrt{K}$ of number of samples used while the computing expense is increasing with K is, of course, accurate for independent samples drawn from a population.

However, this relationship is only valid if the whole statistical population (from which the training set is also drawn) is considered. This does not account for the nature of numerical optimization algorithms. The more sophisticated among them follow the descent direction. The gradient's statistical deviation is relatively small with respect to the statistical population, but this does not guarantee that a so-determined estimated descent direction is not, in fact, an ascent direction. The difference between descent and ascent may easily be within the gradient estimation error — the batch-based gradient is always a sample estimate, with standard deviation depending on the unknown variance of the individual derivatives within the training set. By contrast, optimizing over the training set itself, the training set gradient is computed deterministically, with zero deviation. Then, the descent direction is certain to lead to a decrease in loss.

The explicit task of the optimization algorithm is to minimize the loss over the training set. If the goal of optimizing over the whole (explicitly unknown) population is adopted, the appropriate means would be biased estimates that can have lower errors over the population, such as ridge regression for linear problems (van Wieringen, 2023). The biased estimate theory provides substantial results concerning this goal but also shows that it is difficult to reach because of unknown regularization parameters, which can only be determined with computationally expensive experiments using validation data sets.

Even if the loss is extended with regularization terms to enhance the model's performance on the whole population (represented by a validation set), the optimized regularized fit is reached at the minimum of the extended loss function once more over *the given training set*. Thus, as mentioned above, it is incorrect from the optimization algorithm's viewpoint to compare the precision of the training set gradient with that of the batches, which are subsamples drawn from the training set. The former is precise, while the latter are approximations.

The related additional argument frequently cited is that what is genuinely sought is the minimum for the population and not for the training set. However, this argument is somewhat misleading. There is no method for finding the true, exact minimum for the population only based on a subsample such as the training set — the training set is the best and only information available. Also, the loss function values used in the algorithm to decide whether to accept or reject a solution are values for the given training set. Examples in (Hrycej et al., 2023) show that no law guarantees computing time savings through incremental learning for the same performance.

2.3 Convexity Around the Minimum Is not Exploited

Another problem is that in a specific environment of the local minimum, every smooth function is convex — this directly results from the minimum definition. Then, the location of the minimum is not determined solely by the gradient; the Hessian matrix also captures the second derivatives. Although using an explicit estimate of the Hessian is infeasible for large problems with millions to billions of parameters, there are second-order algorithms that exploit the curvature information implicitly. One of them is the well-known conjugate gradient algorithm (Hestenes and Stiefel, 1952; Fletcher and Reeves, 1964), thoroughly described in (Press et al., 1992), which requires only the storage of an additional vector with a dimension equal to the length of the plain gradient. However, batch sampling substantially distorts the second-order information more than the gradient (Goodfellow et al., 2016). This leads to a considerable loss of efficiency and convergence guarantee of second-order algorithms, which is why they are scarcely used in the neural network community, possibly sacrificing the benefits of their computing efficiency.

Second-order algorithms cannot be used with the batch scheme for another reason. They are usually designed for continuous descent of loss values. Reaching a specific loss value with one batch cannot guarantee that this value will not become worse with another batch. This violates some assumptions for which the second-order algorithms have been developed. Mediocre computing results with these algorithms in the batch scheme seem to confirm this hypothesis.

3 SUBSTITUTING THE TRAINING SET BY A SUBSET

To summarize the arguments in favor of batch-oriented training, the batch-based procedure is justified by the assumption that the gradients for individual batches are roughly consistent with the gradient over the whole training set (epoch). So, a batch-based improvement is frequently enough (but not always, depending on the choice of the metaparameters) also an improvement for the epoch. This is also consistent with the computing experience record. On the other hand, one implicitly insists on optimizing over the whole training set to find an optimum, as one batch is not expected to represent the training set fully.

Batch-oriented gradient optimization hypothesizes that the batch-loss gradient approximates the training set gradient and the statistic population gradient well enough.

By contrast, the hypothesis followed here is related but essentially different. It is assumed that *the optimum of the loss subset is close to the optimum of the training set.*

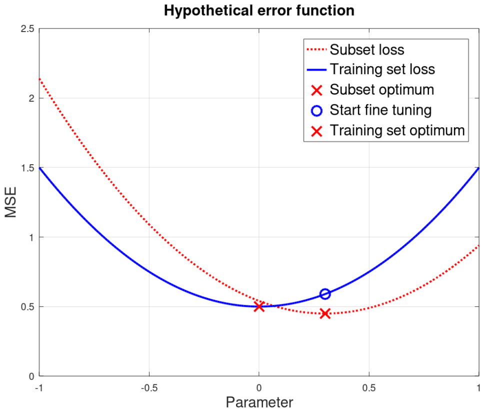


Figure 1: Optima for a subset and the whole training set.

Even if the minimum approximation is imperfect, it can be expected to be a very good initial point for fine-tuning the whole training set so that a few iterations may suffice to reach the true minimum. This principle is illustrated in Fig. 1. The subset loss function (red, dotted) is not identical to the training set loss function (blue, solid). Reaching the minimum of the subset loss function (red cross) delivers an initial point for fine-tuning on the training set (blue circle). This initial point is close to the training set loss minimum (blue cross) and is very probably within a convex region around the minimum. This motivates using fast second-order optimization methods such as the conjugate gradient method (Press et al., 1992).

Another benefit of such a procedure is that for a fixed subset, both the gradient and the intermediary loss values are exact. This further justifies the use of second-order optimization methods.

Of course, the question is how large the subset has to be for the approximation to be sufficiently good. As previously noted, a smooth function is always locally convex around a minimum. If the approximate minimum over the training subset is in this environment, conditions for efficient minimization with the help of second-order algorithms are satisfied. Then, a fast convergence to the minimum over the whole training set can be expected.

Consequently, it would be desirable for the minimum of the subset loss to be within the convex region of the training set loss.

4 SETUP OF COMPUTING EXPERIMENTS

The following computing experiments investigate the support of these hypotheses. The experimental method substitutes the training set with representative subsamples of various sizes. Subsequently, a short fine-tuning on the whole training set has been performed to finalize the optimum solution. The model is trained on the subsamples for exactly 1,000 epochs, while the fine-tuning on the whole training set is limited to 100 epochs.

4.1 Benchmark Datasets Used

The benchmarks for the evaluation have been chosen from the domain of computer vision. They are medium-sized problems that can be run for a sufficient number of experiments. This would not be possible with large models such as those used in language modeling.

For the experiments, three well-known image classification datasets MNIST (LeCun et al., 1998), CIFAR-10, and CIFAR-100 (Krizhevsky, 2009) were used. MNIST contains grayscale images of handwritten digits (0–9) while CIFAR-10 contains color images of exclusively ten different mundane objects like “horse”, “ship”, or “dog”. CIFAR-100 contains the same images; however, they are classified into 100 fine-grained classes. They contain 60,000 (MNIST) and 50,000 (CIFAR-10 and CIFAR-100) training examples. Their respective preconfigured test split of each 10,000 examples are used as validation sets. While both CIFAR-10 and CIFAR-100 are evenly distributed among the classes, MNIST is roughly evenly distributed. We opted to proceed without mitigating the slight class imbalance.

4.2 The Model Architecture

The model is a convolutional network inspired by the *VGG* architecture (Simonyan and Zisserman, 2015). It uses three consecutive convolutional layers with the same kernel size of 3×3, 32/64/64 filters, and the ReLU activation function. Each is followed by a maximum pooling layer of size 2×2. The last feature map is flattened, and after a classification block with one layer of 64 units and the ReLU activation function, a linear dense layer classifies it into a softmax vector. All trainable layers' parameters are initialized randomly from the Glorot uniform distribution (Glorot and Bengio, 2010) with a unique seed per layer such that all trained models throughout the experiments have an identical starting point. The biases of each layer are initialized by zeros. The number of parameters for the models differs only because MNIST has one input channel, while CIFAR-10 and CIFAR-100 have three, and CIFAR-100 has 100 class output units instead of 10.

4.3 Preventing Underdetermination of Model Parameters

An important criterion is that the training set size is sufficient for this procedure. The size of the training subsets (as related to the number of model parameters) must be large enough for the model not to be underdetermined. This should be true for most of the subsets tested so that we can fairly compare subsets that are a relatively small fraction of the training set. As a criterion for this, the overdetermination ratio of each benchmark candidate has been evaluated (Hrycej et al., 2023):

$$Q = \frac{KM}{P} \tag{6}$$

with K being the number of training examples, M being the output vector length (usually equal to the number of classes), and P being the number of trainable model parameters.

This formula justifies itself by ensuring that the numerator KM equals the number of constraints to be satisfied (the reference values for all training examples). This product must be larger than the number of trainable parameters for the system to be sufficiently determined. (Otherwise, there are infinite solutions, most of which do not generalize.) This is equivalent to the requirement for the overdetermination ratio Q to be larger than unity. It is advisable that this is satisfied for the training set subsets considered, although subsequent fine-tuning on the whole training set can "repair" a moderate underdetermination.

The two datasets MNIST and CIFAR-10 have ten classes. This makes the number of constraints KM in Eq. (6) too small for subsets with $b > 4$. This is why these two datasets have been optionally expanded by image augmentation. This procedure implies slight random rotations, shifts, and contrast variations. So, the number of training examples has been increased tenfold by augmenting the training data. With CIFAR-100 containing 100 classes, this problem does not occur, and it was not augmented.

4.4 Processing Steps

The processing steps for every given benchmark task and a tested algorithm have been the following:

- The number of subsets b such that a subset is the fraction $^1/_b$ of the training set has been defined. These numbers have been: $b \in B$ with $B = \{2, 4, 8, 16, 32, 64, 128\}$. With a training set size K, a subset contains $^K/_b$ samples. For example, a value of $b = 2$ results in a subset with half of the samples from the original training set.
- All b subsets of size $^K/_b$ have been built to support the results statistically. Each subset $B_{bi}, i = 1, \ldots, b$, consists of training samples with index i selected so that the subsets partition the entire training set. The number of experiments is excessive for larger values of b, so only five random subsets are selected. All randomness is seeded such that each experiment receives the same subset.
- For every $b \in B$ and every $i = 1, \ldots, b$, the subset loss E_{bi} has been minimized using the selected training algorithm. The number of epochs has been set to 1,000. Additionally, the losses for the whole training set (E_{BTbi}) and validation set (E_{BVbi}) have been evaluated. Subsequently, a fine-tuning on the whole training set for 100 epochs has been performed, and the metrics for the training set (E_{Tbi}) and validation set (E_{Vbi}) have been evaluated. In summary, the set of loss characteristics E_{bi}, E_{BTbi}, E_{BVbi}, E_{Tbi}, and E_{Vbi} have represented the final results.
- For comparison, the typical training on the original training set is given by choosing $b = 1$.

The conjugate gradient algorithm would be the favorite for optimizing the subset (because of its relatively small size) and fine-tuning (because of its expected convexity in the region containing the initial point delivered by the subset training). Unfortunately, this algorithm is unavailable in deep learning frameworks like *Keras*. This is why the popular Adam algorithm has been used. For reproducibility and removing additional hyperparameters, a fixed learning rate of 0.001 was employed for all training steps.

5 RESULTS

5.1 Dataset MNIST

The results for the non-augmented MNIST dataset are depicted in Figs. 2 and 3. The training has two phases:

1. the *subset training* phase, in which only a fraction of the training set is used; and
2. the *fine-tuning* phase, in which the optimized parameters from the subset training phase are fine-tuned by a (relatively short) training over the whole training set.

On the x-axis, fractions of the complete training set are shown as used for the subset training. The axis is logarithmic, so the variants are equally spaced. These are $^1/_2$, $^1/_4$, $^1/_8$, $^1/_{16}$, $^1/_{32}$, $^1/_{64}$, and $^1/_{128}$, as well as the baseline (fraction equal to unity). This baseline corresponds to the conventional training procedure over the full training set.

The plotted magnitudes in Figs. 2 and 3 refer to

- the loss or accuracy reached for the given subset (*Subset pre-training*);
- the loss or accuracy over the whole training set in the subset training optimum (*Tr.set pre-training*);
- the loss or accuracy over the validation set in the subset training optimum (*Valid.set pre-training*);
- the loss or accuracy over the whole training set attained through fine-tuning (*Tr.set fine-tuning*); and
- the loss or accuracy over the validation set in the fine-tuning optimum (*Valid.set fine-tuning*).

All of them are average values over the individual runs with disjoint subsets.

The dotted vertical line marks the subset fraction with overdetermination ratio Eq. (6) equal to unity. To the left of this line, the subsets are underdetermined; to the right, they are overdetermined.

Both loss (Fig. 2) and accuracy (Fig. 3) suggest similar conjectures:

- The subset training with small subsets leads to poor training set and validation set losses. This gap diminishes with the growing subset fraction.
- Fine-tuning largely closes the gap between the training and validation sets. The optimum value for the training set tends to be lower for large fractions (as they have an "advance" from the subset training, but this does not lead to a better validation set performance. The baseline loss (the rightmost point) exhibits the highest validation set loss.

The overdetermination ratio delivers, together with the mentioned vertical lines in Fig. 2 and Fig. 3, an additional finding: The gap between the performance on the subset and on the whole training set after the subset training is very large for $Q < 1$ (the left side of the plot) and shrinks for $Q > 1$ (the right side).

The results for the augmented data are depicted in the plots Fig. 4 and Fig. 5. As the augmented data are more challenging to fit, their performance characteristics are generally worse than those of the non-augmented dataset. However, an important point can be observed: the performance after the pre-training (particularly for the validation set) does not differ to the same extent as it did with non-augmented data. As with the non-augmented dataset, the baseline loss (the rightmost point) exhibits the highest validation set loss. There, the difference between the lowest and the highest subset losses has been tenfold, while it is roughly the same for all subset fractions with the augmented data.

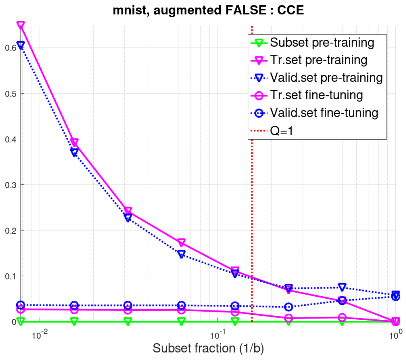


Figure 2: Dataset MNIST (not augmented), loss optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

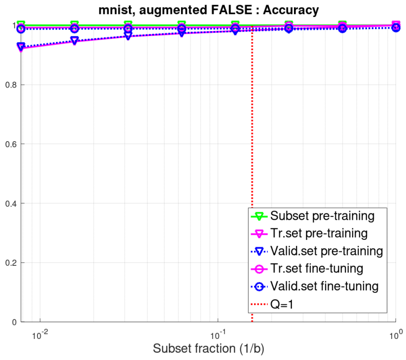


Figure 3: Dataset MNIST (not augmented), accuracy optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

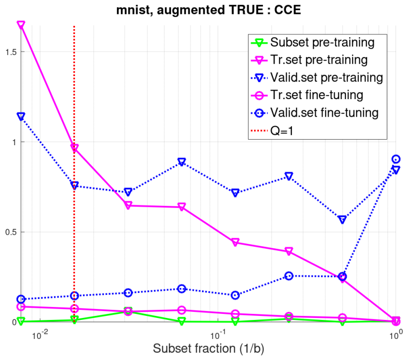


Figure 4: Dataset MNIST (augmented), loss optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

The ten times larger size of the augmented dataset leads to overdetermination ratios Q mostly (except for the fraction $^{1}/_{128}$) over unity. Then, even the small-fraction subsets generalize acceptably (which is the goal of sufficient overdetermination).

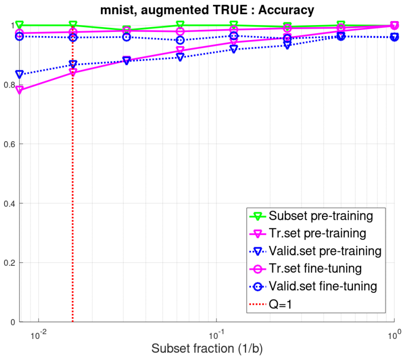


Figure 5: Dataset MNIST (augmented), accuracy optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

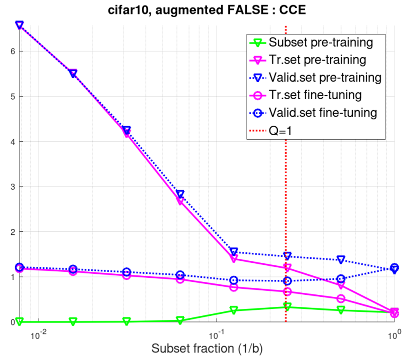


Figure 6: Dataset CIFAR-10 (non-augmented), loss optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

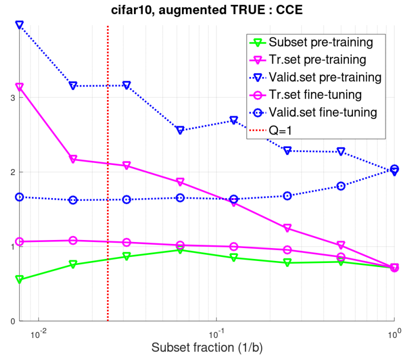


Figure 7: Dataset CIFAR-10 (augmented), loss optima for a pre-trained subset and the whole training set dependent on the subset size (as a fraction of the training set).

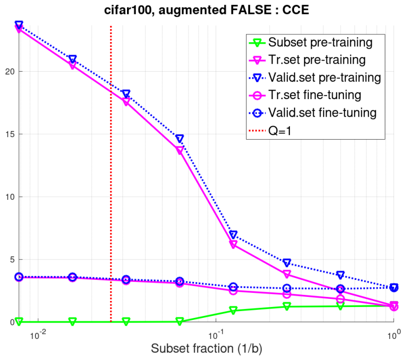


Figure 8: Dataset CIFAR-100 (non-augmented), loss optima for a pre-trained subset and the whole training set in dependence from the subset size (as a fraction of the training set).

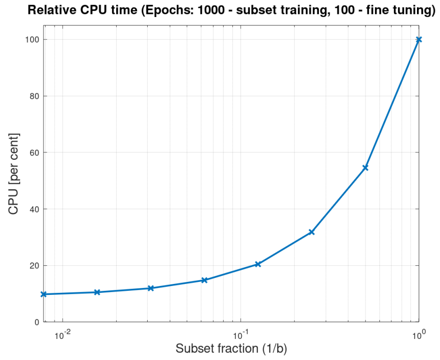


Figure 9: Training time relative to the conventional training in dependence from the subset size (in percent).

5.2 Dataset CIFAR-10

The results for non-augmented CIFAR-10 data are depicted in Fig. 6, those for augmented data in Fig. 7. Due to the size of CIFAR-10 being close to MNIST, the overdetermination ratios are also very similar. Since CIFAR-10 is substantially harder to classify, losses and accuracies are worse.

The accuracy is a secondary characteristic (since the categorical cross-entropy is minimized), and its explanatory power is limited. For this and the space reasons, accuracies will not be presented for CIFAR-10 and CIFAR-100.

Nevertheless, the conclusions are similar to those from MNIST. The gap between the subset's and the entire training set's fine-tuning performance diminishes as the subset grows. This gap is large with non-augmented data in Fig. 6 because of low to overdetermination ratios Q but substantially smaller for augmented data in Fig. 7 where overdetermination ratios are sufficient. The verification set performance after both training phases is typically better with subsets of most sizes than with the whole training set.

5.3 Dataset CIFAR-100

The results for (non-augmented) CIFAR-100 data are depicted in Fig. 8. This classification task differentiates 100 classes so that there are only 500 examples per class. Optimum losses for this benchmark are higher than for the previous ones.

For small subset fractions, the representation of the classes is probably insufficient. This may explain the large gap between the subset loss and the training set loss after the subset training with small subset fractions. These may contain, on average, even as few examples per class as four. Nevertheless, the loss for the validation set with various subset sizes is close to the baseline loss for the conventional full-size training.

6 CONCLUSION

The experiments presented support the concept of subset training. We demonstrated the following elements.

- The subset training leads to results comparable with conventional training over the whole training set.
- The overdetermination ratio Q (preferably above unity) should determine the subset size. Nevertheless, even underdetermined subsets may lead to a good fine-tuning result, although they put more workload on the fine-tuning (to close the large generalization gap).
- To summarize, even small subsets can be representative enough to approximate the training set loss minimum well whenever the overdetermination ratio sufficiently exceeds unity.

The most important achievement is the reduction of computing expenses. Most optimizing iterations are done on the subset, where the computational time per epoch is a fraction of that for the whole training set. In our experiments with ten times more subset training epochs than fine-tuning epochs, the relative computing time in percent of the baseline is shown in Fig. 9. Computational resource savings of 90 % and more are possible.

This empirical evaluation using five benchmarks from the CV domain is insufficient for making general conclusions. Large datasets such as ImageNet are to be tested in the future. They have been omitted because many experiments are necessary to produce sufficient statistics. Furthermore, these experiments can be extended to language benchmarks and language models.

It is also important to investigate the behavior of the second-order optimization algorithms such as conjugate gradient (Hestenes and Stiefel, 1952; Fletcher and Reeves, 1964). Their strength can develop only with a sufficient number of iterations. This is an obstacle if very large training sets are a part of the task. Appropriately chosen subsets can make such training feasible and help to reach good performance even with models of moderate size.

REFERENCES

Fletcher, R. and Reeves, C. M. (1964). Function minimization by conjugate gradients. *The Computer Journal*, 7(2):149–154.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9, pages 249–256, Sardinia, Italy. PMLR.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts.

Hestenes, M. R. and Stiefel, E. (1952). *Methods of conjugate gradients for solving linear systems*, volume 49. NBS Washington, DC.

Hrycej, T., Bermeitinger, B., Cetto, M., and Handschuh, S. (2023). *Mathematical Foundations of Data Science*. Texts in Computer Science. Springer International Publishing, Cham.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations*.

Krizhevsky, A. (2009). Learning Multiple Layers of Features from Tiny Images. Dataset, University of Toronto.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in C (2nd ed.): the art of scientific computing*. Cambridge University Press, USA.

Robbins, H. and Monro, S. (1951). A Stochastic Approximation Method. *Annals of Mathematical Statistics*, 22(3):400–407. 3515 citations (Crossref) [2021-08-17].

Simonyan, K. and Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ICLR*.

van Wieringen, W. N. (2023). Lecture notes on ridge regression. arXiv:1509.09169 [stat].

Multi-Label Classification for Fashion Data: Zero-Shot Classifiers via Few-Shot Learning on Large Language Models

Dongming Jiang, Abhishek Shah, Stanley Yeung, Jessica Zhu, Karan Singh and George Goldenberg
CaaStle Inc, U.S.A.
{*dj, abhishek.shah, stanley.yeung, jessica.zhu, karan, golden*}*@caastle.com*

Keywords: Large Language Model, Few-Shot Learning, Zero-Shot Learning, Inference, Knowledge Generation, Multi-Label Classification, Scalability, Fashion Dynamics.

Abstract: Multi-Label classification is essential in the fashion industry due to the complexity of fashion items, which often have multiple attributes such as style, material, and occasion. Traditional machine-learning approaches face challenges like data imbalance, high dimensionality, and the constant emergence of new styles and labels. To address these issues, we propose a novel approach that leverages Large Language Models (LLMs) by integrating few-shot and zero-shot learning. Our methodology utilizes LLMs to perform few-shot learning on a small, labeled dataset, generating precise descriptions of new fashion classes. These descriptions guide the zero-shot learning process, allowing for the classification of new items and categories with minimal labeled data. We demonstrate this approach using OpenAI's GPT-4, a state-of-the-art LLM. Experiments on a dataset from CaaStle Inc., containing 2,480 unique styles with multiple labels, show significant improvements in classification performance. Few-shot learning enhances the quality of zero-shot classifiers, leading to superior results. GPT-4's multi-modal capabilities further improve the system's effectiveness. Our approach provides a scalable, flexible, and accurate solution for fashion classification, adapting to dynamic trends with minimal data requirements, thereby improving operational efficiency and customer experience. Additionally, this method is highly generalizable and can be applied beyond the fashion industry.

1 INTRODUCTION

Multi-label classification plays a crucial role in fashion applications due to the complex nature of fashion items, which often possess multiple attributes such as style, material, occasion, and season. For example, a single dress might be labeled as "casual," "floral," "cotton," and "summer." Accurate classification is fundamental for various functions, including merchandising, inventory management, trend analysis, and personalized customer experiences. An efficient multi-label classification system can significantly enhance operational efficiency, customer satisfaction, and sales by aligning products with consumer preferences.

This paper presents our work in addressing multi-label classification for CaaStle Inc., a company that provides advanced technology and services to apparel brands, focusing on optimizing business operations and consumer engagement. CaaStle manages a vast inventory where garments often carry multiple labels, with some labels being far less frequent than others. The imbalanced, high-dimensional, and sparse nature of this data creates challenges for traditional machine learning approaches. Moreover, with new styles and items constantly entering the inventory, the need for continuous re-labeling and model retraining becomes costly and time-consuming.

To address these challenges, we propose a novel approach that utilizes the reasoning capabilities of Large Language Models (LLMs) to enhance multi-label classification. By integrating few-shot and zero-shot learning, our system can effectively classify new and existing fashion items with minimal labeled data. We demonstrate this approach using OpenAI's GPT-4 on a real-world dataset from CaaStle, showcasing improved classification performance and scalability. This solution adapts to fashion trends with minimal data requirements and offers potential applications beyond the fashion industry.

To our knowledge, no prior work has combined the three elements of LLMs, few-shot learning, and zero-shot learning for multi-label classification in the fashion industry. This novel integration marks a significant advancement in the field. Specifically, we are the first to leverage LLMs to generate detailed and

Jiang, D., Shah, A., Yeung, S., Zhu, J., Singh, K. and Goldenberg, G.
Multi-Label Classification for Fashion Data: Zero-Shot Classifiers via Few-Shot Learning on Large Language Models.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 250-257
ISBN: 978-989-758-716-0; ISSN: 2184-3228

precise descriptions of new fashion categories using few-shot learning. These descriptions serve as guidelines for zero-shot learning, enabling accurate classification of emerging categories.

2 RELATED WORK

The fashion industry has seen significant growth and evolution in classification techniques over the past few decades (Abbas et al., 2024; Saranya and Geetha, 2022; Abd Alaziz et al., 2023; Xhaferra et al., 2022; Guo et al., 2019a; Kolisnik et al., 2021; Q. Ferreira et al., 2019; Inoue et al., 2017; Ferreira et al., 2021). Traditional classification techniques in the fashion industry primarily relied on manual categorization, for example, based on silhouette and shapes that characterize a garment's outlines and fit, garment types and purposes such as top, dress, and pants, and design elements as well as detailed attributes of a garment style such as hemline length and neckline shape. Moving into the 21st century, the fashion industry began to adopt more sophisticated hierarchical taxonomies and categorization systems to organize garments into multiple levels using various semantic grouping and logic. Recent research has focused on hierarchical multi-label classification models (Seo and Shin, 2019; Zhong et al., 2023; Mallavarapu et al., 2021; Al-Rawi and Beel, 2020) that mimic human classification processes, and predict and produce multiple labels at different taxonomy levels for each garment. With the advent of computer vision and deep learning, more advanced and automated classification approaches like Convolutional Neural Networks (CNNs) (LeCun et al., 1998; Krizhevsky et al., 2017; Szegedy et al., 2015; He et al., 2016) have emerged, enabling image-based classification of garments, styles, and attributes directly from visual data. More recently, inspired by the rapid advancement and widespread adoption of Artificial Intelligence (AI) foundation models, application of the multi-modal techniques (Guo et al., 2019b; Ngiam et al., 2011; Lu et al., 2019) with the ability to understand and generate data across multiple modalities, for example, text and image, has become active research in fashion classification.

However, due to the complexity of algorithms that require vast amounts of training data and substantial computational power, current techniques face significant challenges in addressing the rapidly evolving dynamics of the fashion industry, particularly in classification problems. In this paper, we introduce a novel approach to multi-label classification, integrating LLMs (Chen et al., 2020), few-shot learning (Kadam and Vaidya, 2020), and zero-shot learning (Raffel et al., 2020) to develop a scalable, accurate, and flexible system tailored to the dynamic, trend-sensitive nature of fashion.

3 APPROACH

We describe our algorithm and demonstrate an implementation in more detail in this section.

3.1 Algorithm

3.1.1 Step 1: Leveraging LLM for Few-Shot Learning

1. Initial training with few-shot learning
    - Utilize a small, labeled dataset to train the LLM on specific fashion categories.
    - The LLM learns from this limited data to understand and identify key attributes and features associated with each category.
2. Inference and reasoning
    - The LLM applies its inference and reasoning capabilities to generalize from the few examples provided.
    - It identifies patterns, trends, and unique characteristics of the fashion items within the limited data, improving its understanding of the categories.

3.1.2 Step 2: Generating Descriptions for New Classes

1. Guiding LLM to generate descriptions
    - When a new fashion category and class is introduced, the LLM uses its learned knowledge and the few-shot learning context to generate a detailed and precise description of the new class.
    - This description includes key attributes, styles, materials, and other relevant features that define the new category.
2. Semantic enrichment
    - The generated description can be enriched with semantic information, leveraging embeddings and attributes that the LLM has learned from existing data.

3.1.3 Step 3: Zero-Shot Learning with Generated Descriptions

1. Utilizing descriptions for zero-shot learning

- The detailed class description generated by the LLM serves as a guideline for the zero-shot learning process.
- The system uses the description to map features of unseen instances to the new class, leveraging semantic similarities and relationships.

2. Building binary classifiers
    - For each new class, the system constructs binary classifiers using the LLM. These classifiers determine whether an instance belongs to the new class based on the description and semantic guidance.
    - The binary classifiers are integrated into the overall multi-label classification framework, enabling the system to handle multiple labels simultaneously.

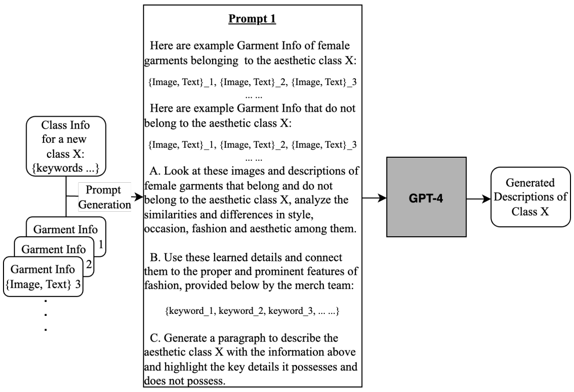
3.1.4 Step 4: Multi-Label Classification

1. Integrating classifiers
    - The binary classifiers for new classes are combined with existing classifiers to create a comprehensive multi-label classification system.
    - The system evaluates each fashion item against all relevant classifiers to assign the appropriate labels.
2. Inference and prediction
    - During inference, the system processes new fashion items, applying both the few-shot learned models and the zero-shot classifiers guided by the LLM-generated descriptions.
    - The LLM's reasoning capabilities ensure accurate and context-aware predictions, even for classes with minimal or no labeled examples.

3.2 Implementation

There exist various options for LLMs in an implementation of our proposed approach. In this paper, we present experiments and results from one of our implementations using OpenAI GPT-4 (Achiam et al., 2023). GPT-4 is a state-of-the-art LLM that is pretrained. In addition to its proficiency in language understanding and generation, it excels in understanding context, following guidelines and instructions, logical inference, and basic reasoning.

Figure 1 shows our implementation of the approach for Step 1. Garment Info contains examples of the garments that belong to and that do not belong to the new class, in the form of the image and text descriptions of the garments. Class Info contains classification guidelines for the class, which can be in various forms that can be as simple as keywords that best describe the fashion class. Class Info and Garment Info are the inputs to GPT-4 for the few-shot learning. They can either be provided by humans or be generated by LLMs. We will compare and discuss these two different methods in more detail in the Experiment section. These inputs are structured into Prompt 1 which is sent into GPT-4 through the GPT API. The goal of Prompt 1 is to guide GPT-4 to do the few-shot learning using the labeled data and produce the class descriptions accordingly. This learning process can iterate with various examples and guidelines in multiple rounds, each of which results in a class description.



Prompt 1

Here are example Garment Info of female garments belonging to the aesthetic class X:

{Image, Text}_1, {Image, Text}_2, {Image, Text}_3
... ...

Here are example Garment Info that do not belong to the aesthetic class X:

{Image, Text}_1, {Image, Text}_2, {Image, Text}_3
... ...

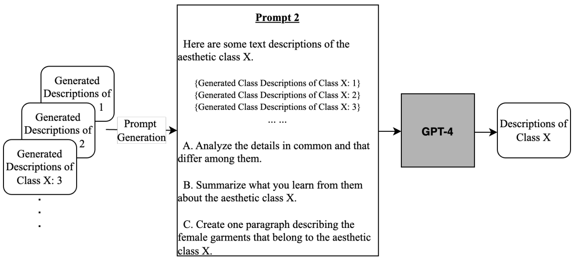
A. Look at these images and descriptions of female garments that belong and do not belong to the aesthetic class X, analyze the similarities and differences in style, occasion, fashion and aesthetic among them.

B. Use these learned details and connect them to the proper and prominent features of fashion, provided below by the merch team:

{keyword_1, keyword_2, keyword_3, ...}

C. Generate a paragraph to describe the aesthetic class X with the information above and highlight the key details it possesses and does not possess.

Figure 1: Implementation of the Algorithm Step 1 for running a few-shot learning with GPT-4.



Prompt 2

Here are some text descriptions of the aesthetic class X.

{Generated Class Descriptions of Class X: 1}
{Generated Class Descriptions of Class X: 2}
{Generated Class Descriptions of Class X: 3}
... ...

A. Analyze the details in common and that differ among them.

B. Summarize what you learn from them about the aesthetic class X.

C. Create one paragraph describing the female garments that belong to the aesthetic class X.

Figure 2: Implementation of the Algorithm Step 2 for generating the final descriptions of a new fashion class through GPT-4.

Figure 2 illustrates how the final descriptions of a new fashion class are generated. With potentially multiple class descriptions generated by the few-shot learning process, Prompt 2 carries these results to GPT-4. The goal of Prompt 2 is to teach GPT-4 with the knowledge that is learned from the small number of labeled data in Step 1, and instruct GPT-4 to analyze and refine them using its inference and reasoning capabilities, producing precise final class descriptions at the end.

Figure 3 demonstrates a zero-shot binary classifier. It uses Prompt 3 to instruct GPT-4 to do proper

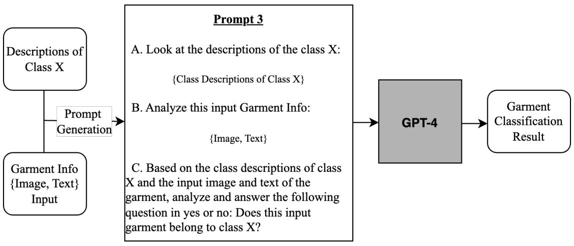

Descriptions of Class X
Prompt Generation
Garment Info {Image, Text} Input
Prompt 3
A. Look at the descriptions of the class X:
{Class Descriptions of Class X}
B. Analyze this input Garment Info:
{Image, Text}
C. Based on the class descriptions of class X and the input image and text of the garment, analyze and answer the following question in yes or no: Does this input garment belong to class X?
GPT-4
Garment Classification Result

Figure 3: Implementation of the Algorithm Step 3 that builds a zero-shot binary classifier using the generated class descriptions on GPT-4.

inference and answer a binary classification question, taking into account the knowledge learned for the new fashion class and the query garment.

4 EXPERIMENT

4.1 Dataset

To support the development of the classification models and system, CaaStle picked a small proportion of its inventory pool and manually tagged and validated all of their labels. This dataset includes 2480 different styles and each style can have 1 or more of 18 different labels (Table 1). Each style comes with a vendor description and key characteristics edited by CaaStle's merchandising team. Data formats of each style include a primary image, multiple images of the same style in various views such as front view, side view, and back view, and descriptions in text. Examples of the data can be browsed at https://closet.gwynniebee.com/ and https://www.haverdash.com/. In the rest of the paper, when we refer to an image of a style, it is always the primary image. When multiple views of a style are used in certain approaches, we will explicitly call them out as multi-view images. We will refer to the edited vendor description of each style Human product description in this paper. The merchandising team also provides a natural-language description of each class / label and classification guidelines, and uses them to train the team for the manual tagging and validation of the class labels. We will call this data Human classification guidelines in this paper. Each style can be tagged with multiple classes or labels in Aesthetic Styles as well as in Occasions, and only a single class or label in Weather. In the rest of the paper, we use the terms class and label interchangeably.

Table 1: Category and Class labels in the dataset.

Aesthetic Styles	Occasion	Weather
Feminine	Party	Cold
Classic	Casual/Lounge	Warm
Edgy	Resort	Year-round
Boho	Day Night	
Retro	Work	
Athleisuren	Everyday	
Minimalist	Wedding Guest	
Preppy		

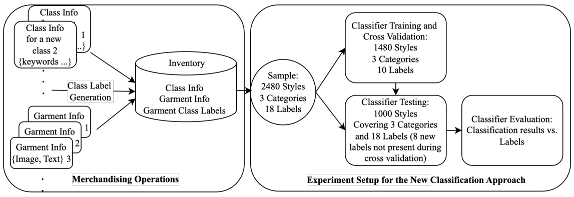

Class Info
Class Info for a new class 2 {keywords ...}
1
..}
Class Label Generation
Garment Info
Garment Info
Garment Info {Image, Text} 3
1
2
Inventory
Class Info
Garment Info
Garment Class Labels
Merchandising Operations
Sample: 2480 Styles 3 Categories 18 Labels
Classifier Training and Cross Validation: 1480 Styles 3 Categories 10 Labels
Classifier Testing: 1000 Styles Covering 3 Categories and 18 Labels (8 new labels not present during cross validation)
Classifier Evaluation: Classification results vs. Labels
Experiment Setup for the New Classification Approach

Figure 4: Workflow and setup of the experiment.

4.2 Experiment Setup

The data selection process is guided by general fashion classification criteria and the high-level distribution of style attributes, such as product types (e.g., tops, dresses, pants), fabric, labor costs, and the constraints of manual tagging and validation. The merchandising team continuously provides subsets of the dataset through the data pipeline. This approach aligns with our model exploration, testing, and system development processes. The workflow and experimental setup are illustrated in Figure 4. We use 60% of the dataset, which arrived earlier in the pipeline, for experimentation, model training, and validation. The remaining 40%, including new labels absent during the training phase, is used to test the classification approach. This setup simulates a real-world scenario where not only new styles of existing labels emerge, but entirely new classes and labels also appear over time. The classification system adapts by learning and building new classifiers for these emerging classes and labels, using a few example labels generated by the merchandising team throughout the process.

4.3 Metrics

To evaluate the classification performance, we consider three relevant metrics.

4.3.1 Accuracy

Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + True Negatives + False Negatives)

Accuracy gives a straightforward measure of overall performance. However, it can be misleading in the case of imbalanced datasets where the majority class dominates the metric.

4.3.2 F1-Score

F1-score is the harmonic mean of precision and recall. F1-score helps alleviate the bias of Accuracy towards dominant classes in imbalanced data. It is more informative than Accuracy especially when the dataset has uneven class distribution by balancing both precision and recall.

- *F1-score = 2 * (Precision * Recall) / (Precision + Recall)*
- *Precision = (True Positives) / (True Positives + False Positives)*
- *Recall = (True Positives) / (True Positives + False Negatives)*

4.3.3 Weighted F1-Score

It is insufficient to compute only the F1-score for each class independently because CaaStle judges the quality of the multi-label classification at the category level across all its classes in addition to the quality of each class. When evaluating quality, the business regards every instance of a single labeling equally, and every label equally. Therefore, we compute a weighted F1-score using a weight that reflects the proportion of the true instances from each class over the total instances of the category.

$$Weighted\ F1 = \sum_{i=1}^{N} w_i F1_i \tag{1}$$

This method takes class imbalance into account, where N is the number of classes in the category, w_i is the ratio of the number of true instances for each class to the total instances for the category, and $F1_i$ is the F1-score for each class.

We present the results in F1-scores for each class and Weighted F1-scores for each category and dataset in this paper.

CaaStle's quality target of the classification system is to achieve at least 0.7 of F1-score for each class, and 0.8 of weighted F1-score for the category that includes the classes.

4.4 Experiments on State-of-the-Art Models

With the labeled styles and their image and text data, we attempt to train a multi-label classification model, using the 2480 unique styles, text description for each of them, 10K multi-view images for all the styles, and 18 possible labels, through the typical training, validation, and testing process. This is an important task in our experiments because we need to understand whether the state-of-the-art modeling methods can support the multi-label classification requirements, and if they do not, what problems we need to address in designing the new methods. The modeling methods we test include Google Vertex by training a classifier from scratch, and three widely adopted pre-trained image classification models, ResNet-50 (Koonce and Koonce, 2021), Vision Transformer (ViT) (Dosovitskiy et al., 2020), and Contrastive Language-Image Pre-training (CLIP) (Radford et al., 2021). We use only image data for Vertex, RestNet-50, and ViT, but {image, text caption} data for CLIP to take advantage of CLIP's multi-modal capability. Experiments show common evidence of serious overfitting across all these different methods. The class-level F1-scores spread from 0.1 to 0.8, and the category-level weighted F1-scores are usually around 0.5 and below. The main challenge comes from the lack of labeled data for the multi-label classification problem. For example, during fine-tuning of the pre-trained models, we need to tune the last layer by having the number of nodes match the number of labels, using the sigmoid rather than the softmax activation function for each node, and fitting with the binary cross-entropy loss function. This more complex mathematical form of the models requires much more labeled data for training. To validate the hypothesis about the impacts of the problem complexity, we also test by reducing the complexity of the problem from multi-label to multi-class and eventually to one-vs-all classification problems. Notice that by reducing the problem complexity we also change the goal of the classification problem itself. We only do so to get a better understanding of the possible causes of the overfitting problem. Transforming the multi-label problem to a multi-class and one-vs-all classification setup indeed helps in improving the testing F1-scores, however, the overfitting is still present, and the F1-scores are still nowhere close to CaaStle's quality target. To continue in this technical direction, even for fine-tuning a pre-trained model, we will need to label a lot more styles especially styles that have multiple labels to start with. In contrast, we will show the results of our proposed approach which significantly outperforms.

4.5 Evaluating CaaStle Approaches

In this section, we summarize the key experiments and results that show the superior performance of the

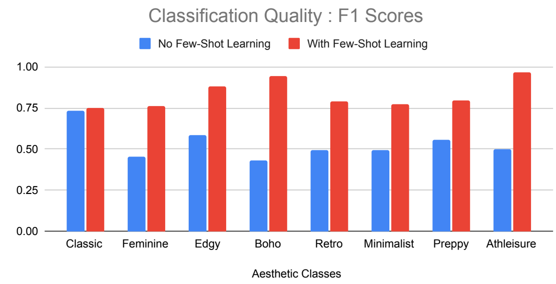


Figure 5: We compare the quality of zero-shot classification between the approaches with and without few-shot learning.

classification that is driven by integrating few-shot learning, LLM, and zero-shot learning.

4.5.1 Few-Shot Learning on LLMs Boosts Zero-Shot Classification

The crucial difference between the two zero-shot classification approaches, shown in Figure 5, is in class description generation. In the approach with no few-shot learning, we use the human classification guidelines that are crafted by the merchandising team. This approach is considered the best effort in zero-shot learning because it leverages the knowledge best known by humans. On the other hand, in the approach with few-shot learning, the classification guideline uses the class description generated by few-shot learning on GPT-4 (Figure 1, Figure 2). We are essentially comparing zero-shot binary classifiers using knowledge learned by few-shot on GPT-4 with that using human knowledge and the best efforts. The improvement in classification quality by the few-shot learning on GPT-4 is significant. Figure 5 shows that the few-shot learning always outperforms, from 2% to 118% better than the zero-shot approach without it. Even though the Aesthetic Classes are very diverse, our proposed approach of few-shot learning is quite robust, showing consistently high performance across all the classes. Compared to the other Aesthetic Classes, styles in the Classic class appear more consistent, as their characteristics are well-captured by human knowledge and descriptions. As a result, learning from additional examples does not provide significant value.

During the experimentation and related sensitivity analyses, we gain more insights into how few-shot learning and GPT-4 interplay. LLMs, including GPT-4, work well in discovering and generalizing common patterns from examples. Fashion items, however, often require attention to some subtle and seemingly minor details that can be decisive in fashion classification but not so much in machine learning. Therefore the prompt needs to be designed and experimented with to better guide GPT-4 to perform learning more specifically. The learning outcome from GPT-4 can be sensitive to the input examples. We test with various strategies, including using positive examples, negative examples, and sampled examples according to certain distribution considerations. We find that it is beneficial for running few-shot learning in multiple epochs, which allows us to run representative but diverse examples throughout the entire learning. Thereafter, we can apply different strategies and algorithms in generating the final class descriptions based on multiple candidates of the class descriptions, coming out of the few-shot learning epochs. Figure 1 and Figure 2 illustrate the prompts we design in both steps for guiding GPT-4 to perform the learning and class description generation tasks.

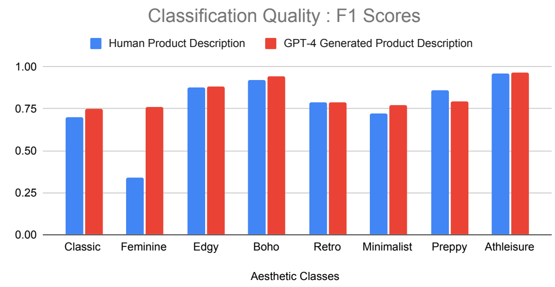


Figure 6: We compare the quality of two different methods in producing the product description of a fashion item. The product description is an important parameter for Garment Info that is required by Prompt 1 in Figure 1.

4.5.2 LLM Generated Garment Data Improves Classification

In the last section, we have already shown that the class description generated by LLM (GPT-4) through few-shot learning significantly improves the classification performance. In this section, we show that the classification performance is further improved by leveraging the product description that is generated by LLM (GPT-4). Figure 6 illustrates that, compared to the approach of using the product descriptions that are provided by the vendors or crafted by humans, the approach of using the GPT-4 generated texts is consistently better. The Preppy class is an exception, as the GPT-4-generated product descriptions are sometimes overly specific about certain details, which can negatively affect the class description generation. At the category level for Aesthetic Styles across all classes, using the Human Product Description yields a weighted F1-score of 0.66, while the GPT-4-Generated Product Description achieves a weighted F1-score of 0.80. This represents a 20% improvement in classification performance when using GPT-4-generated descriptions. It highlights a significant advantage of LLMs like GPT-4, which are trained on vast amounts of internet data, enabling them to reason with richer and broader contexts than the domain-specific expertise of humans.

4.5.3 Multi-Modality Improves Few-Shot Learning Performance

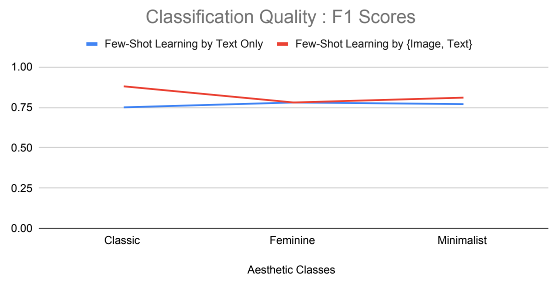


Figure 7: We show the benefit of multi-modality in few-shot learning. Here since we have a smaller number of data samples, we use a line chart that helps show the performance differences between the two lines more clearly.

We can leverage both image and text data in few-shot learning because GPT-4 supports multi-modals. Figure 7 demonstrates the benefits of multi-modality in few-shot learning. Leveraging both image and text data with GPT-4 improves classification performance by 5% to 17%, demonstrating the advantage of GPT-4's multi-modal capabilities.

To conclude, Figure 8 summarizes the classification performance of our proposed approach for all the 18 classes in the testing dataset (Table 1, Figure 4). The weighted F1-score for the entire dataset across all the 18 classes from 3 different categories is 0.802, reaching higher than CaaStle's quality target for the multi-label classification task for every single class and category in CaaStle's dataset. This demonstrates that our new approach is robust.

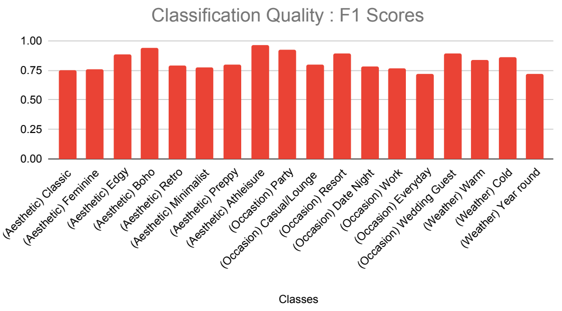


Figure 8: We show the classification performance for all 18 classes in our dataset.

5 CONCLUSIONS

In this paper, we introduced a novel approach that integrates the strengths of LLMs, few-shot learning, and zero-shot learning to create a robust multi-label classification system tailored for the fashion industry. By generating detailed descriptions of new classes and using them as guidelines, our system ensures accurate and scalable classification, adapting seamlessly to the dynamic nature of fashion trends with minimal data requirements. This innovative methodology significantly enhances the efficiency and effectiveness of multi-label classification for fashion items.

Our approach is the first to combine these advanced techniques to address the unique challenges of fashion classification. Through the integration of OpenAI's GPT-4, a state-of-the-art pre-trained LLM, we demonstrated substantial improvements in classification performance, particularly in scenarios with limited labeled data. The few-shot learning process, supported by GPT-4, generates precise class descriptions, which are crucial for effective zero-shot learning. This enables the system to classify new and existing fashion items accurately, maintaining high performance despite the constant influx of new styles and labels.

Additionally, GPT-4's multi-modal capabilities, which allow it to process both image and text data, contribute to the superior performance of our classification system. By leveraging these features, we observed significant improvements in weighted F1-scores across various fashion categories.

This multi-label classification system has already made significant contributions to CaaStle's merchandising and operations. The rapid development of automated, high-quality classification has provided CaaStle with rich semantic data about its inventory, enhancing product capabilities in inventory management, optimization, and personalization. Our approach offers a scalable, flexible, and highly accurate solution, paving the way for further advancements in the fashion industry and beyond.

ACKNOWLEDGEMENTS

We would like to extend our sincere gratitude to the entire Merchandising Team at CaaStle, with special thanks to Francesca De la Rama, Stephanie Shum, Daphne Shapir, Palley Jackson, and Gianni Fuller, for their dedicated efforts in data generation, cleanup, and preparation for the inventory classification project. Their tireless guidance and support have been instrumental in making this work possible.

REFERENCES

Abbas, W., Zhang, Z., Asim, M., Chen, J., and Ahmad, S. (2024). Ai-driven precision clothing classification: Revolutionizing online fashion retailing with hybrid two-objective learning. *Information*, 15(4):196.

Abd Alaziz, H. M., Elmannai, H., Saleh, H., Hadjouni, M., Anter, A. M., Koura, A., and Kayed, M. (2023). Enhancing fashion classification with vision transformer (vit) and developing recommendation fashion systems using dinova2. *Electronics*, 12(20):4263.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Al-Rawi, M. and Beel, J. (2020). Towards an interoperable data protocol aimed at linking the fashion industry with ai companies. *arXiv preprint arXiv:2009.03005*.

Chen, Y.-C., Li, L., Yu, L., El Kholy, A., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. (2020). Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Ferreira, B. Q., Costeira, J. P., and Gomes, J. P. (2021). Explainable noisy label flipping for multi-label fashion image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3916–3920.

Guo, S., Huang, W., Zhang, X., Srikhanta, P., Cui, Y., Li, Y., Adam, H., Scott, M. R., and Belongie, S. (2019a). The imaterialist fashion attribute dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0.

Guo, W., Wang, J., and Wang, S. (2019b). Deep multimodal representation learning: A survey. *Ieee Access*, 7:63373–63394.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Inoue, N., Simo-Serra, E., Yamasaki, T., and Ishikawa, H. (2017). Multi-label fashion image classification with minimal human supervision. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 2261–2267.

Kadam, S. and Vaidya, V. (2020). Review and analysis of zero, one and few shot learning approaches. In *Intelligent Systems Design and Applications: 18th International Conference on Intelligent Systems Design and Applications (ISDA 2018) held in Vellore, India, December 6-8, 2018, Volume 1*, pages 100–112. Springer.

Kolisnik, B., Hogan, I., and Zulkernine, F. (2021). Condition-cnn: A hierarchical multi-label fashion image classification model. *Expert Systems with Applications*, 182:115195.

Koonce, B. and Koonce, B. (2021). Resnet 50. *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, pages 63–72.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Mallavarapu, T., Cranfill, L., Kim, E. H., Parizi, R. M., Morris, J., and Son, J. (2021). A federated approach for fine-grained classification of fashion apparel. *Machine Learning with Applications*, 6:100118.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. (2011). Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696.

Q. Ferreira, B. Costeira, J. R. G., Gui, L.-Y., and Gomes, J. P. (2019). Pose guided attention for multi-label fashion image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Saranya, M. and Geetha, P. (2022). Fashion image classification using deep convolution neural network. In *International Conference on Computer, Communication, and Signal Processing*, pages 116–127. Springer.

Seo, Y. and Shin, K.-s. (2019). Hierarchical convolutional neural networks for fashion image classification. *Expert systems with applications*, 116:328–339.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Xhaferra, E., Cina, E., and Toti, L. (2022). Classification of standard fashion mnist dataset using deep learning based cnn algorithms. In *2022 International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 494–498. IEEE.

Zhong, S., Ribul, M., Cho, Y., and Obrist, M. (2023). Textilenet: A material taxonomy-based fashion textile dataset. *arXiv preprint arXiv:2301.06160*.

Optimizing High-Dimensional Text Embeddings in Emotion Identification: A Sliding Window Approach

Hande Aka Uymaz[a] and Senem Kumova Metin[b]
İzmir University of Economics, Department of Software Engineering, İzmir, Turkey
{*hande.aka, senem.kumova*}@*ieu.edu.tr*

Keywords: Natural Language Processing, Emotion, Large Language Models, Vector Space Models.

Abstract: Natural language processing (NLP) is an interdisciplinary field that enables machines to understand and generate human language. One of the crucial steps in several NLP tasks, such as emotion and sentiment analysis, text similarity, summarization, and classification, is transforming textual data sources into numerical form, a process called vectorization. This process can be grouped into traditional, semantic, and contextual vectorization methods. Despite their advantages, these high-dimensional vectors pose memory and computational challenges. To address these issues, we employed a sliding window technique to partition high-dimensional vectors, aiming not only to enhance computational efficiency but also to detect emotional information within specific vector dimensions. Our experiments utilized emotion lexicon words and emotionally labeled sentences in both English and Turkish. By systematically analyzing the vectors, we identified consistent patterns with emotional clues. Our findings suggest that focusing on specific sub-vectors rather than entire high-dimensional BERT vectors can capture emotional information effectively, without performance loss. With this approach, we examined an increase in pairwise cosine similarity scores within emotion categories when using only sub-vectors. The results highlight the potential of the use of sub-vector techniques, offering insights into the nuanced integration of emotions in language and the applicability of these methods across different languages.

1 INTRODUCTION

Natural language processing (NLP) is a field at the intersection of computer science, artificial intelligence, and linguistics that aims to enable machines to understand and generate human language. In text-based natural language processing, the first step is to convert the given textual content into a numerical format that computers can process. These numerical representations are expected to reflect the complex elements of language, including grammatical rules, vocabulary, and various linguistic components. In the field, the process of converting textual data into numerical representations is commonly referred to as vectorization. The combined representation of documents within a common vector space is known as the vector space model (Manning et al., 2008). This model, which is grounded in linear algebra, allows for vector-based operations like addition, subtraction, and similarity calculations.

We can examine vectorization methods in three groups: traditional (i.e., one-hot encoding, TF, IDF), semantic (i.e., Word2Vec (Mikolov et al., 2013) and GloVe (Global Vectors for Word Representation) (Pennington et al., 2014)), and contextual (i.e., BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018), GPT (Generative pretrained transformers) (OpenAI, 2023), ELECTRA (Clark et al., 2020)) methods. Traditional methods represent words as discrete, sparse vectors without capturing semantic meaning. Semantic methods generate dense vectors that are designed to capture semantics but fail to account for word polysemy. Contextual methods create vectors that vary with context, capturing deeper semantics and polysemy information. Considering the problems with traditional methods, such as the increased computational demand as the number of existing words increases and the lack of semantic information, or in semantic vectors, the neglect of polysemy information and having a single vector for each word independent of its context in a sentence, recently, contextual vectors are more frequently used in NLP problems and achieve better success.

[a] https://orcid.org/0000-0002-3535-3696
[b] https://orcid.org/0000-0002-9606-3625

Aka Uymaz, H. and Kumova Metin, S.
Optimizing High-Dimensional Text Embeddings in Emotion Identification: A Sliding Window Approach.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 258-266
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Unlike static word embeddings, models such as ELMO (Peters et al., 2018), BERT (Devlin et al., 2018), and DistilBERT (Sanh et al., 2019) produce embeddings that consider the word sense and polysemy by adapting to the specific context in which a word is used. ELMO employs a bi-directional long short-term memory architecture to create multiple vectors for words in different contexts, enhancing tasks such as question answering and sentiment detection. BERT, introduced by Google, utilizes a multi-layer bidirectional transformer encoder and a masked language model approach, showing performance in various NLP applications through transfer learning. BERT's significant potential and performance have led to the development of efficient variants such as RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019), and DistilBERT (Sanh et al., 2019). Beyond BERT-based models, approaches like ULMFit and XLNet have also shown promising results in tasks like sentiment and emotion analysis, further diversifying the landscape of contextual embeddings in NLP.

The vectors created to represent any text unit are high-dimensional vectors (e.g., the vectors produced from BERT-base and BERT-large models have dimensions of 768 and 1024, respectively). When performing classification, measuring similarity, and/or running other procedures employing these high-dimensional vectors, they can lead to significant memory and computational costs, especially when working with large datasets. Furthermore, feature engineering holds great importance in classification problems. Although high-dimensional vectors carry detailed information, not all dimensions may be necessary in the solution of a specific problem. Eliminating irrelevant or low-information features can improve the model's performance and prevent overfitting. Additionally, feature selection can reduce the computational costs and memory requirements of the model, providing a significant advantage. In this context, we investigated the following 3 research questions (RQ) for this study:

RQ1. How can we enhance the effectiveness of vector representations by optimizing computational efficiency?

Our goal was to tackle the computational challenges associated with high-dimensional vectors, particularly when handling large datasets. By employing a sliding window method, we systematically examined recurring patterns within these vectors to enhance computational efficiency.

RQ2. Can we have insights into the nuanced integration of emotions within language representations of text units?

As detailed in Section 3, we investigated whether the method we applied to BERT vectors of words/sentences labeled with different emotions could detect emotional information in specific parts of the vector representations.

RQ3. What are the differences or similarities between the application of an optimization approach on vectors in the English and Turkish languages?

In the literature, while many methods used in the field of NLP on texts demonstrate success in the English language, it is observed that the same method may not yield the same success or effects when applied to different languages. Therefore, both for this reason and to make comparisons, we conducted experiments for the proposed method in both English and Turkish languages. The reason for choosing Turkish as a second language is that it differs significantly from English in terms of grammar. Among the general features of Turkish, its agglutinative structure, vowel harmony, and frequent usage of idioms and proverbs can be counted. For example, the 22-letter Turkish word "Anlamlandıramadıklarım." can be expressed in English as the 6-word sentence "What I couldn't make sense of.".

In summary, we examined whether certain dimensions within the representations of text units might include concealed information, such as emotions. This led us to explore the possibility of detecting emotional cues through a detailed analysis of these dimensions. To achieve this goal, we employed a sliding window approach to partition vectors and identify consistent patterns, aiming to enhance computational efficiency and gain a deeper understanding of the integration of emotions within these vectors. Our experiments involve emotion lexicon words and emotionally labeled sentences, and we also utilized BERT as an embedding model. Ultimately, this approach, which offers a new perspective on emotional representation, can be applied to any text unit, any embedding model, and any hidden information that can be detected. The contributions of the study can be listed as follows:

1. A dimensionality reduction technique through a sliding window approach is introduced to partition high-dimensional vector representations of texts into smaller sub-vectors, improving computational efficiency while maintaining or enhancing the effectiveness of representations.
2. Specific sub-vectors within BERT vectors that contain emotional information have been identified, suggesting that emotional clues are localized within certain dimensions of the vectors.
3. Experiments utilizing only sub-vectors are conducted in both English and Turkish, demonstrating the effectiveness of the proposed method for

two languages with different grammatical structures.

In the subsequent sections of the paper, Section 2 provides a literature review, Section 3 details the proposed method, Section 4 presents the experiments and results, and Section 5 concludes with the findings and implications.

2 LITERATURE REVIEW

Vector space models refer to the numerical representation of text units (like words or phrases) in a vector space. As can be seen in Figure 1, the models can be considered in two different groups: context-free and contextual models.

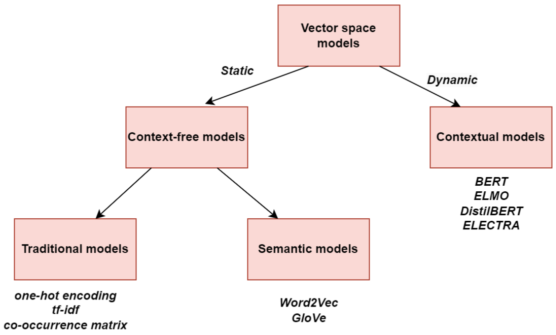


Figure 1: Vector space models.

From the context-free models, traditional models like one-hot encoding, tf-idf, and co-occurrence matrix representation lack semantic understanding. For instance, co-occurrence matrix representation counts word occurrences but fails to capture the nuances of word meanings and their semantic associations. Thus, these models struggle to comprehend the deeper meaning and context of language, which brings a drawback in tasks requiring semantic understanding, such as sentiment analysis and language translation. Semantic embeddings like Word2Vec and GloVe provide the representation of words with similar meanings close together in vector space. Capturing semantic relationships between words helps these models manage tasks like semantic similarity and word analogy. Although they have been a significant innovation in the field of NLP for containing semantic information, these models generate only a single static vector for each word. In other words, these models that produce context-free vectors do not consider polysemy and content.

Contextual models like BERT and ELMO produce different embeddings based on the context in which they are used, even for the same words with different meanings. These contextual models are designed to capture nuanced information in language and represent the complex relationships between words in various contexts. The representations are based on high-dimensional embeddings, typically ranging from 512 to 1024 dimensions. For instance, BERT has two versions: BERT-base with 768 dimensions and BERT-large with 1024 dimensions. Similarly, ELMO embeddings have 1024 dimensions. Two embedding models from GPT, *text-embedding-3-small*, and *text-embedding-3-large*, produce vectors with lengths of 1536 and 3072, respectively. While these high-dimensional embeddings capture rich and detailed linguistic information, they have challenges such as increased computational complexity and memory requirements. In the literature, dimensionality reduction techniques, such as PCA (Principal Component Analysis) and t-SNE (t- Stochastic Neighbor Embedding), are often used to address these issues while preserving the performance in several tasks (Raunak et al., 2019; Ayesha et al., 2020; George and Sumathy, 2022; Álvaro Huertas-García et al., 2022; Zhang et al., 2024). For example, (Zhang et al., 2024) study investigates the effects of reducing the dimensionality of high-dimensional sentence embeddings. The research assesses various unsupervised dimensionality reduction techniques, such as PCA, SVD (truncated Singular Value Decomposition), KPCA (Kernel PCA), GRP (Gaussian Random Projections), and autoencoders, to compress these embeddings. The aim is to cut down on storage and computational expenses while preserving performance in different downstream NLP tasks. Their findings indicate that PCA is the most efficient method, achieving a 50% reduction in dimensionality with only a 1% performance loss. Notably, for some sentence encoders, reducing dimensionality even enhanced accuracy. In the research conducted by (Su et al., 2021), they utilize a technique referred to as “whitening”, which is based on PCA (Principal Component Analysis), to process BERT sentence representations. This method reduces the embedding size to 256 and 384, aiming to address the issue of anisotropy and diminish dimensionality. Experimental results on seven benchmark datasets demonstrate that their method substantially enhances performance and reduces vector size, optimizing memory storage and accelerating retrieval speed.

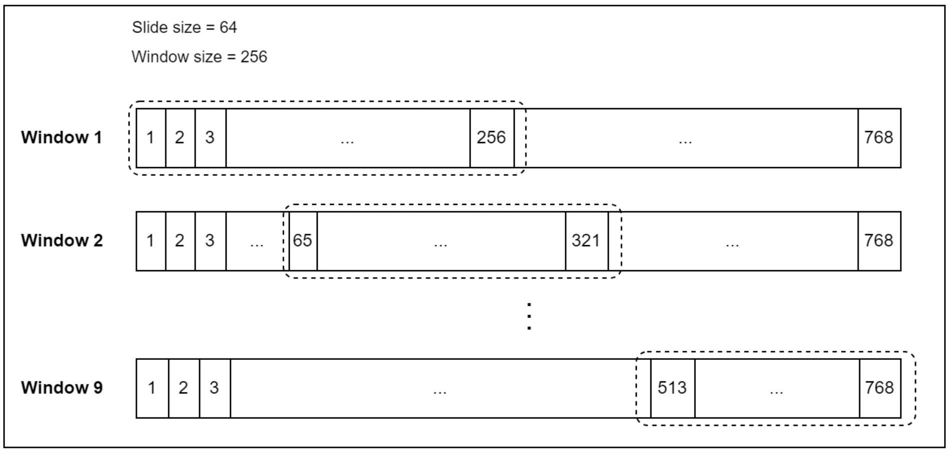


Figure 2: Framework for vector partitioning with sliding window technique.

3 METHOD: DIMENSIONALITY REDUCTION

Although contextual embeddings effectively capture both semantic and contextual knowledge, their high-dimensional vectors can be both space-consuming and computationally expensive, especially with large datasets. Additionally, specific dimensions or segments of these vectors might capture information related to specific features of language or properties of the text unit they represent. In this study, we proposed an alternative approach that emphasizes identifying patterns within vectors of any text unit, thereby reducing the complexity of the analysis. This approach is adaptable to any vectorization model.

We conducted an experimental study to find sub-vectors containing emotion information within BERT vectors of sentences and words labeled with different emotion categories (anger, fear, sadness, and joy) and measured the performance of word and sentence representations using only these sub-vectors. To perform a comparative study and observe the method's effectiveness in different languages, we conducted the experiments in both English and Turkish. Our proposed methodology is summarized as follows:

1. A sliding window technique is employed to examine and extract meaningful patterns from BERT vectors. This method divides the vectors into smaller, fixed-size parts (windows), enabling us to obtain local contextual information.
2. Cosine similarity between words (both for English and Turkish) labeled with the same emotion category is measured using only certain windows of BERT vectors for word representations. Here, an increase in cosine similarity values is expected if there is emotion-specific information in certain windows of the vectors.

To determine the window size for the sliding window technique we referred to the study of (Su et al., 2021). They proposed another dimensionality reduction technique to decrease BERT vectors to lengths of 256 and 384. Thus, in our study, the window size is selected as 256. Initially, BERT word vectors, labeled by 4 different emotion categories and having a length of 768, are divided into sub-vectors with a window size of 256. The slide size is determined to be 64 to cover every dimension of the BERT vectors. For example, the first sub-vector (window) starts at dimension 1 and ends at dimension 256, and the second one spans from dimension 65 to 321 as can be seen detailly in Figure 2. To sum up, employing the sliding window technique, we segmented the 768-dimensional word BERT vectors into nine subvectors.

4 EXPERIMENTS

In this study, we utilized the NRC English emotion lexicon (Mohammad and Turney, 2013) words and the Turkish-translated NRC emotion lexicon (TT-NRC) (Aka Uymaz and Kumova Metin, 2023). Both lexicons are annotated by Plutchick's (Plutchik, 1980) emotion categories. In the experimental study, we considered the lexicon words labeled by four emotion categories, namely anger, fear, sadness, and joy, for both languages. The initial step was obtaining BERT vectors of each lexicon word. Because BERT constructs vectors for words based on their surrounding context, the words and the sentences constituting the words should be given as parameters to BERT. We

Table 1: Pairwise in-category cosine similarity results of *English words* while using only one window.

		Windows								
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
In-category cosine similarity	**Anger-Anger**	0.249	0.597	0.628	0.633	0.630	0.361	0.256	0.244	0.233
	Fear-Fear	0.220	0.607	0.634	0.640	0.637	0.340	0.236	0.226	0.215
	Sadness-Sadness	0.236	0.598	0.629	0.636	0.633	0.357	0.254	0.250	0.242
	Joy-Joy	0.285	0.665	0.687	0.692	0.690	0.403	0.311	0.305	0.283

Table 2: Pairwise in-category cosine similarity results of *Turkish words* while using only one window.

		Windows								
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
In-category cosine similarity	**Anger-Anger**	0.288	0.330	0.300	0.324	0.312	0.767	0.766	0.768	0.775
	Fear-Fear	0.276	0.318	0.292	0.321	0.306	0.760	0.760	0.761	0.768
	Sadness-Sadness	0.275	0.317	0.295	0.321	0.302	0.760	0.760	0.762	0.770
	Joy-Joy	0.276	0.318	0.316	0.342	0.341	0.797	0.796	0.798	0.805

followed the same technique as (Aka Uymaz and Kumova Metin, 2023) for deriving BERT vectors utilizing the collection of three sentence datasets labeled by emotion four emotion categories (anger, fear, sadness, joy): TEI (Mohammad and Bravo-Marquez, 2017), TEC (Mohammad, 2012), and TREMO (Tocoglu and Alpkocak, 2018). After applying our proposed sliding window technique, we divided each BERT vector of lexicon words into 9 sub-vectors. Then, utilizing these sub-vectors individually to represent each word vector, we measured the pairwise cosine similarity score between each word belonging to emotion categories (in-category cosine similarity). Cosine similarity takes values between 0 and 1. 0 indicates that two vectors are completely different, while 1 means they are identical. In this study, a high cosine similarity score may indicate that certain sub-vectors are better at capturing that emotion category. For instance, when assessing cosine similarity between two words labeled with *joy*, we utilized only the subvectors spanning dimensions 1 to 256 and computed the cosine similarity. This procedure was repeated for other windows, resulting in nine cosine similarity experiments for each word represented by a single subvector. The outcomes were shown as heat maps in Tables 1 and 2 for English and Turkish lexicon words, respectively.

The heat maps reveal that certain dimensions within BERT vectors contain emotional clues. Consequently, employing specific subsets of these vectors in cosine similarity assessments yields higher similarity compared to others. This implies that focusing on subsets can be sufficient instead of utilizing all 768-dimensional vectors. Specifically, our examination of English word vectors identified emotional data within windows 2, 3, 4, and 5, while in Turkish, emotional intensity may also found within windows 6, 7, 8, and 9.

Following analyzing the in-category cosine similarity among lexicon words represented by a window-based vector, we applied these findings to a specific process in emotion identification: emotion enrichment of text units. The experimental study on emotion enrichment consists of two phases: sentence sub-vector construction and emotion enrichment on sentence vectors.

In this phase of the experimental study, we utilize the TEI (Mohammad and Bravo-Marquez, 2017), TEC (Mohammad, 2012), and TREMO (Tocoglu and Alpkocak, 2018) datasets. Among these, TREMO is a Turkish dataset, while the others are English datasets. To enable experiments with both English and Turkish, we translated the English datasets into Turkish and the Turkish dataset into English. Subsequently, we selected 500 sentences from each emotion category (anger, fear, sadness, joy) randomly, to construct the Emotion Sentence Dataset (ESD) used in the sentence-based experiments. In order to construct sentence sub-vectors, firstly, as an alternative to using the 768-dimensional BERT vectors for sentence representations, we utilized the sub-parts identified as having emotional information prior to word-based experiments for both English and Turkish as can be seen in detail in Figure 3. We combined the sub-parts that yielded the best results in each language. For instance, it was found that the English BERT vectors had more emotive information in sub-vectors 2, 3, 4, and 5. These sub-parts were concatenated to create a vector that spans from the start of the second window's dimensions to the end of the fifth window's dimensions. The process for combining these sub-vectors is illustrated in Figure 4.

Later, we observed the success of BERT vectors and sub-vectors from sentences in both languages in the emotion enrichment process (EEP). In studies on emotion classification or detection, emotion/sentiment enrichment is a frequently researched process in the literature (Agrawal et al., 2018; Wongpatikaseree et al., 2021; Matsumoto et al., 2022). It

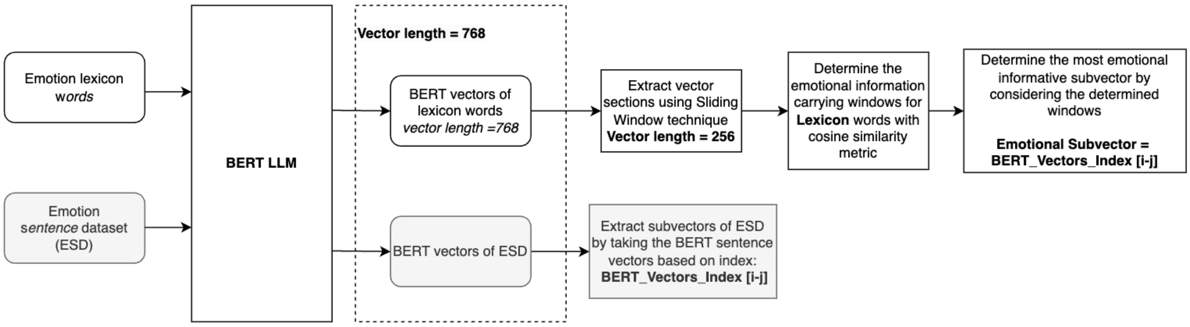

Emotion lexicon *words*
Emotion *sentence* dataset (ESD)
BERT LLM
Vector length = 768
BERT vectors of lexicon words *vector length =768*
BERT vectors of ESD
Extract vector sections using Sliding Window technique **Vector length = 256**
Determine the emotional information carrying windows for **Lexicon** words with cosine similarity metric
Determine the most emotional informative subvector by considering the determined windows
Emotional Subvector = BERT_Vectors_Index [i-j]
Extract subvectors of ESD by taking the BERT sentence vectors based on index: **BERT_Vectors_Index [i-j]**

Figure 3: Flowchart for dimensionality reduction for *word* and *sentence* vectors.

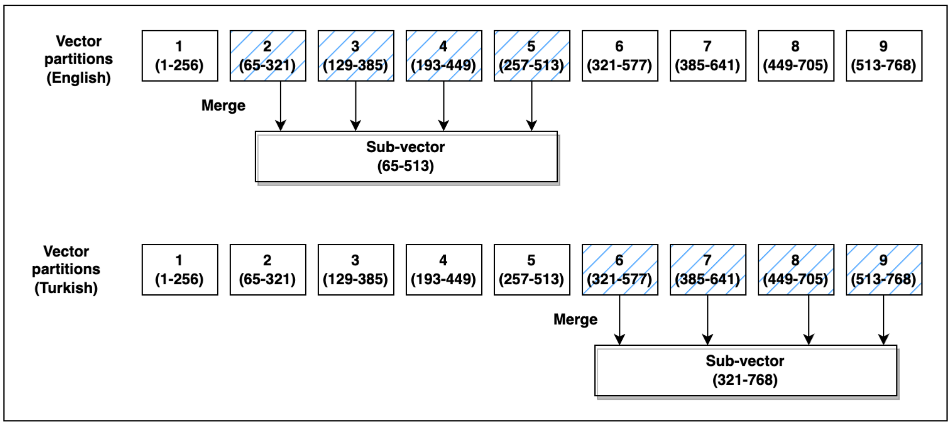


Figure 4: Framework for extracting sub-vectors.

has been observed in studies that although semantic and contextual embeddings demonstrate significant success in representing any text unit, they have some shortcomings in expressing emotional information. Therefore, it has been suggested that these vectors be enhanced by adding emotional information. Studies using cosine similarity-based or classification-based approaches with vectors containing emotional information have shown higher success. Various methods have been proposed in the literature. In this study, we applied the emotion enrichment method proposed by (Aka Uymaz and Kumova Metin, 2023) to our English and Turkish sentence datasets. In summary, this method works by comparing the vectors to be enriched with the vectors of emotion lexicon words. In this comparison, the similarity (cosine similarity) of each word to the emotional words in the lexicon is calculated. The closest emotional words are identified, and their vectors are used to enhance the original word's vector by weighting and averaging them based on their emotional relevance. Finally, a hybrid word representation is constructed by integrating semantic/contextual and emotional embeddings.

In the experiments involving the emotion enrichment process, we used Turkish and English sentences as the text units to be enriched with emotional information. Then, we calculated pairwise in-category cosine similarity scores within every emotion category before and after enrichment. For the vector representation of the sentences, we used 768-dimensional BERT vectors and the BERT sub-vectors obtained in the previous stage. The lexicons we used in the emotion enrichment process were the NRC and TT-NRC lexicons. We followed the same procedure for the vector representation of the lexicon words as we did for the sentences. That is, we first represented the lexicon words with BERT, then subjected the words to the enrichment process as in (Aka Uymaz and Kumova Metin, 2023), and finally obtained their sub-vectors.

Tables 3 and 4 present the emotion enrichment process of English and Turkish sentences by representing them with BERT and BERT sub-vectors. The first row in each table presents the average cosine similarity results within emotion categories for sentences, using BERT vectors of 768 lengths without additional enrichment. We used these values as a baseline and evaluated the outcomes of various enrichment combinations in comparison, showcasing the increments as percentages in the tables. In the second row,

Table 3: English Sentence embeddings enrichment with several combinations. (The best results are shown in bold.).

Sentence embedding	**Enrichment process**	**Enrichment by**	**In-category similarity (% improvement)**									
			Anger		**Fear**		**Joy**		**Sadness**		**Average**	
BERT	**-**	**-**	0,610	-	0,593	-	0,623	-	0,597	-	0,606	-
BERT	✓	**Emotion Lexicon Words (BERT + EEP)**	0,844	38,36%	0,838	41,32%	0,879	41,09%	0,845	41,54%	0,852	40,57%
BERT Subvector	✓	**Emotion Lexicon Words Subvector (BERT + EEP)**	**0,885**	**45,09%**	**0,880**	**48,44%**	**0,905**	**45,28%**	**0,883**	**47,88%**	**0,888**	**46,65%**

Table 4: Turkish Sentence embeddings enrichment with several combinations. (The best results are shown in bold.).

Sentence embedding	**Enrichment method**	**Enrichment by**	**In-category similarity (% improvement)**									
			Anger		**Fear**		**Joy**		**Sadness**		**Average**	
BERT	**-**	**-**	0,752	-	0,747	-	0,758	-	0,747	-	0,751	-
BERT	✓	**Emotion Lexicon Words (BERT + EEP)**	0,922	22,61%	0,931	24,63%	0,943	24,41%	0,927	24,10%	0,931	23,93%
BERT Subvector	✓	**Emotion Lexicon Words Subvector (BERT + EEP)**	**0,953**	**26,67%**	**0,959**	**28,45%**	**0,966**	**27,45%**	**0,956**	**28,03%**	**0,959**	**27,65%**

768-dimensional BERT vectors were subjected to the emotion enrichment process with 768-dimensional lexicon word vectors, while in the third line, alternatively, both sentence and lexicon word sub-vectors were used to represent and subjected to the emotion enrichment process. As can be seen, in both languages, the in-category cosine similarity results of emotionally enriched sentence vectors have yielded the best outcome when subvectors of both sentence and lexicon words' vectors are utilized for all four emotions. The best results in both languages have been observed in the *joy* emotion category with scores of 0,905 for English and 0,956 for Turkish. These results provide promising insights into the effectiveness of using sub-vectors instead of high-dimensional vectors, both for the emotion enrichment process and potentially reducing computational costs due to decreased vector size.

5 CONCLUSION

Natural language processing stands as a bridge between computer science, artificial intelligence, and linguistics, which focuses on machines that can comprehend and generate human language better through extensive analyses in various domains such as sentiment analysis, text summarization, and classification.

One of the most important processes in NLP studies is vectorization, which is simply the transformation of textual data into numerical representations, for any computational analysis. Unlike traditional methods like TF-IDF, newer techniques such as Word2Vec and BERT gained popularity because of having semantic and contextual knowledge, respectively, enriching the depth of linguistic representation. Especially, contextual models like BERT and its derivatives not only capture word semantics but also adapt to the nuanced contextual usage and polysemy, thereby addressing the limitations of traditional and semantic approaches.

However, the use of high-dimensional vectors poses computational challenges, particularly in large datasets. Feature selection and computational efficiency enhancements emerged as considerations as optimization strategies. In this context, we identified three research questions for our study:

RQ1. How can we enhance the effectiveness of vector representations by optimizing computational efficiency?

RQ2. Can we have insights into the nuanced integration of emotions within language representations of text units?

RQ3. What are the differences or similarities between the application of an optimization approach on vectors in the English and Turkish languages?

Firstly, related to *RQ1*, we proposed a sliding window technique to partition vectors into smaller, fixed-size parts, enabling the extraction of local contextual information. This method was evaluated through pairwise cosine similarity metric among emotion lexicon words which were annotated by four emotion categories, using both English and Turkish for addressing *RQ3*.

Our experimental findings as an answer to *RQ2* revealed that utilizing BERT vectors demonstrated that certain dimensions are more informative regarding emotional content. This suggests that using subvectors may effectively capture emotional clues and nuances in the languages, potentially reducing the need to utilize entire high-dimensional vector representations.

In the subsequent phase, we applied our findings to sentence vectors, constructing sentence sub-vectors based on the identified emotional dimensions (according to determined windows) from the word-based experiments. Then, to test our hypothesis on the effectiveness of using specific vector segments, we conducted experiments with these subvectors in comparison to using the original vectors in a case study related to the emotion enrichment process on vectors. This process simply incorporates vectors with additional emotional information. The comparative analysis between English and Turkish highlighted the adaptability of our method to different languages, acknowledging the grammatical and structural differences of Turkish.

When we examined the experimental results, we found that using specific sub-vectors instead of the original BERT vectors was both sufficient and could improve performance in cosine similarity calculations within emotion categories at both the word and sentence levels. As far as we know, this perspective and method have not been previously studied in terms of their applicability to any text unit represented by any vectorization method. Additionally, this approach is might be effective in capturing different types of information in vector representations and adapting to different problems.

In future studies, similar experiments can be conducted on other large language models (e.g., GPT models (OpenAI, 2023), RoBERTa (Liu et al., 2019), ELMO (Peters et al., 2018)) that have shown successful results in the literature. This approach may enable the investigation of different sub-vectors containing emotional information in these models and to get new perspectives. In our study, we carried out comparative analyses on English, a language rich in resources, and Turkish, an agglutinative language with fewer resources and a different grammatical structure. This study can be expanded to include languages from different language families and with various features. Additionally, vectors can be reanalyzed for different problems or information searches and the effectiveness of the approach in various scenarios can be examined.

REFERENCES

Agrawal, A., An, A., and Papagelis, M. (2018). Learning emotion-enriched word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 950–961, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Aka Uymaz, H. and Kumova Metin, S. (2023). Emotion-enriched word embeddings for Turkish. *Expert Systems with Applications*, 225:120011.

Ayesha, S., Hanif, M. K., and Talib, R. (2020). Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, 59:44–58.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

George, L. and Sumathy, P. (2022). An integrated clustering and bert framework for improved topic modeling.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.

Matsumoto, K., Matsunaga, T., Yoshida, M., and Kita, K. (2022). Emotional similarity word embedding model for sentiment analysis. *Computación y Sistemas*, 26(2).

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*.

Mohammad, S. (2012). #emotional tweets. *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM)*.

Mohammad, S. and Bravo-Marquez, F. (2017). Emotion intensities in tweets. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pages 65–77, Vancouver, Canada. Association for Computational Linguistics.

Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

OpenAI (2023). Gpt-large language model.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *In EMNLP*.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, New Orleans, Louisiana. Association for Computational Linguistics.

Plutchik, R. (1980). A general psychoevolutionary theory of emotion. In Plutchik, R. and Kellerman, H., editors, *Theories of Emotion*, pages 3–33. Academic Press.

Raunak, V., Gupta, V., and Metze, F. (2019). Effective dimensionality reduction for word embeddings. In Augenstein, I., Gella, S., Ruder, S., Kann, K., Can, B., Welbl, J., Conneau, A., Ren, X., and Rei, M., editors, *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Su, J., Cao, J., Liu, W., and Ou, Y. (2021). Whitening sentence representations for better semantics and faster retrieval.

Tocoglu, M. and Alpkocak, A. (2018). Tremo: A dataset for emotion analysis in Turkish. *Journal of Information Science*, 44:016555151876101.

Wongpatikaseree, K., Kaewpitakkun, Y., Yuenyong, S., Matsuo, S., and Yomaboot, P. (2021). Emocnn: Encoding emotional expression from text to word vector and classifying emotions—a case study in thai social network conversation. *Engineering Journal*, 25(7):73–82.

Zhang, G., Zhou, Y., and Bollegala, D. (2024). Evaluating unsupervised dimensionality reduction methods for pretrained sentence embeddings.

Álvaro Huertas-García, Martín, A., Huertas-Tato, J., and Camacho, D. (2022). Exploring dimensionality reduction techniques in multilingual transformers.

Text-Based Feature-Free Automatic Algorithm Selection

Amanda Salinas-Pinto[1][a], Bryan Alvarado-Ulloa[1][b], Dorit Hochbaum[2][c],
Matías Francia-Carramiñana[1][d], Ricardo Ñanculef[1][e] and Roberto Asín-Achá[1][f]
[1]*Universidad Técnica Federico Santa María, Chile*
[2]*University of California, Berkeley, U.S.A.*
{*amanda.salinas, bryan.alvarado*}*@usm.cl, dhochbaum@berkeley.edu,*
{*matias.francia, ricardo.nanculef, roberto.asin*}*@usm.cl*

Keywords: Algorithm Selection, Deep Learning, SAT, CSP.

Abstract: Automatic Algorithm Selection involves predicting which solver, among a portfolio, will perform best for a given problem instance. Traditionally, the design of algorithm selectors has relied on domain-specific features crafted by experts. However, an alternative approach involves designing selectors that do not depend on domain-specific features, but receive a raw representation of the problem's instances and automatically learn the characteristics of that particular problem using Deep Learning techniques. Previously, such raw representation was a fixed-sized image, generated from the input text file specifying the instance, which was fed to a Convolutional Neural Network. Here we show that a better approach is to use text-based Deep Learning models that are fed directly with the input text files specifying the instances. Our approach improves on the image-based feature-free models by a significant margin and furthermore matches traditional Machine Learning models based on basic domain-specific features, known to be among the most informative features.

1 INTRODUCTION

Automatic Algorithm Selection (AAS) aims to predict the optimal solver for a given problem instance from a portfolio. Traditionally, this process relies on domain-specific features crafted by experts, which, while effective, limits scalability and transferability due to the need for extensive domain knowledge and labor-intensive analysis.

Recent advances in Deep Learning (DL) (Vaswani et al., 2017), where models learn from raw data, offer a compelling alternative to feature-based models. Previous work (Loreggia et al., 2016) in AAS has transformed raw data into fixed-sized images processed by Convolutional Neural Networks (CNNs), but this still requires image-processing techniques.

Our study introduces a novel text-based deep learning approach that directly processes raw textual files specifying problem instances, simplifying the computational pipeline, and enhancing representation.

In this paper, we present our text-based deep learning framework for AAS and evaluate its performance against traditional image-based and feature-based models. Our analysis shows that text-based models are superior in capturing complex information in problem descriptions, leading to more effective and adaptable algorithm selection strategies as compared to image-based methods. Nevertheless, there is still a gap in performance as compared to specialized feature-base models, and closing this gap will still be the base of future research in the area of feature-free algorithm selection.

Our contributions include demonstrating the feasibility of text-based deep learning for AAS and providing a thorough analysis of how these techniques outperform existing feature-free methods. We establish new benchmarks, advancing the field of feature-free AAS, and offer insights into the performance gap between feature-free and feature-based methodologies.

The subsequent sections review relevant literature, define key terms and criteria, outline our text-based AAS framework, present empirical assessments, and conclude with findings and future research directions.

[a] https://orcid.org/0009-0007-2216-4371
[b] https://orcid.org/0009-0008-7468-5723
[c] https://orcid.org/0000-0002-2498-0512
[d] https://orcid.org/0009-0000-8680-7347
[e] https://orcid.org/0000-0003-3374-0198
[f] https://orcid.org/0000-0002-1820-9019

Salinas-Pinto, A., Alvarado-Ulloa, B., Hochbaum, D., Francia-Carramiñana, M., Ñanculef, R. and Asín-Achá, R.
Text-Based Feature-Free Automatic Algorithm Selection.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 267-274
ISBN: 978-989-758-716-0; ISSN: 2184-3228

2 RELATED WORK

2.1 Algorithm Selection Systems

Automatic Algorithm Selection (AAS), introduced by (Rice, 1976), optimizes computational processes by selecting the most suitable algorithm for a given problem instance. This approach is rooted in the "No Free Lunch" theorem (Adam et al., 2019), which posits that no single algorithm universally excels across all scenarios.

AAS typically employs a training phase to associate problem instance features with algorithm performance. The trained model then evaluates new instances to predict the most effective algorithm. Recent literature has explored AAS in various domains, including timetabling (Seiler et al., 2020; Bossek and Neumann, 2022), SAT (Xu et al., 2008), and Multi-Agent Path-Finding (Bulitko, 2016; Achá et al., 2022).

Kerschke et al. (Kerschke et al., 2019) provide a comprehensive survey of algorithm selection and configuration, introducing a taxonomy that distinguishes between "per-set" and "per-instance" methods. Our focus is on "per-instance" AAS, which considers each problem instance individually.

While many AAS systems employ complex strategies, such as the hybrid methodology of semi-static solver schedules (3S) (Kadioglu et al., 2011) or Autofolio (Lindauer et al., 2015), our study concentrates on straightforward approaches. We assume an ML model receives an instance characterization and selects a single solver to execute until completion or time limit.

Most AAS research relies on domain-specific, expert-crafted features. However, an alternative approach involves developing ML methods that utilize raw/generic instance representations, allowing the learning process to identify relevant features autonomously. This approach was first explored by (Loreggia et al., 2016).

2.2 Deep Learning for Algorithm Portfolios

(Loreggia et al., 2016) introduced a groundbreaking approach to Automatic Algorithm Selection (AAS) based on deep learning. Unlike traditional AAS techniques that use hand-crafted, domain-specific features, this method leverages generic raw data — the text file contents describing the problems.

The process transforms text files into a fixed-size image format suitable for Convolutional Neural Network (CNN) analysis:

1. Convert textual input into a vector of ASCII codes.
2. Reorganize the vector into a $\sqrt{N} \times \sqrt{N}$ matrix, where N is the total character count.
3. Resize the resulting "ASCII image" to a uniform scale.

The CNN can be trained as a multi-class classifier, multi-label classifier, or regressor. Evaluated using SAT and Constraint Satisfaction Problems (CSP) instances, this method showed potential to outperform the Single Best Solver (see Subsection 2.3).

Despite its successes, this approach may not perform as well as methods utilizing domain-specific features.

2.3 Performance Metric for Meta-Solvers

We define an algorithm-selection-based meta-solver as a system comprising a portfolio of solvers. It analyzes an input instance and runs one or more solvers to resolve it. A solver *solves* an instance if it can decide its satisfiability (for decision problems) or find and certify the optimal solution (for optimization problems) within a time limit.

All our meta-solvers here operate uniformly:

1. Accept an input instance.
2. Use an ML model to predict the most efficient solver, identify capable solvers, or estimate solving times.
3. Select and run one solver based on these predictions.

We evaluate the meta-solver's performance using two baselines:

Single Best Solver (SBS): The solver performing best on average across all training instances.

Virtual Best Solver (VBS): A hypothetical meta-solver always choosing the most effective algorithm for each instance.

Performance is measured using the PAR10 metric (Lindauer et al., 2019). For a solver s on instance i:

$$m_s(i) = \begin{cases} t_s(i) & \text{if } t_s(i) \leq \tau \\ 10\tau & \text{otherwise} \end{cases}$$

where τ is the timeout constant and $t_s(i)$ is the solving time.

We use the performance measure $\hat{m}$ (Lindauer et al., 2019) to evaluate meta-solvers:

$$\hat{m}_{ms} = \frac{m_{ms} - m_{VBS}}{m_{SBS} - m_{VBS}} \tag{1}$$

Values of $\hat{m}_{ms}$ close to 0 indicate performance near VBS, while values close to 1 suggest performance similar to SBS. Values above 1 indicate the meta-solver is less effective than SBS.

3 TEXT-BASED FEATURE-FREE AAS

We follow (Loreggia et al., 2016)'s approach, working directly with raw problem instance representations. Our Deep Learning models are fed with raw text representations, rather than pre-processed image-like inputs.

3.1 Architecture Overview

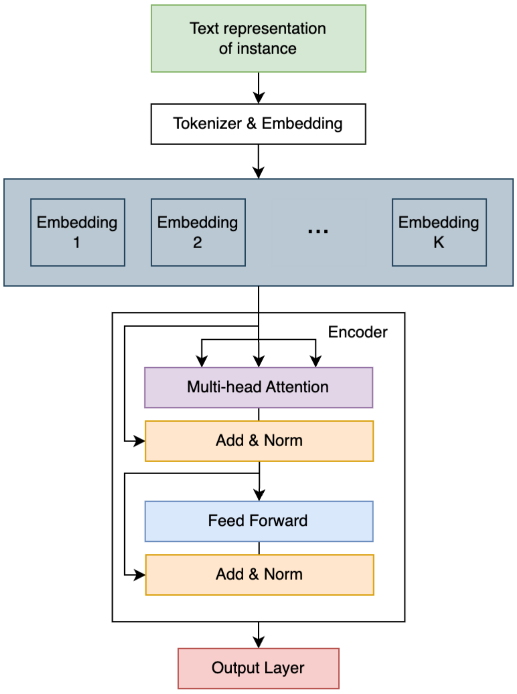


Figure 1: Overall architecture of our text-based Deep Learning Model for AAS.

Our architecture (Figure 1) is a modified Transformer neural network, using only the encoder component similar to the one of (Vaswani et al., 2017). The input text x is truncated, tokenized, and converted into embeddings $\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \rangle$. The encoder's outputs $\mathbf{z} = \langle \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n \rangle$ are fused into a global descriptor $\mathbf{z}$ using Global Max Pooling (Christlein et al., 2019), then mapped to a prediction through a fully connected output layer.

3.2 Tokenizers and Embeddings

We explore two tokenization approaches:

Pre-trained Tokenization: Using *SentencePiece* (Kudo and Richardson, 2018).

Trained Tokenization: Using *Charformer* (Tay et al., 2021).

3.3 Encoder Architecture

Our encoder computes $M = 4$ hierarchical transformations $\mathbf{Z}^{(k)} = \text{EBlock}(\mathbf{Z}^{(k-1)})$. Each block includes a self-attention mechanism and a position-wise feed-forward net. The self-attention mechanism computes:

$$\mathbf{P} = \text{SelfAttention}(\mathbf{Z}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d'}}\right)\mathbf{Z}, \quad (2)$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{Z}$ are learnable matrices that project $\mathbf{Z}^{(k-1)}$ into a d'-dimensional latent space. We use multi-head attention with $H = 4$ heads. The final block's output $\mathbf{Z}^{(k)}$ is obtained after applying a residual connection (He et al., 2016) and layer normalization (Ba et al., 2016) around each sublayer. We did not use positional embeddings.

3.4 Problem Framing Strategies

We explore three strategies:

Multi-Class Classification: Identifies the most suitable solver and the meta-solver runs it. The output layer is a softmax function, and the loss function is categorical cross-entropy.

Multi-Label Classification: Identifies all solvers capable of solving the instance within the defined time limit τ. Each solver corresponds to an element in the output vector, with a sigmoid function applied element-wise. The loss is measured through the Hamming loss function. Since the probabilities here are not complementary, they determine the likelihood that a solver will be fit for the problem instance. The meta-solver executes the solver that exhibits the highest likelihood.

Regression: Estimates normalized log delta runtime for each solver. The mean squared error function serves as the loss function, and the output layer is linear. The meta-solver runs the solver predicted to have the shortest runtime.

$$r_{s,i} = log(1 + m_s(i) - \min_{s \in S}(m_s(i)))$$

$$y_{s,i} = \frac{r_{s,i} - mean(r_{s,i})}{std(r_{s,i})}$$

4 EXPERIMENTAL SETUP AND BASELINES

4.1 Libraries and Hardware

We implemented our Deep Learning models in Python 3.10, using PyTorch 2.0.0. For the text-based models, we used the Charformer tokenizer 0.0.4[1], and SentencePiece 0.2.0. For the image scaling, needed by the image-based models, we used OpenCV 4.7.0.72. The feature-based models were implemented using scikit-learn 1.4.2.

The experiments were carried out on a machine with an Intel Xeon Skylake (2x16 @2.1 GHz) processor and an Nvidia A40 GPU. The machine runs Scientific Linux 7 and has 48GB of RAM.

4.2 Benchmark Sets

To evaluate our approach, first, we aimed to use the same benchmark sets used in (Loreggia et al., 2016). However, the precise sets of instances and the partitions used in that study were not disclosed publicly and could not be provided by the authors when asked in an internal communication. We then searched for similar-nature benchmarks for which the instance files and hand-crafted features used in the AAS community were available. Unfortunately, we could not find meaningful benchmark sets similar to the ones named “SAT Random” and “SAT Crafted” in (Loreggia et al., 2016). However, we were able to collect the most interesting benchmark sets reported in such study, “SAT Industrial” and “CSP”. These benchmark sets are the more interesting because of their diversity in size, complexity, and complementarity of the solvers.

SAT Industrial. This benchmark includes instances used in the SAT competition between 2003 and 2016 in the industrial/application categories. The performance of the solvers in these competitions was retrieved from *ASLib*, specifically from the *SAT03-16-INDU-ALGO* scenario. We removed 269 instances that could not be solved by any solver in the portfolio within the given τ time limit. After filtering, the dataset contains $1,730$ instances and 10 different solvers.

CSP. We used the benchmark from the 2009 CSP competition[2]. The performance data for each solver was obtained from the *PROTEUS-2014* scenario (Hurley et al., 2014) in *ASlib*. We filtered the instances by removing the “easy” instances that could be solved by all solvers within the time-limit equivalent to compute the instance’s features, in addition to removing the “difficult” instances that were not solved by any of the solvers within the given time limit τ. This resulted in a total of $1,613$ instances and 22 different solvers.

4.3 Data Partitioning and Evaluation Criteria

We split each benchmark into train and test datasets. For the train dataset we used 80% of the total instances, and the remaining 20% is reserved as the test dataset. The training dataset is used for training and model selection, while the test dataset is used to compare the in-production performance of the best text-based, image-based, and feature-based approaches.

To select the best model for each approach, we performed 10-fold cross-validation with the training set. We compared the models based on the $\hat{m}$ metric associated with a meta-solver using them. We then selected the best model based on the mean $\hat{m}$ metric across the different folds.

4.4 Feature-Based Models

To offer a comprehensive view of our study on feature-free models, we also implement and evaluate feature-based models employing both state-of-the-art crafted features and basic informative features, using Random Forest models. The comparison of feature-free models with these feature-based counterparts serves a dual purpose: firstly, to analyze and document the performance disparities between these two paradigms, and secondly, to provide the research community with a benchmark on the effectiveness of applying state-of-the-art crafted features in a straightforward manner on ASLib scenarios that are widely used.

Basic Features: Two basic features extracted from the text describing a problem instance are: the number of variables and the number of constraints. The motivation for these two features is that the instance size usually appears among the most simple and informative ones. We expect that

[1] as implemented in https://github.com/lucidrains/charformer-pytorch

[2] https://www.cril.univ-artois.fr/CSC09/results/globalbybench.php?idev=30&idcat=38&idSubCat=60

training ML models on these two features establishes a baseline for the other methods.

We note here that, for the CSP benchmark, the number of variables and constraints in the text file differ from the *direct_nvariables* and *direct_nclauses* features of the ASLib scenario, since the former seem to be computed after grounding the CSP formula to SAT.

Full Set of Features: These features represent the state-of-the-art in domain-specific algorithm selection, as provided in the corresponding scenarios of *ASLib*. All these 483 SAT features were introduced in (Xu et al., 2008), and constitute the by-default standard for AAS in SAT.

For CSP, ASLib provides the 198 domain-specific features as proposed in (Hurley et al., 2014). We note here that our evaluation does not consider the time needed to compute all these features, even though some of them are expensive to compute and others are captured during runtime, from a reference solver.

Although these features and scenarios are commonly referenced in the literature, we were unable to find reported performance values ($\hat{m}$) for meta-solvers that utilize these features directly. Consequently, our aim is to document these values to serve as a reference for future research.

4.5 Image-Based Models

We implemented the approach presented by (Loreggia et al., 2016) carefully following the experimental setup described there. For the training, we used Stochastic Greedy Descent (SGD) with Nesterov momentum of 0.9 and a learning rate of 0.03. As first layer, we included a batch normalization layer, as proposed in (Ioffe and Szegedy, 2015). The output layer changes depending on the learning task, as mentioned in Section 3.4. We set a training batch size of 128 and 100 epochs.

4.6 Text-Based Models

Due to limitations on the hardware needed to train our model on arbitrary-size instances, we truncated the size of the instances to $10,000$ characters. To avoid introducing biases into the model, we removed from the text files any comments or other kind of meta-information like the folder name where the instance is located, or the name of the generator of the instance. In a preliminary evaluation, we noted that these meta-information fields may unfairly help the text-based models and decided not to consider this information.

For training our text-based models, we used AdamW optimizer (Loshchilov and Hutter, 2017) with a *learning rate* of 10^{-5}. We set a batch size of 8 samples and an *embedding size d* of 128. The training was set to take 100 *epochs*.

Since the sequence length produced by SentencePiece can vary among instances while our encoder accepts a fixed-length sequence, we computed the median length of SentencePiece's output, truncating longer sequences and padding shorter ones. The vocabulary size v for SentencePiece, was set to 1024. Charformer, which operates at the character level, had a vocabulary size of 257 (256 ASCII values plus one token reserved for padding). We set the *max block size* and the *downsample factor* to their default values (4). Additionally, we employed the block attention scores proposed in Section 2.1.4 of (Tay et al., 2021) to form latent subwords.

5 RESULTS

5.1 Feature-Based Validation Results

Table 1: $\hat{m}$ metric values across 10-fold validation sets for different handcrafted-features-based meta-solvers for CSP and SAT Industrial benchmark sets. Here, HF = Handcrafted-based, F=Full set of features, B=Basic set of features, ML=Multi-label model, Reg= Regression model, MC=Multi-class model.

Model	CSP	SAT Industrial
HF-F-ML	**0.409 ± 0.064**	0.680 ± 0.312
HF-F-Reg	0.416 ± 0.087	**0.640 ± 0.291**
HF-F-MC	0.546 ± 0.066	0.676 ± 0.228
HF-B-ML	0.638 ± 0.068	1.054 ± 0.361
HF-B-Reg	**0.557 ± 0.066**	**0.939 ± 0.365**
HF-B-MC	0.582 ± 0.075	1.22 ± 0.387

Table 1 shows the average and standard deviation of the $\hat{m}$ values computed across 10-fold cross-validation subsets for six feature-based meta-solvers. The first three meta-solvers are based on the full set of features provided in the ASLib, while the last three meta-solvers only use the two basic features related to the size of the instances. For a fair comparison with our feature-free model, these feature-based meta-solvers can cast AAS as a multi-label task (ML), a regression task (Reg), or a multi-class (ML) problem. As can be seen, for the CSP benchmark, the most successful meta-solver using the full set of features is the one based on multi-label classification (ML). In contrast, for the SAT Industrial benchmark, the best meta-solver, using the full set of features, is the one based on regression (Reg). Nevertheless, we note that even for these state-of-the-art crafted features, the meta-solvers are quite sensible to the test set in SAT, as is evident from the considerable standard deviation.

Regarding the meta-solvers using only the two basic features, the meta-solvers based on regression show better performance in both benchmark sets. We note that, on average, only using these two basic features allows the meta-solvers to outperform the SBS. For CSP, we found a considerable margin of advantage, and for SAT Industrial, a smaller margin.

We report the performance of these feature-based solvers in the test set in Subsection 5.3. All the results reported here are consistent with the literature.

5.2 Image-Based Validation Results

Table 2: $\hat{m}$ metric values across 10-fold validation sets for different image-based meta-solvers for CSP and SAT Industrial benchmark sets. Here, Im = Image-based, ML=Multi-label model, Reg= Regression model, MC=Multi-class model.

Model	**CSP**	**SAT Industrial**
Im-ML	0.640 ± 0.088	1.25 ± 0.407
Im-Reg	**0.609 ± 0.104**	**1.14 ± 0.346**
Im-MC	0.898 ± 0.109	1.66 ± 0.527

Table 2 shows the statistics of the $\hat{m}$ values computed across 10-fold cross-validation subsets for three image-based meta-solvers. For a fair comparison with our text-based model, we trained image-based meta-solvers based on multi-label, regression, and multi-class formulations. The results in Table 2 demonstrate that, although the regression approach was not considered in (Loreggia et al., 2016), the most successful image-based meta-solver is the one based on regression for both benchmark sets.

The meta-solver for CSP outperforms CSP's Single Best Solver by a significant margin while maintaining a considerable gap with the Virtual Best Solver for CSP. These results are in line with the ones reported in (Loreggia et al., 2016). However, an exact match between our image-based results and those in (Loreggia et al., 2016) is virtually impossible since the training/validation/test differ.

Image-based SAT meta-solvers cannot outperform the Single Best Solver. This result diverges from the results of (Loreggia et al., 2016), which reported an image-based meta-solver that outperforms SBS on SAT. This discrepancy may happen due to differences in the specific SAT industrial benchmark set used or differences in the training/test partitions. However, we also observe that the performance of the SAT image-based meta-solver varies significantly depending on the training and validation set (standard deviation of 0.346 among cross-validation folds).

5.3 Text-Based Validation Results

Table 3: $\hat{m}$ metric values across 10-fold validation sets for different text-based ML models for CSP and SAT Industrial benchmark sets. Here, Txt = Text-based, Cha=Trained tokenizer Charformer, Sen= Pre-trained tokenizer Sentenpiece, ML=Multi-label model, Reg= Regression model, MC=Multi-class model.

Model	**CSP**	**SAT Industrial**
Txt-Cha-ML	0.488 ± 0.047	0.952 ± 0.281
Txt-Cha-Reg	**0.469 ± 0.050**	**0.889 ± 0.303**
Txt-Cha-MC	0.581 ± 0.076	1.312 ± 0.354
Txt-Sen-ML	0.482 ± 0.082	1.078 ± 0.252
Txt-Sen-Reg	0.536 ± 0.100	1.119 ± 0.448
Txt-Sen-MC	0.608 ± 0.120	1.470 ± 0.319

Table 3 shows the average and standard deviation of the $\hat{m}$ values for our text-based meta-solvers computed by 10-fold cross-validation. The first three meta-solvers are text-based models jointly trained with the tokenizer (Charformer), while the last three meta-solvers use the pre-trained tokenizer (SentencePiece). As can be seen, the most successful meta-solver is the one that uses a regression model jointly trained with the tokenizer.

The CSP meta-solver significantly improves the performance of the SBS for this domain. With an average $\hat{m}$ value equal to 0.469 and little standard deviation, this meta-solver's performance can be interpreted as closer to the VBS than to the SBS.

Despite the formulation, obtaining a $\hat{m}$ lower than 1 for SAT Industrial was impossible using image-based methods. Noticeably, our best text-based meta-solver outperforms the Single Best Solver with an average $\hat{m}$ value of 0.889 in this benchmark. Nevertheless, as for the previous models, the standard deviation is high (0.303), which suggests that the meta-solver's performance varies considerably depending on the validation instances used.

5.4 Test Set Results

Here we compare feature-based, image-based and text-based meta-solvers on the *test set* of each benchmark. For each category, we selected the best approach using 10-fold cross-validation, and trained the model with the whole training set. Again, we note that results given on feature-based models are reported as a reference to gain perspectives as well as to communicate performance values of meta-solvers using straightforward models.

As anticipated, the meta-solvers that yield the best

Table 4: $\hat{m}$ metric values of the testing set, for each "best" model for each approach and benchmark set.

Model	CSP	SAT Industrial
HF-B-Reg	0.549	0.975
HF-F-ML	**0.442**	—
HF-F-Reg	—	**0.674**
Im-Reg	0.642	1.309
Txt-Char-Reg	**0.556**	**1.037**

results are those that utilize expert-designed features specific to the domain. In the case of CSP, the meta-solver employing a multi-label classification model achieves an $\hat{m}$ value of 0.442. This significantly narrows the performance disparity between the SBS and the VBS in CSP scenarios. Similarly, for the SAT Industrial benchmark, the regression-based meta-solver records an $\hat{m}$ value of 0.674. Considering the complexity of this benchmark, this score is notably satisfactory. These outcomes align with those from contemporary meta-solvers specialized for CSP and SAT Industrial. It is important to note that this assessment only gauges the effectiveness of the features in a well-adjusted ML model. This overview omits the consideration that many sophisticated features, while beneficial, are computationally intensive and may not be regularly employed in elaborate Algorithm Selection Systems that utilize both a presolver and a solver scheduler. Hence, the current $\hat{m}$ values of the meta-solvers that incorporate these advanced features likely represent a lower bound for any straightforward methodology.

When comparing the two feature-free meta-solvers, our text-based method significantly surpasses the image-based method and nearly matches the performance of the meta-solvers that incorporate the two basic crafted features. This suggests that the image-based models may fail to capture even basic information, such as the size of the problem instance. Conversely, the text-based models appear capable of recognizing information akin to these features, even though our system uses only basic vanilla encoders. Converting these $\hat{m}$ scores to average running times reveals that the expected average time for the text-based model is approximately 13% lower than that of the image-based model for the CSP benchmark. For the SAT Industrial benchmark, this reduction is about 20%. Collectively, these figures demonstrate that our novel text-based feature-free framework significantly decreases the performance gap between feature-free and feature-based Algorithm Selection Systems (AAS).

6 CONCLUSIONS AND FUTURE WORK

We present here a novel approach to Automatic Algorithm Selection that leverages the capabilities of text-based deep learning models. Our results clearly demonstrate that this method not only simplifies the feature extraction process (by eliminating the need of image-based preprocessing) but also significantly enhances the performance of existing feature-free algorithm selection paradigms. By directly processing raw textual descriptions of problem instances, our approach has shown a marked improvement over traditional, image-based CNN approaches in terms of both performance and robustness across benchmarks.

The effectiveness of our method was validated through extensive experiments on benchmarks containing a variety of problem instances. The experimental results underscore the potential of deep learning techniques that operate directly on raw data, providing a more scalable and flexible end-to-end solution for the field of AAS.

Our experiments confirm that, up to date, no feature-free algorithm selection approach can outperform meta-solvers based on validated domain-specific crafted features by experts. However, results also show that text-based feature-free models can match the performance of meta-solvers based on basic informative features. This finding suggests that deep learning methods can learn problem representations beyond the most crude and elementary characterization.

While our study has made significant strides in the application of text-based models to algorithm selection, several avenues remain open for further exploration. Future work may include:

- **More Complex AAS Systems:** Our proposal can be the base for more complex AAS systems, including dynamic portfolios and schedulers.
- **More Complex ML Models:** More complex transformer architectures can also be tested. Besides, AAS can be framed in a more sophisticated way to leverage advances in ranking, metric learning, and recommender systems.
- **Handling the Whole Text Files:** A plethora of architectures have been proposed for long text modeling in deep learning. These methods should be systematically evaluated to overcome the limitations of our text-based meta-solver.
- **Anytime AAS:** Extending our method to Anytime Algorithm Selection could significantly benefit environments where decisions should be made based on the available computational resources.

- **Transfer Learning:** Exploring transfer learning techniques to adapt models trained on one set of problem instances to handle others effectively could contribute to a general purpose AAS.
- **Interpretable AI Models:** Enhancing the interpretability of deep learning models used in AAS to provide insights into why certain algorithms are preferred for specific instances could help refine the models further and in gaining trust from users.
- **Benchmarks and Datasets:** Applying our framework to other domains, possibly including optimization problems whose domain metrics $\hat{m}$ involve the values of the objective function.

In conclusion, the research presented in this paper sets a new benchmark in the field of feature-free AAS and opens up numerous possibilities for the evolution of more intelligent and autonomous algorithm selection systems. Our future efforts will focus on expanding the capabilities of our framework and exploring these promising directions to further enhance the field of algorithm selection.

ACKNOWLEDGEMENTS

Authors 1st, 2nd, 3rd, 4th, and 6th are supported in part by AI institute NSF award 2112533.

REFERENCES

Achá, R. A., López, R., Hagedorn, S., and Baier, J. A. (2022). Multi-agent path finding: A new boolean encoding. *Journal of Artificial Intelligence Research*, 75:323–350.

Adam, S. P., Alexandropoulos, S.-A. N., Pardalos, P. M., and Vrahatis, M. N. (2019). No free lunch theorem: A review. *Approximation and optimization: Algorithms, complexity and applications*, pages 57–82.

Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bossek, J. and Neumann, F. (2022). Exploring the feature space of tsp instances using quality diversity. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 186–194.

Bulitko, V. (2016). Evolving real-time heuristic search algorithms. In *Artificial Life Conference Proceedings 13*, pages 108–115. MIT Press.

Christlein, V., Spranger, L., Seuret, M., Nicolaou, A., Král, P., and Maier, A. (2019). Deep generalized max pooling. In *2019 International conference on document analysis and recognition (ICDAR)*, pages 1090–1096. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hurley, B., Kotthoff, L., Malitsky, Y., and O'Sullivan, B. (2014). Proteus: A hierarchical portfolio of solvers and transformations. In *Integration of AI and OR Techniques in Constraint Programming: 11th International Conference, CPAIOR 2014, Cork, Ireland, May 19-23, 2014. Proceedings 11*, pages 301–317. Springer.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr.

Kadioglu, S., Malitsky, Y., Sabharwal, A., Samulowitz, H., and Sellmann, M. (2011). Algorithm selection and scheduling. In *International Conference on Principles and Practice of Constraint Programming*, pages 454–469. Springer.

Kerschke, P., Hoos, H. H., Neumann, F., and Trautmann, H. (2019). Automated algorithm selection: Survey and perspectives. *Evolutionary computation*, 27(1):3–45.

Kudo, T. and Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Lindauer, M., Hoos, H. H., Hutter, F., and Schaub, T. (2015). Autofolio: An automatically configured algorithm selector. *Journal of Artificial Intelligence Research*, 53:745–778.

Lindauer, M., van Rijn, J. N., and Kotthoff, L. (2019). The algorithm selection competitions 2015 and 2017. *Artificial Intelligence*, 272:86–100.

Loreggia, A., Malitsky, Y., Samulowitz, H., and Saraswat, V. (2016). Deep learning for algorithm portfolios. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Rice, J. R. (1976). The algorithm selection problem. In *Advances in computers*, volume 15, pages 65–118. Elsevier.

Seiler, M., Pohl, J., Bossek, J., Kerschke, P., and Trautmann, H. (2020). Deep learning as a competitive feature-free approach for automated algorithm selection on the traveling salesperson problem. In *International Conference on Parallel Problem Solving from Nature*, pages 48–64. Springer.

Tay, Y., Tran, V. Q., Ruder, S., Gupta, J., Chung, H. W., Bahri, D., Qin, Z., Baumgartner, S., Yu, C., and Metzler, D. (2021). Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Xu, L., Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2008). Satzilla: portfolio-based algorithm selection for sat. *Journal of artificial intelligence research*, 32:565–606.

Flow Is Best, Fast and Scalable: The Incremental Parametric Cut for Maximum Density and Other Ratio Subgraph Problems

Dorit S. Hochbaum[a]
Department of Industrial Engineering and Operations Research, University of California, Berkeley, U.S.A.
dhochbaum@berkeley.edu

Keywords: Densest Subgraph, Graph Structures, Monotone Integer Programming, Breakpoints Algorithm, Conductance.

Abstract: The maximum density subgraph, or densest subgraph, problem has numerous applications in analyzing graph and community structures in social networks, DNA networks and financial networks. The densest subgraph problem has been the subject of study since the early 80s and polynomial time flow-based algorithms are known, yet research in the last couple of decades has been focused on developing heuristic methods for solving the problem claiming that flow computations are computationally prohibitive. We introduce here a new polynomial time algorithm, the *incremental parametric cut* algorithm (IPC) that solves the maximum density subgraph problem and many other max or min ratio problems in the complexity of a single minimum-cut. A characterization of all these efficiently solvable ratio problems is given here as problems with monotone integer programming formulations. IPC is much more efficient than the parametric cut algorithm since instead of generating all *breakpoints* it explores only a tiny fraction of those breakpoints. Compared to the heuristic methods, IPC not only guarantees optimality, but also runs orders of magnitude faster than the heuristic methods, as shown in an accompanying experimental study.

1 INTRODUCTION

We introduce here a new efficient algorithm for the maximum density (MD), or densest, subgraph problem and many other ratio problems. The maximum density subgraph problem is to identify a subset of nodes in the graph that maximizes the density, defined as the ratio of the weights of the edges with both endpoints in the subset, divided by the sum of weights of the nodes in the subgraph. The densest subgraph has played a central role in analyzing network structures since the 1970's. The more recent applications of the problem are in the context of very large scale networks, such as identifying emerging cyber-communities (Kumar et al., 1999), DNA motif finding (Fratkin et al., 2006), and real-time story identification (Angel et al., 2014).

The maximum density problem was studied since the late 70's. (Picard and Queyranne, 1982) are likely the first to study the problem and recognize its link to the max-flow min-cut problem. Their method was based on a general "linearization" approach that applies for any ratio optimization problem, reducing it to the λ-*question*, defined next, which they proposed to solve with a min-cut procedure on a related graph.

A general ratio problem $\max_{\mathbf{x}\in\mathcal{F}} \frac{f(\mathbf{x})}{g(\mathbf{x})}$ can be reduced to a sequence of calls to an oracle that provides a yes/no answer to the λ-*question*:
Is there a feasible solution $\mathbf{x} \in \mathcal{F}$ such that $\frac{f(\mathbf{x})}{g(\mathbf{x})} > \lambda$?
Or equivalently "Is there a feasible solution $\mathbf{x} \in \mathcal{F}$ such that $f(\mathbf{x}) - \lambda g(\mathbf{x}) > 0$?"
To answer this λ-question it is sufficient to solve:

$$(\lambda\text{-problem}) \quad \max_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x}) - \lambda g(\mathbf{x}).$$

If the maximum value is greater than 0 then there is a feasible solution of ratio value strictly greater than λ. Otherwise the answer is no. Specifically, if the maximum value is strictly less than 0, then there is no feasible solution of ratio value great or equal to λ. If the answer is 0 then the respective optimal solution for the λ-question has a ratio value of λ which is the maximum ratio.

Therefore, any ratio problem that has the corresponding λ-problem polynomial time solvable, and the log of the number of possible values of the ratio bounded by a polynomial quantity, is solvable in polynomial time by applying binary search on the value of the parameter λ.

(Picard and Queyranne, 1982) showed that the λ-

[a] https://orcid.org/0000-0002-2498-0512

Hochbaum, D.
Flow Is Best, Fast and Scalable: The Incremental Parametric Cut for Maximum Density and Other Ratio Subgraph Problems.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 275-282
ISBN: 978-989-758-716-0; ISSN: 2184-3228

problem for MD can be solved as a min-cut (minimum s,t-cut) on a related graph, the construction of which appeared ad-hoc. Their method was essentially a predecessor of our IPC algorithm, showing that the λ-problem for MD would be solved up to n times, where n is the number of nodes in the graph. Here we show a systematic method that maps any optimization (and ratio) problem that is a *monotone integer* program to an associated graph and therefore all these problems are solvable in polynomial time, which as proved here, is the complexity of one min-cut procedure.

For the maximum density problem, a follow up paper by (Goldberg, 1984) improved on the algorithm of Picard and Queyranne, by using binary search on the λ-problem making multiple call to a min-cut procedure, up to $\log n$ times for the edge-unweighted node-unweighted problem. A major breakthrough, the *parametric flow* procedure, was introduced in (Gallo et al., 1989), identifying the solutions for *all* values of the parameter λ that correspond to all possible solutions to the λ-problem, and in the complexity of a single min-cut procedure. This parametric procedure used the push-relabel algorithm of (Goldberg and Tarjan, 1988). Later (Hochbaum, 1998; Hochbaum, 2008) showed a parametric cut procedure using HPF (Hochbaum PseudoFlow) also with the complexity of a single min-cut. We will refer to this parametric procedure also as fully parametric, to differentiate it from “simple” parametric, reviewed in Section 2.2.

Despite its theoretical efficiency, the parametric flow procedure has never been used to solve the densest subgraph problem, to the best of our knowledge. One contributing factor for the lack of use is that there is no implementation available for the parametric push-relabel version proposed by (Gallo et al., 1989). (However, for HPF there is a parametric flow/cut implementation publicly available, (Hochbaum, 2020a).) Instead, flow algorithms have been employed using multiple calls to min-cut in a binary search process, resulting in high running times. This perceived inefficiency gave rise to current state-of-the-art algorithms for the maximum density problem that are based on greedy heuristics that do not guarantee optimality, (Charikar, 2000), (Boob et al., 2020), (Harb et al., 2022). A recent justification for not using the polynomial time flow algorithms is that “flow computations are expensive” (Boob et al., 2020).

Our main contribution here is a new polynomial time algorithm, the *incremental parametric cut* (IPC) algorithm, that solves optimally and efficiently the densest subgraph problem and many other minimum or maximum ratio problems. We also provide an easy characterization of the ratio problems that are solvable with this procedure, as those that can be formulated as monotone integer programming problems. For those problems we describe the respective s,t-graph construction that follows from the formulation.

In a separate experimental study (Hochbaum et al., 2024) we show that the number of breakpoints IPC generates is in the range of $2-13$ even for datasets on millions of nodes and hundreds of million edges, which is typically less than 1% of the total number of breakpoints. This results in very fast running times that are orders of magnitude faster than those of the parametric flow procedure and recent state-of-the-art heuristics that do not produce optimal solutions.

To summarize, the main contributions here are:
1. The incremental parametric cut algorithm IPC that solves “monotone” ratio optimization problems in the complexity of a single min-cut.
2. A new, previously unknown, formulation of densest subgraph problem and its generalizations, that uses half of the number of arcs as compared to the known formulation.
3. An easy characterization of all ratio problems that are solved by IPC. Examples are given in Table 1.

1.1 Ratio Problems Solved with IPC

Notation. We consider the graph representation of the problems, firstly for undirected graphs corresponding to symmetric problems. Let $G=(V,E)$ denote an undirected graph with n denoting the number of nodes in V, and m denoting the number of edges in E. Every edge $[i,j]\in E$ has an associated weight $w_{ij}\geq 0$. Let the *weighted degree* of node $i\in V$ be $d_i=\sum_{[i,j]\in E} w_{ij}$. For $B_1,B_2\subseteq V$, let $C(B_1,B_2)=\sum_{\substack{[i,j]\in E,\\ i\in B_1,\, j\in B_2}} w_{ij}$ be the sum of weights of the edges between nodes in the set B_1 and those in set B_2. Let q_i denote a nonnegative cost value associated with each node, and u_i, or u_i' denote two types of values associated with each node, which could be positive or negative. Let the *degree volume* of a set of nodes S be $d(S)=\sum_{i\in S} d_i$, $q(S)=\sum_{i\in S} q_i$ and $U(S)=\sum_{i\in S} u_i$.

Some ratio problems are defined on directed graphs, $G=(V,A)$, where each arc $(i,j)\in A$ has an associated weight $w_{ij}\geq 0$. The weighted *outdegree* of a node i is $d_i^+=\sum_{j|(i,j)\in A} w_{ij}$, and the outdegree volume of a set of nodes S is $d^+(S)=\sum_{i\in S} d_i^+$.

A sample list of some of the ratio problems solved here is given in Table 1. The *Max density* problem is defined with weighted edges but unit weight on the nodes. This name refers more often to the special case of the unweighted problem where both edges weights are 1 and node weights are 1.

Many ratio problems appear in contexts where the size of optimal set is bounded. For example, the expansion ratio of a graph problem is $\min_{|S|\leq \frac{n}{2}} \frac{C(S,\bar{S})}{|S|}$. This added *size restriction* turns the problem NP-hard. The Cheeger constant problem is typically presented as $\min_{S\subset V} \frac{C(S,\bar{S})}{\min\{d(S),d(\bar{S})\}}$, which is equivalent to the size restricted ratio problem $\min_{d(S)\leq \frac{1}{2}d(V)} \frac{C(S,\bar{S})}{d(S)}$. The conductance problem is $\min_{\pi(S)\leq \frac{1}{2}\pi(V)} \frac{C(S,\bar{S})}{\pi(S)}$ where π_i is interpreted as the stationary probability of node i. We add here the $*$ to the name of the problem to indicate that there is no size restriction, and then the problem is polynomial time solvable. For the minimization problems, the entire set of nodes V is often the optimal solution of value 0. To avoid that trivial solution, the problem is typically solved on a subgraph of nodes V_1. For example Metis (Karypis and Kumar, 1998) has been used to identify a subgraph which is likely to contain the optimal solution for these problems and then the minimization is subject to $\emptyset \subset S \subseteq V_1$.

Table 1: A list of some of the ratio problems solved with the incremental parametric cut. *No size restriction.

Problem name	Objective
Max density	$\max_{S\subseteq V} \frac{C(S,S)}{\lvert S\rvert}$
Weighted max density	$\max_{S\subseteq V} \frac{C(S,S)}{q(S)}$
Ratio quadratic Knapsack	$\max_{S\subseteq V} \frac{C(S,S)+U(S)}{q(S)}$
HNC	$\max_{\emptyset\subset S\subset V} \frac{C(S,S)}{C(S,\bar{S})}$
HNC-equivalent	$\max_{\emptyset\subset S\subset V} \frac{d(S)}{C(S,\bar{S})}$
Max HNC-extension	$\max_{\emptyset\subset S\subset V} \frac{U(S)}{C(S,\bar{S})+U'(S)}$
Expansion ratio*	$\min_{\emptyset\subset S\subset V} \frac{C(S,\bar{S})}{\lvert S\rvert}$
Cheeger*/HNC	$\min_{\emptyset\subset S\subset V} \frac{C(S,\bar{S})}{d(S)}$
Conductance*	$\min_{\emptyset\subset S\subset V} \frac{C(S,\bar{S})}{q(S)}$

The problem HNC (Hochbaum Normalize Cut), also named NC' or SNC, was presented in (Sharon et al., 2006) as an NP-hard problem identical to the Normalized Cut (Shi and Malik, 2000), but shown polynomial time solvable in (Hochbaum, 2010). The same mistake was repeated in (Fortunato, 2010), who stated that Cheeger*/HNC, equation (22), $\min_{\emptyset\subset S\subset V} \frac{C(S,\bar{S})}{d(S)}$, is the normalized cut problem and NP-hard.

2 THEORETICAL BACKGROUND

2.1 Characterization of Polynomial Time Solvability: Monotone Ratio Problems

If the linearized problem can be formulated as monotone integer programming, IPM[1], then it is solvable with a min-cut procedure on an associated s,t graph, where the graph construction is uniquely mapped from the formulation, (Hochbaum, 2002).

IPM problems are classified as *monotone IP2* and *monotone IP3* where IP3 generalizes IP2. An integer program is a monotone IP2 if each constraint contains at most two of the variables that appear with opposite sign coefficients. An integer program is a monotone IP3 if each constraint contains at most two of the variables that appear with opposite sign coefficients and a third variable that appears in that constraint only. (There is an additional requirement that the "third variables" must have nonnegative coefficients in a minimization objective function, and nonpositive coefficients in a maximization objective function.) It is thus easy to recognize whether a formulation is monotone.

The formulation of monotone integer program for a set of n x-variables and a set of constraints involving a collection of pairs of variables A and a respective set of z-variables is,

$$
\begin{array}{lll}
\text{(IPM)} \ \max & \displaystyle\sum_{i=1}^{n} w_i x_i - \sum_{(i,j)\in A} e_{ij} z_{ij} & \\
\text{s.t.} & a_{ij}x_i - b_{ij}x_j \leq c_{ij} + z_{ij} & \forall\ (i,j) \in A \\
 & \ell_i \leq x_i \leq u_i, \ \text{integer} & \forall\ i \in V \\
 & z_{ij} \geq 0, \ \text{integer} & \forall\ (i,j) \in A.
\end{array}
$$

Here there is a restriction that the coefficients of e_{ij} in the objective function are nonnegative for maximization and non-positive for minimization.

Any IPM problem is equivalent to the following binary *s-excess* problem which is formulated on the variables $x_i = 1$ iff node i is in the optimal set S:

$$
\begin{array}{lll}
(s\text{-excess}) & \max & \sum_{j\in V} w_i x_i - \sum_{(i,j)\in A} u_{ij} z_{ij} \\
\text{subject to} & & x_i - x_j \leq z_{ij} \quad \text{for } (i,j) \in A \\
 & & x_j \quad \text{binary } j = 1,\ldots,n \\
 & & z_{ij} \quad \text{binary } (i,j) \in A.
\end{array}
$$

The respective graph G_{st} is constructed as follows, (Hochbaum, 2002): We add nodes s and t to the graph G, with an arc from s to every positive weight node i,

[1] We use the acronym IPM rather than MIP so as not to confuse it with Mixed Integer Programming

of capacity $u_{si} = w_i$, and an arc from every negative weight node j to t of capacity $u_{jt} = -w_j$. Let this added set of arcs, adjacent to s and t (source node and sink node respectively) be denoted by A_{st}. The arcs of A each carry the capacity u_{ij} which is infinite if the constraint has only two variables. The graph G_{st} is then $(V \cup \{s,t\}, A \cup A_{st})$. The proof of the following lemma is given in (Hochbaum, 2002) and omitted here.

Lemma 1. *S^* is a set of maximum s-excess capacity in the original graph G if and only if S^* is the source set of a minimum s,t-cut in the associated graph G_{st}.*

We say that a ratio problem is a monotone integer program (IPM), if the corresponding λ-problem is IPM. For the λ-problem, the corresponding flow graph G_λ has arc capacities that are functions of the parameter λ. An s,t-graph is said to be a *parametric flow graph* if it has source-adjacent capacities that are monotone non-increasing with the parameter λ and the sink-adjacent capacities that are monotone non-decreasing with λ (or vice versa). For a λ-problem represented as a parametric flow graph, G_λ, the parametric cut procedure solves the λ-problem, for all values of the parameter. This is the case for all the problems listed in Table 1 and many more.

2.2 Parametric Cut, Nestedness and the "Continue" Property

Let the minimum cut for graph G_λ be $(S_\lambda, \bar{S}_\lambda)$ with S_λ the "source set" of the minimum cut and $\bar{S}_\lambda$ the "sink set". A property of the parametric flow graph is that as the values of λ are increasing, the source sets of the minimum cuts can only decrease, each a subset of the previous. Formally, for a monotone increasing sequence of p λ values, $\lambda_1 < \lambda_2 < \ldots < \lambda_p$, the corresponding optimal solutions, the source sets of the minimum cuts in the graph G_λ, satisfy $S_{\lambda_1} \supseteq S_{\lambda_2} \supseteq \ldots \supseteq S_{\lambda_p}$, and the respective sink sets satisfy $\emptyset = \bar{S}_{\lambda_0} \subseteq \bar{S}_{\lambda_1} \subseteq \ldots \subseteq \bar{S}_{\lambda_p}$. This property is called *nestedness* and is proved as a corollary of the parametric flow algorithms of (Gallo et al., 1989; Hochbaum, 1998; Hochbaum, 2008). As the value of the parameter λ increases, the respective cut solutions change when the sink set strictly increases. The values of the parameter where the change occurs are called *breakpoints*. Because of the nestedness the solution set changes by adding at least one node to the sink set, and therefore there are at most n breakpoints. For ℓ breakpoints, $\lambda'_1 < \lambda'_2 < \ldots < \lambda'_\ell$, the respective sink sets are strict subsets of each other: $\bar{S}_{\lambda'_1} \subset \bar{S}_{\lambda'_1} \subset \ldots \subset \bar{S}_{\lambda'_\ell}$.

There are two variants of the parametric cut procedure. The *fully parametric* variant generates all the breakpoints (see (Hochbaum, 2020a)). The *simple parametric* variant takes as input a sequence of values of λ, or a sequence of source adjacent capacities and sink adjacent capacities that are monotone non-increasing on one side, and monotone non-decreasing on the other, (Hochbaum, 2020b), and outputs the minimum cut solution for each of them. A property required of a min-cut max-flow algorithm in order for either the fully or simple parametric cut to work in the complexity of a single min-cut procedure, $T(n,m)$, is the *continue* property: Once an optimal solution has been found for one setting of the capacities, it is used as the initial solution for the new problem with updated, monotone, capacities. This is done while maintaining the labels and the invariant structure of the algorithm, which for HPF is called *normalized tree*. To-date, only push-relabel and HPF are max-flow min-cut algorithms that have the continue property. For HPF the routine **HPF-para-continue**(λ, S) is the part that takes a solution, which is the subset S in the related graph, and updated capacities corresponding to λ to find the optimal solution for the updated problem which is a subset of S.

The *continue* operation for HPF using monotonicity is referred to as **HPF-para-continue** and takes as input the solution source set for the value of λ previously used, that is guaranteed to contain the optimal ratio solution (because of nestedness, and the new value λ).

2.3 The Concave Envelope of the Breakpoints

For a general maximum ratio problem $\max_{\mathbf{x}\in\mathcal{F}} \frac{f(\mathbf{x})}{g(\mathbf{x})}$, we consider the graph that maps any value of $g(\mathbf{x}) = B$, so-called "budget", to the maximum value of $f(\mathbf{x}_B) = \arg\max_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x}) | g(\mathbf{x}) \leq B$, referred to as the "benefit". Finding those maximum benefits is in general NP-hard.

Consider the lower envelope of all the lines that have the entire collection of optimal solutions below them. This envelope, shown in red line segments in Figure 1, is concave piecewise linear and the points at which the line segment changes, are called *breakpoints* (marked by boxes in Figure 1).

The ratio value corresponding to each optimal point is the slope of the line connecting it to the origin. Hence the first, leftmost, breakpoint is also the optimal solution to the maximum ratio problem.

The properties of the concave envelope were studied, in the context of the dynamic evolution problem, in (Hochbaum, 2009). These properties include:

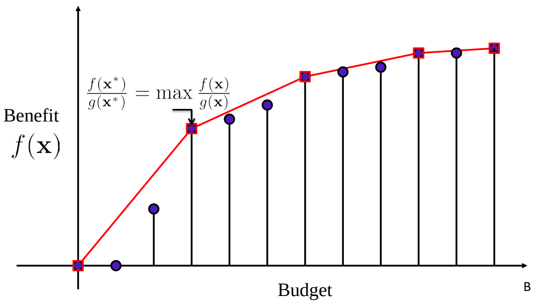


Figure 1: The concave envelope, the breakpoints and the ratio maximizing solution.

- The concave envelope and breakpoints are found with fully parametric cut procedure, (Hochbaum, 2020a).
- The breakpoints of the envelope correspond to the breakpoints of the respective parametric cut solutions, and the left derivative at the ith breakpoint is equal to the ith parameter breakpoint value λ'_i.
- At the breakpoints of the envelope the solutions are optimal.
- The first breakpoint – the smallest positive budget breakpoint – corresponds to the solution which attains the largest ratio of the benefit to the budget.
- The breakpoints correspond to solutions that are *nested* and their number is at most n, the number of variables, or nodes, in the respective graph.

For the respective minimization problems the envelope of the breakpoints is *convex*, and the first breakpoint corresponds to the solution that attains the smallest ratio of benefit to budget, see e.g. Figure 5.

2.4 Incremental Parametric Cut Procedure

Consider the general ratio maximization problem $\max_{\mathbf{x}\in\mathcal{F}} \frac{f(\mathbf{x})}{g(\mathbf{x})}$ where any feasible vector $\mathbf{x}'$ is associated with a subset of nodes in the associated graph, $S' = \{i \in V | x'_i = 1\}$.

The procedure starts with a set of nodes S^0 that is to contain the optimal ratio solution, which for the maximum density problem can be the entire graph, $S^0 = V$. The initial value of the parameter is $\lambda_0 = \frac{f(S^{)})}{g(S^0)}$. Solving the λ_0-problem either provides a solution with strictly higher ratio value, that is also a breakpoint solution, or else its value is 0 and therefore it is the maximum ratio solution. Because of the nested property, each subsequent solution set is strictly contained in the previous iteration's solution set. The value of the ratio is then updated and used as λ in the next iteration. Let S^0 be an initial feasible solution.

```
PROCEDURE INCREMENTAL PARAMETRIC
$(f(), g(), S^0 \subseteq \mathcal{F}$,k=0$)$.
Step 1: $\lambda_k = \frac{f(S^k)}{g(S^k)}$.
Step 2: HPF-para-continue$(\lambda_k, S^k)$ to solve
    $improve(\lambda_k) = \max_{S\subseteq\mathcal{F}\cap S^k} f(S) - \lambda_k g(S)$.
    Let $S^{k+1} = \arg\max_{S\subseteq\mathcal{F}\cap S^k} f(S) - \lambda_k g(S)$.
Step 3: If $\{improve(\lambda_k) > 0\}$ let $k := k+1$. Go to step 1,
    else stop. Output $S^* = S^k$.
```

We now prove the correctness of the procedure in that it visits a sequence of budget-decreasing breakpoints.

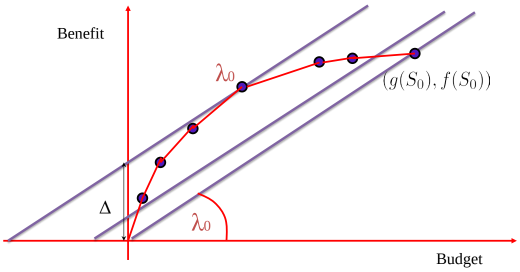


Figure 2: Identifying a breakpoint with $\lambda_0 = \frac{f(S_0)}{g(S_0)}$ subgradient, skipping over several breakpoints.

Lemma 2. *The optimal solution to* $\max_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x}) - \lambda_0 g(\mathbf{x})$, $\mathbf{x}^1$, *is either a breakpoint on the concave envelope at a budget* $< g(\mathbf{x}^0)$ *and with strictly larger ratio than that of* $\mathbf{x}^0$, *or* $\mathbf{x}^1 = \mathbf{x}^0$ *and it is the maximum ratio solution.*

Proof. Consider the line equation $f(\mathbf{x}) = \lambda_0 g(\mathbf{x}) + \Delta$ where Δ the intercept of the line, of slope λ_0, on the vertical axis, as in Figure 2. Maximizing Δ is equivalent to $\max_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x}) - \lambda_0 g(\mathbf{x}) = \Delta^*$. Therefore the line $f(\mathbf{x}) = \lambda_0 g(\mathbf{x}) + \Delta^*$ lies above all feasible solutions and is tangent to the concave envelope at breakpoint $\mathbf{x}^1$, where $\mathbf{x}^1 = \arg\max_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x}) - \lambda_0 g(\mathbf{x})$. $\mathbf{x}^1$ is a breakpoint with a left subgradient equal to λ_ℓ and right subgradient equal to λ_r, such that $\lambda_\ell \geq \lambda_0 \geq \lambda_r$. □

The complexity of the incremental parametric cut procedure is that of a single min-cut HPF procedure on the graph, $T(n,m)$. More precisely, the complexity is $T(n,m) + O(qn)$ where q is the number of breakpoints visited. [2] As noted in the introduction, this number is very small in practice.

[2] (Hochbaum, 2023) mistakenly stated that such a procedure visits adjacent breakpoints.

3 THE METHOD FOR WEIGHTED MAX DENSITY

Let the weighted maximum density problem, WMD, be given on a graph $G = (V,E)$ with positive edge weights u_{ij} and node weights q_i, $\max_{S\subseteq V} \frac{C(S,S)}{q(S)}$. The standard integer programming formulation of the problem has binary variables for each node $i \in V$: $x_i = 1$ if node i is selected in S and 0 otherwise, and binary variables for each edge $[i,j] \in E$, $y_{ij} = 1$ if both i and j are in S, and 0 otherwise. With this notation the formulation of WMD is,

$$\begin{array}{lll} \text{(WMD)} & \max & \frac{\sum_{[i,j]\in E} u_{ij} y_{ij}}{\sum_{j\in V} q_i x_i} \\ \text{subject to} & x_i \leq y_{ij} & \text{for } [i,j] \in E \\ & x_j \leq y_{ij} & \text{for } [i,j] \in E \\ & x_j & \text{binary } j \in V \\ & y_{ij} & \text{binary } [i,j] \in E \end{array}$$

The graph corresponding to this IP2 formulation has $m+n$ nodes, one for each variable. We next present a general procedure for generating an equivalent compact (monotone) formulation for WMD and other ratio problems. Let d_i denote the weighted degree of node i in G: $d_i = \sum_{j|[i,j]\in E} u_{ij}$, and $d(S) = \sum_{i\in S} d_i$. It is easy to see that for any non-empty subset of nodes $S \subset V$, we have the identity $d(S) = 2C(S,S) + C(S,\bar{S})$. Therefore, $C(S,S) = \frac{1}{2}(d(S) - C(S,\bar{S}))$.

Hence, $\max_{S\subseteq V} \frac{C(S,S)}{q(S)} = \frac{1}{2} \max_{S\subseteq V} \frac{d(S)-C(S,\bar{S})}{q(S)}$ which is formulated as monotone integer program as well, with up to 3 variables per inequality using the same x-variables as in WMD, and "cut" variables z_{ij} that are equal to 1 if $i \in S$ and $j \in \bar{S}$ and zero otherwise:

$$\begin{array}{lll} \text{(WMD-compact)} \max & \frac{\sum_{j\in V} d_i x_i - \sum_{[i,j]\in E} u_{ij} z_{ij}}{\sum_{j\in V} 2q_i x_i} & \\ \text{subject to} & x_i - x_j \leq z_{ij} & \text{for } [i,j] \in E \\ & x_j - x_i \leq z_{ji} & \text{for } [i,j] \in E \\ & x_j & \text{binary } j \in V \\ & z_{ij}, z_{ji} & \text{binary } [i,j] \in E. \end{array}$$

The graph associated with the linearized problem, λ-WMD-compact, has one node for each x_i variable and two arcs for each edge in E resulting in a compact formulation on $n+2$ nodes and $2m+2n$ arcs.

Improved Formulation and Smaller Associated Graph. For WMD as well as for any ratio problem that includes only $C(S,S)$ along with linear terms, there is an even more efficient formulation that includes only one z_{ij} variable for every pair that has positive utility, instead of two. This results in a graph with $n+2$ nodes and $m+2n$ arcs which is about half of the number of arcs as compared to the formulation above.

The key is to observe that the problem can be represented on a directed graph $G = (V,A)$ where for each pair i and j with positive utility and $i < j$ there is one arc $(i,j) \in A$ from i to j.

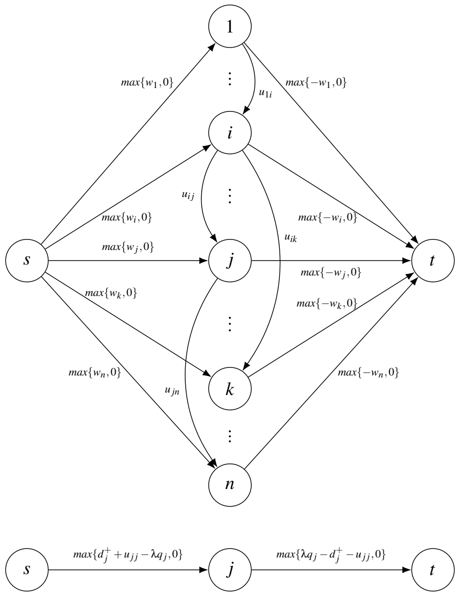


Figure 3: The flow graph G_λ for λ-WMD-compact1.

Let d_i^+ be the weighted out-degree of node i in G: $d_i^+ = \sum_{j|(i,j)\in A} u_{ij}$. Then, for any subset of nodes $S \subset V$, $d^+(S) = C(S,S) + C(S,\bar{S})$. Therefore (WMD-compact1) is an IPM formulation of WMD:

$$\begin{array}{lll} \text{(WMD-compact1)} & \max \frac{\sum_{j\in V} d_i^+ x_i - \sum_{(i,j)\in A} u_{ij} z_{ij}}{\sum_{j\in V} q_i x_i} & \\ \text{subject to} & x_i - x_j \leq z_{ij} & \text{for } (i,j) \in A \\ & x_j & \text{binary } \forall j \in V \\ & z_{ij} & \text{binary } \forall (i,j) \in A. \end{array}$$

The objective function of the linearized ratio problem for the λ-question of (WMD-compact1) is, (λ-WMD) max $\sum_{j\in V} d_i^+ x_i - \sum_{(i,j)\in A} u_{ij} z_{ij} - \lambda \sum_{j\in V} q_i x_i$. The associated graph for this λ-WMD is given in Figure 3 which is obviously a parametric flow graph.

We conclude with an example of finding the densest subgraph with IPC, reported in (Hochbaum et al., 2024), in the dataset COM-YOUTUBE with $n = 1134890$ $m = 2987624$ from (Leskovec and Krevl, 2014). The running time of IPC for this dataset is 1.892 sec. The concave envelope of the breakpoints is shown in Figure 4.

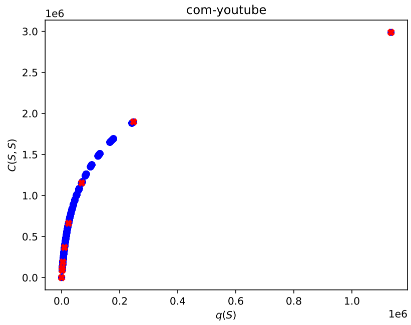


Figure 4: The concave envelope of all 1253 breakpoints, in blue, versus 9 breakpoints explored by IPC, in red. (Courtesy: A. Irribarra-Cortés).

4 APPLICATIONS OF IPC

We consider here the three ratio problems: HNC $\max_{\emptyset\subset S\subset V}\frac{C(S,S)}{C(S,\bar{S})}$, Cheeger's* $\min_{\emptyset\subset S\subset V}\frac{C(S,\bar{S})}{d(S)}$ and conductance*/HNC-extension $\min_{\emptyset\subset S\subset V}\frac{C(S,\bar{S})}{q(S)}$.

We first show, directly from the problem statement, that HNC is an IPM ratio problem. Then provide a transformation showing that HNC is equivalent to Cheeger*'s, and obviously conductance is a slight generalization of both. We then give the formulation for all three problems that leads to the parametric flow graph that is solved with IPC.

We first comment on the use of the constraint $\emptyset\subset S\subset V$ in ratio problems involving the cut $C(S,\bar{S})$. For such problems, unlike WMD, if unrestricted the solution will be the entire graph with cut value 0. In general that means that to solve such problems it is necessary to use *seeds* which are subsets of nodes so at least one belongs to the sink set and at least one belongs to the source set. For these problems, when they have size constraint, such as for Cheeger's, of the form $d(S)\leq\frac{d(V)}{2}$, the problems are NP-hard. To address the issue of the seeds and to solve the size restricted ratio problems heuristically one can choose to first identify a subset of the graph where the optimal subgraph may reside. This was done for example using the Metis graph partitioning heuristic of (Karypis and Kumar, 1998) by (Lang and Rao, 2004). Once the subgraph satisfying the size restriction is found, say V', the problem becomes $\min_{\emptyset\subset S\subset V'}\frac{C(S,\bar{S})}{d(S)}$.

Consider the integer programming formulation of HNC $\max_{\emptyset\subset S\subset V}\frac{C(S,S)}{C(S,\bar{S})}$ with edge weights w_{ij} and binary variables x_i, y_{ij} and z_{ij}. Let $x_i=1$ if $i\in S$, $y_{ij}=1$ if both i and j in S and $z_{ij}=1$ if $i\in S$ $j\in\bar{S}$. The following is the linearized formulation λ-HNC:

$$\begin{array}{lll}
(\lambda\text{-HNC}) & \max & \sum_{[i,j]\in E} w_{ij}y_{ij}-\lambda\sum_{j\in V} w_{ij}z_{ij} \\
\text{subject to} & & x_i\leq y_{ij} \quad \text{for } [i,j]\in E \\
 & & x_j\leq y_{ij} \quad \text{for } [i,j]\in E \\
 & & x_i-x_j\leq z_{ij} \quad \text{for } [i,j]\in E \\
 & & x_j-x_i\leq z_{ji} \quad \text{for } [i,j]\in E \\
 & & x_j \quad \text{binary } j\in V \\
 & & z_{ij},z_{ji},y_{ij} \quad \text{binary } [i,j]\in E
\end{array}$$

This monotone integer program maps into an associated graph on $m+n+2$ nodes and $2m+2n$ arcs. A compact formulation of HNC, equivalent to Cheeger's*, is given in the next lemma (proof omitted for lack of space):

Lemma 3. *The following two problems are equivalent and have the same optimal solutions:* $\max_{\emptyset\subset S\subset V}\frac{C(S,S)}{C(S,\bar{S})}$, *and* $\min_{\emptyset\subset S\subset V}\frac{C(S,\bar{S})}{d(S)}$.

Therefore solving HNC-extension, or conductance*, provides solutions to all three problems since setting $q_i=d_i$ is HNC or Cheeger's* problem. The problem $\min_{\emptyset\subset S\subset V}C(S,\bar{S})-\lambda q(S)$ is formulated as follows.

$$\begin{array}{lll}
(\lambda\text{-HNC-extension}) & \min & \sum_{[i,j]\in E} u_{ij}z_{ij}-\lambda\sum_{j\in V} q_ix_i \\
\text{subject to} & & x_i-x_j\leq z_{ij} \quad \text{for } [i,j]\in E \\
 & & x_j-x_i\leq z_{ji} \quad \text{for } [i,j]\in E \\
 & & x_j \quad \text{binary } j\in V \\
 & & z_{ij},z_{ji} \quad \text{binary } [i,j]\in E.
\end{array}$$

The graph associated with this monotone integer program has $n+2$ nodes and $2m+2n$ arcs which improves on the number of nodes $m+n+2$ in the λ-HNC formulation.

To conclude we provide an example of solving Cheeger's* on a subgraph V' delivered by the Metis procedure, $\min_{\emptyset\subset S\subset V'}\frac{C(S,\bar{S})}{d(S)}$ applied to the dataset EGO-GPLUS of size $n=107614$ $m=12238285$, from (Leskovec and Krevl, 2014) (reported in (Hochbaum et al., 2024)). The convex envelope shown in Figure 5 illustrates the difference between the set of all breakpoints, generated with the fully parametric cut procedure, versus the set of points explored by IPC.

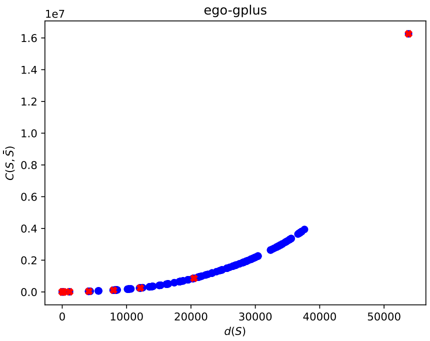


Figure 5: The convex envelope of all 291 breakpoints, in blue, versus 11 breakpoints explored by IPC, in red. (Courtesy: A. Irribarra-Cortés).

ACKNOWLEDGEMENTS

This research was supported in part by the AI4OPT institute NSF award 2112533.

REFERENCES

Angel, A., Koudas, N., Sarkas, N., Srivastava, D., Svendsen, M., and Tirthapura, S. (2014). Dense subgraph maintenance under streaming edge weight updates for real-time story identification. *The VLDB journal*, 23:175–199.

Boob, D., Gao, Y., Peng, R., Sawlani, S., Tsourakakis, C., Wang, D., and Wang, J. (2020). Flowless: Extracting densest subgraphs without flow computations. In *Proceedings of The Web Conference 2020*, pages 573–583.

Charikar, M. (2000). Greedy approximation algorithms for finding dense components in a graph. In *International workshop on approximation algorithms for combinatorial optimization*, pages 84–95. Springer.

Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3-5):75–174.

Fratkin, E., Naughton, B. T., Brutlag, D. L., and Batzoglou, S. (2006). Motifcut: regulatory motifs finding with maximum density subgraphs. *Bioinformatics*, 22(14):e150–e157.

Gallo, G., Grigoriadis, M. D., and Tarjan, R. E. (1989). A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55.

Goldberg, A. V. (1984). Finding a maximum density subgraph. *UC Berkeley manuscript*.

Goldberg, A. V. and Tarjan, R. E. (1988). A new approach to the maximum-flow problem. *Journal of the ACM (JACM)*, 35(4):921–940.

Harb, E., Quanrud, K., and Chekuri, C. (2022). Faster and scalable algorithms for densest subgraph and decomposition. *Advances in Neural Information Processing Systems*, 35:26966–26979.

Hochbaum, D. S. (1998). The pseudoflow algorithm and the pseudoflow-based simplex for the maximum flow problem. In *Integer Programming and Combinatorial Optimization: 6th International IPCO Conference Houston, Texas, June 22–24, 1998 Proceedings 6*, pages 325–337. Springer.

Hochbaum, D. S. (2002). Solving integer programs over monotone inequalities in three variables: A framework for half integrality and good approximations. *European Journal of Operational Research*, 140(2):291–321.

Hochbaum, D. S. (2008). The pseudoflow algorithm: A new algorithm for the maximum-flow problem. *Operations research*, 56(4):992–1009.

Hochbaum, D. S. (2009). Dynamic evolution of economically preferred facilities. *European Journal of Operational Research*, 193(3):649–659.

Hochbaum, D. S. (2010). Polynomial time algorithms for ratio regions and a variant of normalized cut. *IEEE transactions on pattern analysis and machine intelligence*, 32(5):889–898.

Hochbaum, D. S. (2020a). Hpf - hochbaum's pseudoflow. Accessed: May 28, 2022, https://riot.ieor.berkeley.edu/Applications/full-para-HPF/pseudoflow-parametric-cut.html.

Hochbaum, D. S. (2020b). Pseudoflow (simple) parametric maximum flow solver version 1.0. Accessed: May 28, 2022, https://riot.ieor.berkeley.edu/Applications/Pseudoflow/parametric.html.

Hochbaum, D. S. (2023). Unified new techniques for np-hard budgeted problems with applications in team collaboration, pattern recognition, document summarization, community detection and imaging. *Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, 1:365–372.

Hochbaum, D. S., Irribarra-Cortés, A., and Asín-Achá, R. (2024). Fast and optimal incremental parametric procedure for the densest subgraph problem: An experimental study. *UC Berkeley manuscript*.

Karypis, G. and Kumar, V. (1998). A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392.

Kumar, R., Raghavan, P., Rajagopalan, S., and Tomkins, A. (1999). Trawling the web for emerging cyber-communities. *Computer networks*, 31(11-16):1481–1493.

Lang, K. and Rao, S. (2004). A flow-based method for improving the expansion or conductance of graph cuts. In *Integer Programming and Combinatorial Optimization: 10th International IPCO Conference, New York, NY, USA, June 7-11, 2004. Proceedings 10*, pages 325–337. Springer.

Leskovec, J. and Krevl, A. (2014). SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.

Picard, J.-C. and Queyranne, M. (1982). A network flow solution to some nonlinear 0-1 programming problems, with applications to graph theory. *Networks*, 12(2):141–159.

Sharon, E., Galun, M., Sharon, D., Basri, R., and Brandt, A. (2006). Hierarchy and adaptivity in segmenting visual scenes. *Nature*, 442(7104):810–813.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

Federated Learning for XSS Detection: A Privacy-Preserving Approach

Mahran Jazi[a] and Irad Ben-Gal[b]
Department of Industrial Engineering, Tel Aviv University, Tel Aviv, Israel
mohranjazi@mail.tau.ac.il, bengal@tauex.tau.ac.il

Keywords: Federated Learning, Cross-Site Scripting (XSS) Detection, On-Device Learning, Non-IID Data Distribution, Threat Detection in Web Applications.

Abstract: Collaboration between edge devices has increased the scale of machine learning (ML), which can be attributed to increased access to large volumes of data. Nevertheless, traditional ML models face significant hurdles in securing sensitive information due to rising concerns about data privacy. As a result, federated learning (FL) has emerged as another way to enable devices to learn from each other without exposing user's data. This paper suggests that FL can be used as a validation mechanism for finding and blocking malicious attacks such as cross-site scripting (XSS). Our contribution lies in demonstrating the practical effectiveness of this approach on a real-world dataset, the details of which are expounded upon herein. Moreover, we conduct comparative performance analysis, pitting our FL approach against traditional centralized parametric ML methods, such as logistic regression (LR), deep neural networks (DNNs), support vector machines (SVMs), and k-nearest neighbors (KNN), thus shedding light on its potential advantages. The dataset employed in our experiments mirrors real-world conditions, facilitating a meaningful assessment of the viability of our approach. Our empirical evaluations reveal that the FL approach not only achieves performance on par with that of centralized ML models but also provides a crucial advantage in terms of preserving the privacy of sensitive data.

1 INTRODUCTION

Today's digital landscape is crowded with edge devices that are multiplying at an alarming rate. As a result, an unimaginably large amount of personal data, complete with different aspects of users' lives, such as multimedia content and text information, is being accumulated. Using this private data to support machine learning (ML) in user applications has become more common. However, the conventional approach of centralizing the ML training process on powerful servers presents a problem. While data originates and applications execute on edge devices, centralized servers amass substantial portions of user data, triggering significant privacy concerns (McMahan et al., 2017).

This centralization creates a fundamental conflict: on the one hand, centralized servers participate in supplying the computational searching ability and storage for training complicated multilayer models; on the other hand, there exist some safety risks for users. The downside of centralization is possible data security threats and abuse of users' data, as well as large communication costs. As a result, available solutions for on-client machine learning have had to be efficient and privacy-preserving to avoid sending data to large repositories. Therefore, recently, the idea of federated learning (FL) has emerged as a possible solution to both concerns (McMahan et al., 2017). FL implements a distributed collaborative learning algorithm, which does not necessitate storing user data in the cloud or on a central server, as illustrated in 1. Under this setup, each client keeps its local private training dataset and never relinquishes control over sensitive information. Instead of transmitting raw data, clients

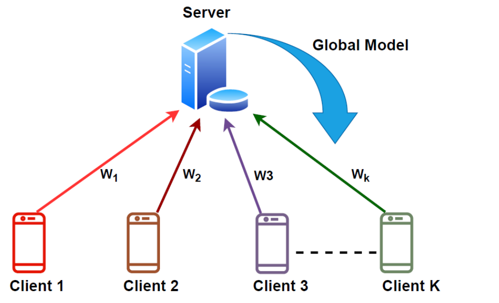


Figure 1: Federated learning scheme.

[a] https://orcid.org/0000-0001-6432-3800
[b] https://orcid.org/0000-0003-2411-5518

Jazi, M. and Ben-Gal, I.
Federated Learning for XSS Detection: A Privacy-Preserving Approach.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 283-293
ISBN: 978-989-758-716-0; ISSN: 2184-3228

only share their local model parameters, like neural network weights, with central servers. These models are often structured similarly and undergo aggregation averaging before being re-distributed to clients.

In addition, nowadays, the world increasingly operates with Internet-based services and web applications (Kotzur, 2022). This overreliance on web-based communication leads to increased website attacks that seek to breach a system's vulnerabilities and corrupt the devices and data, destroying them. One particular threat to note is Cross-Site Scripting (XSS), which remains one of the Open Web Application Security Projects (OWASP) (OWASP, 2017) and has been identified by past studies as a highly persistent issue. Conventional mechanisms for patching security holes in web applications rely on a database of information on known attack signatures(Ariu and Giacinto, 2011). Nevertheless, XSS attacks usually take advantage of vulnerabilities in user input specification or leverage the space between client-side and server-side defenses (Rocha and Souto, 2014; Lee et al., 2022). FL could mitigate web-based communication safety issues in the same way it enhances privacy by decentralized data. Furthermore, novel strategies are needed to mitigate these security challenges by empowering ML. FL can increase the resistance of web applications against XSS attacks, leveraging its principles to preserve user privacy and enhance security.

1.1 Contributions

We propose a novel system that employs FL to detect XSS attacks, addressing the limitations of traditional methods. This innovative approach allows users to leverage shared models while preserving decentralized storage's privacy and scalability benefits. In practical tests, where we employed various ML models within the FL-based system, we effectively demonstrated its capacity to detect and counter XSS attacks. Through a series of experiments, we thoroughly evaluated the effectiveness of diverse ML algorithms in detecting XSS attacks using accuracy metrics. Our assessment included traditional logistic regression (LR) and deep neural network (DNN) algorithms as centralized models, allowing us to juxtapose their performance with that of FL. By comparing the performances of traditional LR and DNN algorithms with that of FL, our goal was to determine the most effective and efficient approach for accurately detecting attacks. Notably, the privacy-preserving nature of FL ensures that clients do not share any private data, addressing the key concerns associated with collaborative learning. Our results are based on the iid and non-iid data distribution settings.

2 BACKGROUND

2.1 Web Applications and JavaScript

Web applications refer to software programs designed to perform specific tasks and are typically requested by a client's web browser over the internet (Ndegwa, 2016). These applications are hosted on remote servers and accessed through web browsers. These tools include two main components: server-side scripts, such as Java Servlets, ASP, and PHP, which manage the processing and retrieval of data from the backend database; and client-side scripts, such as HTML and Java Applets, which are responsible for presenting the information to a user in their web browser.

Examples of typical web applications include email services and e-commerce platforms, which may require server-side processing, as well as applications that do not require any processing on the server. To handle HTTP requests from a client, a web server is necessary, an application server is needed to execute the requested tasks, and a database is used to store information when necessary.

Various security implications arise from poor programming of the web applications (Meyer and Cid, 2008). These bugs can be exploited to gain unauthorized access to a server and its associated databases or steal sensitive data, like credit card information. The term "web application attacks" is used for these types of attacks on web applications.

JavaScript is an object-oriented scripting language commonly employed in designing and implementing dynamic websites. It does not involve server-side processing like other programming languages but relies purely on a client browser to execute the source code. However, JavaScript can also be misused by hackers who want to distribute malicious scripts through different means. For example, they perform Cross-Site Scripting (XSS), Passive Downloads, or SQL injection attacks (Wei-Hong et al., 2013).

2.2 Cross-Site Scripting Attacks

Cross-site scripting (XSS) has become a prevalent way of attacking many websites (Lee et al., 2022). OWASP categorizes such attacks among the top ten in terms of how incapacitating they can be. The purpose of an XSS attack is to place destructive content within a valid web page or application and execute it on the victim's browser. This occurs when a blameless user visits a webpage or web app that includes damaging programming, which then runs on his/her web browser, allowing the attacker to gain unauthorized entry into private information, including cookies/user profiles or installing malware. Commonly used conduits for XSS are message boards, forums, and web pages where users leave comments.

Reflected XSS attacks, stored XSS attacks, and DOM-based XSS attacks are the three major types of these attacks (Galán et al., 2010). The methods each uses to inject attacker codes into applications differ concerning these three attack types and the means through which they affect code execution. Most authors exclude DOM-based XSS attacks when categorizing XSS attacks because these attacks are vulnerable to the script used by web browsers, unlike Reflected and Stored XSS attacks, which exploit vulnerabilities in web applications (Klein, 2005). Reflected and Stored XSS attacks involve injecting script code through an HTTP request. Reflected XSS attacks, also known as nonpersistent XSS attacks, are commonly used to steal sensitive data, such as cookies, by executing malicious scripts on the victims' machines. This type of attack is executed immediately in the victim's browser since the script is included in the HTTP response.

On the other hand, stored XSS attacks are more dangerous because they can affect multiple users who access the infected page. This attack entails directly injecting the script or payload into the target page's database, which allows the attacker to keep running their malicious code.

The third kind of XSS attack is DOM-based XSS, and it is unique in that it does not rely on a vulnerability inherent in a web application itself. Rather, these attacks exploit vulnerabilities within the document object model (DOM) of a web browser to inject malicious code inside targeted pages. The attacker can do this by engineering a malevolent URL that, when clicked, will insert the code straight into the DOM of that page.

Although all kinds of XSS attacks may have different features and execution methods, they all aim to exploit web applications to execute software scripts. Consequently, developers and website administrators must be familiar with this class of attacks and protect their systems against them.

2.3 Data Privacy in XSS Detection and the Role of Federated Learning

A critical aspect of XSS detection involves analyzing the data or scripts embedded in web pages. While the scripts might not always contain sensitive information, they are often associated with user-specific contexts, such as session data, browsing history, or user interactions with web applications. This context can reveal users' private information, such as their browsing habits, preferences, and even personal identifiers.

Traditionally, XSS detection models have relied on centralized servers to aggregate and analyze this data, raising significant privacy concerns. Centralized storage and processing of such data can expose it to potential breaches, misuse, or unauthorized access, compromising user privacy. This is particularly concerning in scenarios involving edge devices, where data is generated and consumed locally, such as in IoT environments.

Federated Learning (FL) presents an innovative solution to this privacy challenge. By allowing devices to collaboratively train models without sharing raw data with a central server, FL ensures that sensitive information remains on the user's device. In XSS detection, each device can contribute to improving the detection model by sharing only model updates rather than the underlying data.

This approach is particularly valuable for protecting data privacy in edge computing environments, where data decentralization is both a necessity and a strength. By applying FL to XSS detection, we can enhance the security and privacy of web applications while still leveraging the collective intelligence of distributed devices.

These privacy considerations motivate the choice of FL for XSS detection. Our work explores this novel application of FL to XSS detection, providing a framework for maintaining high detection accuracy without compromising user privacy. Through comparative analysis with traditional centralized machine learning models, we demonstrate the effectiveness of FL in this context, highlighting its potential to revolutionize the web security field.

3 RELATED WORK

3.1 Cross-Site Scripting Detection

In 2009, Likrash et al. worked on predicting malicious JavaScript code using multiple ML classifiers. These classifiers were used to determine which features of the JavaScript code could help their model determine potentially malicious code (Likarish et al., 2009). The classifiers used in this study were naïve Bayes, ADTree , support vector machine (SVM), and RIPPER classifiers. The authors of this study used a 10-fold cross-validation technique to train and test the models; thus, the data were divided into ten segments: 9 for training and one for testing. However, this process was performed ten times, so each segment was utilized in the training and testing phases at least once

(Likarish et al., 2009). In their work, the authors achieved a precision value ((number of correctly labeled malicious scripts)/(total number of scripts that are marked as malicious)) of 0.92% and a recall rate ((number of correctly labeled malicious scripts)/(total number of malicious scripts)) of 0.787% (Likarish et al., 2009). One concern with this study was that the training set contained 50,000 benign codes and only 62 malicious codes without oversampling the malicious code, which might explain the low recall value.

Komiya et al. (Komiya et al., 2011) used ML techniques, such as SQL injection and XSS attacks, adapted to changes in code characteristics to predict malicious web code. The first stage was the learning process. In this stage, the classifier extracted features from malicious or nonmalicious web code from each training dataset using a feature vector. The vector contained the weight of each feature (term), which was the number of occurrences calculated using the term frequency-inverse document frequency (TF-IDF) method. The second process was the classification process, which used the criteria constructed from the learning process to classify the user input. The authors constructed two separate classifiers, one for XSS attacks and another for SQLIAs(Komiya et al., 2011). The utilized classifiers included an SVM (with a linear kernel), another SVM (with a polynomial kernel), a third SVM (with a Gaussian kernel), naive Bayes, and K-nearest neighbors (KNN). The KNN classifier yielded the highest precision (0.991), and the SVM with a Gaussian kernel yielded the highest accuracy (99.16%). The principal concern regarding this study was that the dataset used for training and testing was relatively small and might not have reflected real-world web attacks(Komiya et al., 2011). Another experiment conducted by Nunan involved XSS attacks (Nunan et al., 2012). By depending on web document content and URLs, the authors aimed to detect malicious pages using ML techniques. Different classification algorithms were used to extract features that helped predict XSS attacks. In this experiment, the authors focused on detecting web page obfuscation by encoding hexadecimal, decimal, octal, Unicode, Base64, and HTML reference characters. The employed classification algorithms included naive Bayes and SVM classifiers. They also performed this experiment using 10-fold cross-validation (Nunan et al., 2012). Wei-Hong et al. worked on detecting malicious scripts by using static analysis techniques to extract features and an SVM to classify scripts (Wei-Hong et al., 2013). The authors extracted features first based on previous work performed by other researchers and second by manually analyzing the data. In their work, utilizing an SVM, they reached an accuracy of 96.59% on the training set and an accuracy of 94.38% on the testing set (Wei-Hong et al., 2013). Other researchers have used ML techniques to distinguish between obfuscated and nonobfuscated scripts (Aebersold et al., 2016). To reach this goal, they used the following classifiers while depending on Azure ML: average perceptron (AP), Bayes point machine (BPM), boosted decision tree (BDT), decision forest (DF), decision jungle (DJ), locally deep SVM (LDSVM), LR, neural network (NN), and SVM classifiers. The authors studied the ability of these classifiers to detect malicious scripts; however, no malicious scripts were included in the training dataset that was used to build the models. The BDT classifier achieved the highest precision (100%), with a recall of 47.71% (Aebersold et al., 2016). Mereani et al. (Mereani and Howe, 2018) aimed to build classifiers to predict a persistent (on-storage) XSS attack in a Java script using ML techniques. Persistent XSS attacks occur when a hacker injects his or her code and saves it in the database of the target web application. Whenever the web application is accessed, the script runs on the user's browser. The authors used three classifiers in their work: an SVM, a KNN, and a random forest. The conventional approach to XSS detection typically involves extracting certain features based on experience and subsequently determining if it constitutes an XSS attack using rule-based matching methods. However, this methodology struggles to identify increasingly intricate XSS attack patterns. With the swift advancements in machine learning, an expanding cohort of researchers has endeavored to address network security issues through machine learning algorithms, with particular emphasis on XSS attack detection, resulting in notable advancements (Yan et al., 2022; Wu et al., 2021a; Wu et al., 2021b; Wu et al., 2021c; Wu et al., 2019). (ZHOU et al., 2019) proposed a model that combines a multilayer perceptron with a hidden Markov model (HMM). (Luo et al., 2020) developed a URL feature representation method by analyzing existing URL attack detection technologies and put forward a multi-source fusion method based on a deep learning model. This approach enhances the overall accuracy and system stability of the XSS detection system.

3.2 Federated Learning

The inception of FL traces its origins to 2017, marked by the unveiling of this innovative paradigm by Google in their seminal paper (McMahan et al., 2017). This pioneering endeavor introduced a decentralized approach, denoted FL, meticulously designed to preserve the privacy of the data belonging to participating clients. In the FL domain, a central server or aggregator assumes a central role in orchestrating a consortium of clients, enabling collaborative data analysis through a shared model. Crucially, FL upholds the principle of data sovereignty, ensuring that each client maintains absolute control over their data, which remains confined within the boundaries of their devices. Within this framework, the model is regarded as communal property shared among the clients, with the sole exchanges encompassing parameter updates. Additionally,(McMahan et al., 2017) proposed a novel decentralized learning technique termed federated averaging that models the performance of a convolutional neural network (CNN) using diverse types of data such as the MNIST, CIFAR-10, and LSTM datasets. Numerous studies have extended the scope of FL. For instance, they may focus on the challenges associated with system heterogeneity and seek to minimize the incurred communication overhead. In their work, Bonawitz et al. (Bonawitz et al., 2017) implemented a secure and efficient FL algorithm with a fixed number of rounds, ensuring low communication overhead and high robustness, especially when handling high-dimensional data received from clients. Addressing uplink costs, Konevcny et al. (Konečný et al., 2016) introduced methods based on structured and sketched updates, showcasing significant communication overhead reductions (by two orders of magnitude).To help with training, they employed techniques such as correcting the momentum and clipping the local gradient to significantly reduce the communication bandwidth overhead in deep gradient compression (DGC) (Lin et al., 2017). Additionally, Hsieh et al. (Hsieh et al., 2020), Li et al. (Li et al., 2020b), and Shamir et al. (Shamir et al., 2014) developed novel distributed learning algorithms by employing multiple minibatches and full-batch stochastic gradient descent (SGD) to alleviate communication overheads and improve overall efficiency of FL. This is believed to be the first time that FL has been used for user privacy protective XSS attack detection (McMahan et al., 2017; Yang et al., 2019; Li et al., 2020a; Zhang et al., 2021; Kairouz et al., 2021).

4 PROBLEM FORMULATION

This section describes our proposed FL scheme, which runs FL on a set of clients. We consider N clients engaged in a classification task, where the goal is to learn a function that maps every input data point to the correct class out of K possible options. Each client n has access to its own private data $\mathcal{D}_n = \{x_i^n\}_{i=1}^{M_n}$ consisting of M_n inputs and their corresponding labels y_i. All the labels are hard-decision vectors formed over the set of all classes. Each client n has a model (e.g., a DNN) with p parameters (e.g., weights): $\omega^n \in \mathbb{R}^p$. We follow conventional FL and assume that all clients have the same architecture for their models so they can be easily averaged.

Let $l(\omega, x, y)$. be the loss incurred on a training data point (x, y). The local training loss function of client n is then

$$\mathcal{L}_n(\omega^n) \triangleq \sum_{x \in \mathcal{D}_n} l(\omega^n, x, y(x)). \tag{1}$$

The goal of the training process in FL is to learn a common model ω that minimizes the total loss induced across all clients, which is defined as follows:

$$\mathcal{L}(\omega) = \sum_{n=1}^{N} \mathcal{L}_n(\omega). \tag{2}$$

The iterations t of the employed FL algorithm consist of two parts. First, the server collects the client models and computes the average model:

$$\overline{\omega}(t) = \frac{1}{N} \sum_{n=1}^{N} \omega^n(t). \tag{3}$$

Then, each client performs a local SGD step to update the average model, with a momentum parameter $0 \leq \beta < 1$:

$$\omega^n(t+1) = \overline{\omega}(t) \eta(t) v(t+1) \tag{4}$$

where $\eta(t)$ is the step size sequence and

$$v(t+1) = \beta v(t) + g_n(\overline{\omega}(t)) \tag{5}$$

Which coincides with the standard SGD strategy for $\beta = 0$. The stochastic gradient $g_n(\overline{\omega}(t))$ is obtained concerning a random subset of data points $\mathcal{S}_n \subset \mathcal{D}_n$ of size B (i.e., the batch size):

$$g_n(\overline{\omega}(t)) = \sum_{x \in \mathcal{S}_n} \nabla l(\overline{\omega}(t), x, y(x)). \tag{6}$$

5 EXPERIMENTAL RESULTS

The experiments were conducted using Google Colab, running the datasets and models within the Python environment. We report the percentage of accurately classified data points in the test dataset (”accuracy”) for the federated learning (FL) model obtained after training.

5.1 Datasets

5.1.1 XSS Dataset

We utilized a balanced XSS dataset comprising scripts from multiple sources. Two datasets containing malicious and benign JavaScript programs were

Table 1: Datasets containing malicious and benign scripts.

Dataset Type	Dataset Source
Malicious Scripts	(Mereani and Howe, 2018)
Benign Scripts	(Mereani and Howe, 2018)

gathered, with the sources listed in Table 1. The dataset consists of 62 attributes and 24,097 data instances.

Selecting the features to be trained by FL models is challenging due to the vast number of available design options. Feature selection is typically divided into two categories.

- **Structural Features.** This refers to the complete set of non-alphanumeric characters, including five additional combinations. For example, a hacker may add unnecessary commands (¡!) or spaces between lines.
- **Behavioral Features.** These are specific commands or functions that may be used in a malicious JavaScript, such as the `(Var)` function. Table 2 outlines both the structural and behavioral features utilized in our experiments.

5.1.2 CICIDS2017 Dataset

The CICIDS2017 dataset (Sharafaldin et al., 2018), created by the Canadian Institute for Cybersecurity (CIC), offers a comprehensive collection of network traffic data representing various cyber threats and normal network behavior. The dataset consists of 85 features with 458,968 data instances. Preprocessing steps included checking for null values, converting categorical objects to numerical values, and normalizing the data to eliminate outliers.

5.2 Model Architectures and Training

We implemented and tested four machine-learning models:

- **Logistic Regression (LR):** A basic linear model used for binary classification tasks. This model served as a baseline for our comparisons.
- **Multilayer Perceptron (MLP or 2NN):** A neural network with two hidden layers containing 200 units and ReLU activation functions as in (McMahan et al., 2017). The architecture is simple yet effective for binary classification tasks involving malicious and benign labels.
- **Support Vector Machine (SVM):** A powerful model used for binary classification, particularly effective in high-dimensional spaces. We used a Radial Basis Function (RBF) kernel, which is a common choice for non-linear classification. The RBF kernel maps the input data into a higher-dimensional space, which makes it easier to classify using a linear decision boundary.
- **k-Nearest Neighbors (KNN):** A simple yet effective non-parametric method for classification tasks. We set k=5 as a default value, balancing bias and variance.

All models were trained using Stochastic Gradient Descent (SGD) where applicable (for LR and MLP) with the following parameters:

- **Learning rate:** $\eta = 0.01$
- **Momentum:** $\beta = 0.9$
- **Batch size:** $B = 32$
- **Local epochs:** $E = 1$ (i.e., SGD operated over the local dataset once)
- **Communication rounds:** 100

The KNN model's classification is non-iterative, so the parameters related to SGD do not apply. Instead, the model computes distances between data points and assigns labels based on the majority class among the nearest neighbors.

Each experiment was run ten times, and the results presented are averages across these runs, with error bars representing one standard deviation.

To simulate a realistic Federated Learning (FL) environment, we divided the XSS and CICIDS2017 datasets into several shards, assigning each shard to a different client and ensuring each client had its private data. We followed an 80:20 train-test split, reserving 20% of the data for evaluating the model's performance on unseen data.

5.3 Reproducibility and Code Availability

We implemented all models using standard libraries in Python. To ensure reproducibility, the models' detailed architectures, preprocessing steps, and training configurations are provided. The code, including data preprocessing scripts, model architectures, and the FL implementation, will be made available in a public repository upon the paper's acceptance. The datasets used are publicly accessible and cited accordingly.

5.4 IID Data Distribution

The results obtained for independent and identically distributed (IID) data are presented in Table 3. In this setting, ten clients were utilized, each with random data points from the specific datasets (XSS and CICIDS 2017); thus, the data distributions among the

Table 2: Behavioral and structural features (Mereani and Howe, 2018).

Features	Description	Type
Readability	The number of alphabetical characters.	Behavioral
Objects	Document, window, I frame, location.	Behavioral
Events	Onload, Onerror.	Behavioral
Methods	createElement, String.fromCharCode.	Behavioral
Tags	DIV, IMG, <script>.	Behavioral
Attributes	SRC, Href, Cookie.	Behavioral
Reserve	Var.	Behavioral
Functions	eval().	Behavioral
Protocol	HTTP.	Behavioral
External File	.js file.	Behavioral
Punctuation	<, #, $, @.	Structural
Combinations	”¿¡”, ”==”.	Structural

Table 3: Performance of FL and Traditional Centralized models in binary classification on XSS and CICIDS2017 datasets based on IID data distribution.

Dataset	Model	Accuracy	Precision	Recall	F1 Score
XSS	FL model (LR)	98.9%	99.9%	97.3%	98.6%
	Traditional Centralized model (LR)	99.9%	99.9%	99.8%	99.9%
	FL model (DNN)	99.9%	99.9%	99.9%	99.9%
	Traditional Centralized model (DNN)	99.9%	99.9%	99.8%	99.9%
	FL model (SVM)	99.9%	99.9%	99.9%	99.9%
	Traditional Centralized model (SVM)	99.96%	99.96%	99.96%	99.96%
	FL model (KNN)	99.7%	99.7%	99.7%	99.7%
	Traditional Centralized model (KNN)	99.90%	99.90%	99.90%	99.90%
CICIDS2017	FL model (LR)	94.01%	89.36%	90.0%	89.86%
	Traditional Centralized model (LR)	97.9%	94.72%	98.33%	96.49%
	FL model (DNN)	98.48%	96.49%	98.33%	97.40%
	Traditional Centralized model (DNN)	99.8%	99.3%	99.9%	99.6%
	FL model (SVM)	98.41%	96.48%	98.09%	97.28%
	Traditional Centralized model (SVM)	99.31%	99.04%	98.57%	98.80%
	FL model (KNN)	96.49%	90.54%	98.09%	94.17%
	Traditional Centralized model (KNN)	99.38%	98.12%	99.76%	98.93%

clients are similar. The experiment aims to identify anomalies in the data. Our results, as an example in Figure 2, confirmed that FL could reach a comparable performance level to traditional centralized models after a few communication rounds while maintaining data privacy and never sharing any sensitive data with the central server. Furthermore, the results for the binary classification of XSS and CICIDS2017 datasets, presented in Table 3, include the performance of additional classifiers, such as SVM and KNN, along with LR and DNN models. These results demonstrate that federated learning offers superior accuracy, precision, recall, and F1 score across multiple models addressing the XSS issue.

5.5 Non-IID Data Distribution

The results for non-independent and identically distributed (non-IID) data are presented in Table 4. In this setting, each client i had data points belonging to class i of the XSS and CICIDS2017 datasets, with $i = 1,2$. To create the most non-IID case, client one was assigned all the malicious data points, while client two was assigned all the benign data points. The results show that the federated model offers slightly lower scores in some cases than the centralized model. However, the difference is not substantial, indicating that the horizontal FL system can perform well in addressing the XSS problem using FedAvg as the aggregating algorithm. Figure 3 provides an example of the behavior of the Logistic Regression (LR) and Deep Neural Network (DNN) mod-

Table 4: Performance of FL and Traditional Centralized models in binary classification on XSS and CICIDS2017 datasets based on non-IID data distribution.

Dataset	Model	Accuracy	Precision	Recall	F1 Score
XSS	FL model (LR)	98.77%	99.9%	97.08%	98.51%
	Traditional Centralized model (LR)	99.9%	99.9%	99.8%	99.9%
	FL model (DNN)	99.85%	99.9%	99.65%	99.82%
	Traditional Centralized model (DNN)	99.9%	99.9%	99.8%	99.9%
	FL model (SVM)	99.91%	99.90%	99.90%	99.90%
	Traditional Centralized model (SVM)	99.96%	99.96%	99.96%	99.96%
	FL model (KNN)	99.81%	99.95%	99.60%	99.77%
	Traditional Centralized model (KNN)	99.90%	99.90%	99.90%	99.90%
CICIDS2017	FL model (LR)	95.8%	89.7%	89.8%	93.1%
	Traditional Centralized model (LR)	97.9%	94.72%	98.33%	96.49%
	FL model (DNN)	96.08%	97.14%	89.04%	92.91%
	Traditional Centralized model (DNN)	99.8%	99.3%	99.9%	99.6%
	FL model (SVM)	97.73%	97.79%	97.73%	97.74%
	Traditional Centralized model (SVM)	99.31%	99.04%	98.57%	98.80%
	FL model (KNN)	96.63%	89.97%	98.33%	93.97%
	Traditional Centralized model (KNN)	99.38%	98.12%	99.76%	98.93%

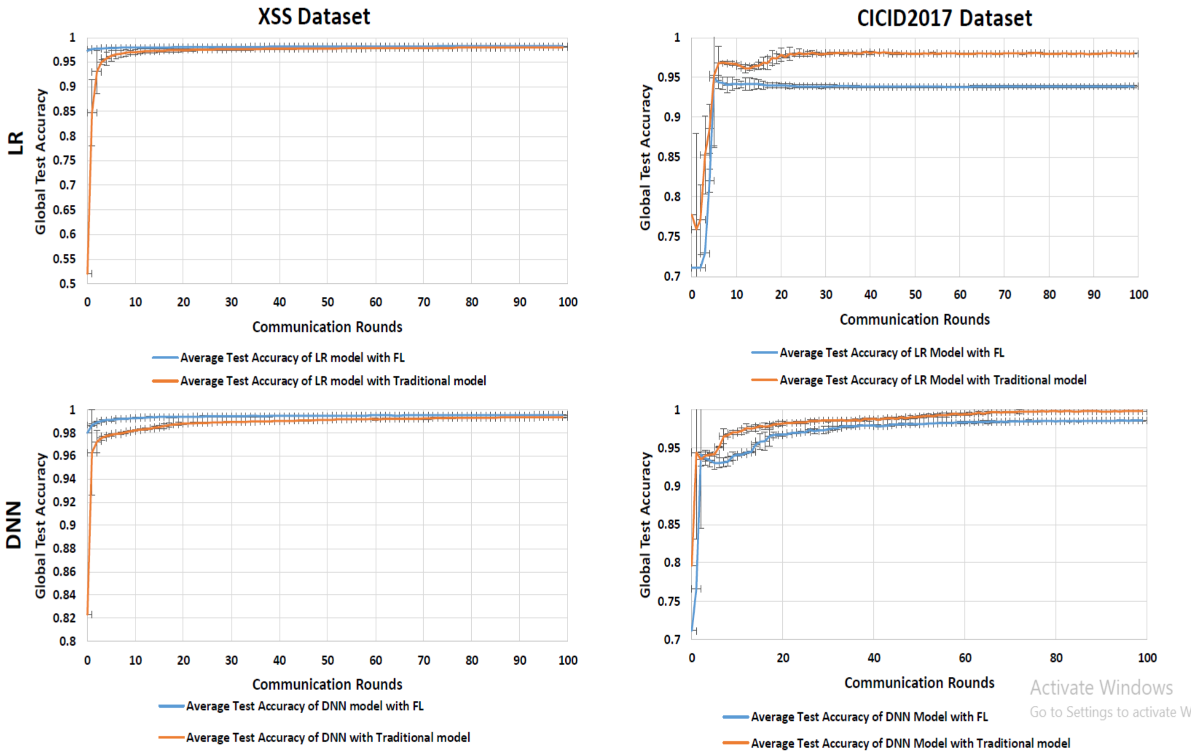


Figure 2: Federated learning with IID data distributions. The columns correspond to the XSS and CICIDS2017 datasets. The rows correspond to the models LR and DNN.

els during the communication rounds, supporting the results shown in Table 4.

Our federated learning framework also supports configurations with more than two clients. We tested the performance of the new classifiers, SVM and KNN, with a setup involving ten clients. Specifically, we assigned five clients to class 0 and the remaining five clients to class 1. In this setup, client i for $i \in \{1,\ldots,5\}$ was assigned all data points of class 0, while client j for $j \in \{6,\ldots,10\}$ was assigned all data points of class 1. This setup, detailed in Table 4, demonstrates the scalability and robustness of our approach.

FL's effectiveness tends to align with the balance

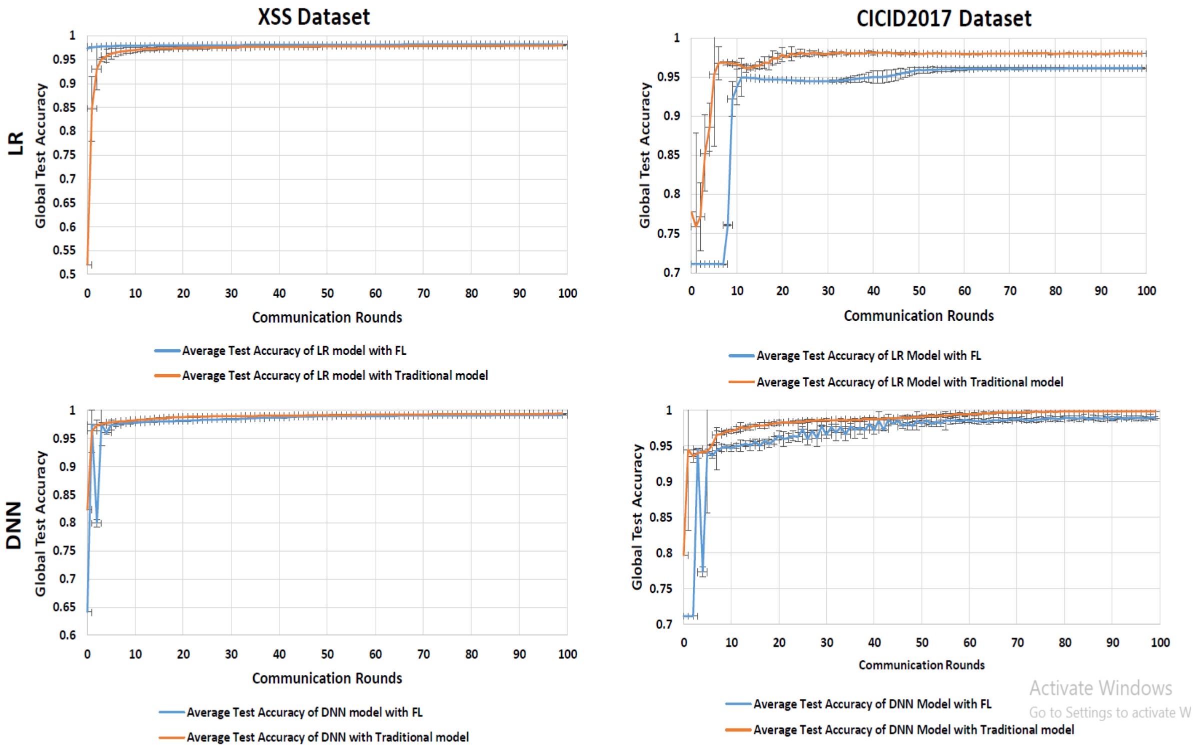


Figure 3: Federated learning with non-IID data distributions. The columns correspond to the XSS and CICIDS2017 datasets. The rows correspond to the models LR and DNN.

between model and dataset complexity. Matching the model complexity to that of the dataset is crucial to fully benefiting from the FL effect. We can evaluate our model against our proposed scheme's benchmarks provided by (Yan et al., 2022; Mereani and Howe, 2018). It is worth noting that all machine learning models referenced in the prior studies are employed as traditional centralized models. A key advantage of our approach is that with federated learning, accessing the clients' data is unnecessary.

6 CONCLUSION

Users' data privacy concerns have become more important, and traditional methods face problems safeguarding sensitive data. Federated learning with ML models can be used as a verification stage to ensure privacy while training ML models. In this research, we presented an innovative and scientifically sound privacy-preserving FL as an alternative method to a centralized model in detecting XSS attacks. Our approach facilitates the training of ML models on distributed devices, effectively mitigating the privacy risks of sensitive data. We comprehensively assessed the proposed framework using authentic, real-world data and compared its efficacy with traditional centralized ML methodologies. The experimental findings strongly indicated that the proposed FL approach attained performance levels comparable to those of centralized models such as LR and DNN while ensuring data privacy. The outcomes affirm that FL holds excellent promise as a viable technique for XSS detection. At the same time, our framework exhibits potential for adaptation to address other security vulnerabilities prevalent in web applications. Future research should aim to gain a more thorough understanding of XSS behavior. We will expand our research to include more attack types, such as SQL injection and cross-site request forgery. Moreover, we will utilize different models and investigate the complexity of these strategies.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge funding from the Koret Foundation grant for Smart Cities and Digital Living 2030.

REFERENCES

Aebersold, S., Kryszczuk, K., Paganoni, S., Tellenbach, B., and Trowbridge, T. (2016). Detecting obfuscated javascripts using machine learning. In *ICIMP 2016 the Eleventh International Conference on Internet Monitoring and Protection, Valencia, Spain, 22-26 May 2016*, volume 1, pages 11–17. Curran Associates.

Ariu, D. and Giacinto, G. (2011). A modular architecture for the analysis of http payloads based on multiple classifiers. In *Multiple Classifier Systems: 10th International Workshop, MCS 2011, Naples, Italy, June 15-17, 2011. Proceedings 10*, pages 330–339. Springer.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191.

Galán, E., Alcaide, A., Orfila, A., and Blasco, J. (2010). A multi-agent scanner to detect stored-xss vulnerabilities. In *2010 International Conference for Internet Technology and Secured Transactions*, pages 1–6. IEEE.

Hsieh, K., Phanishayee, A., Mutlu, O., and Gibbons, P. (2020). The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2021). Advances and open problems in federated learning. *Foundations and trends® in machine learning*, 14(1–2):1–210.

Klein, A. (2005). Dom based cross site scripting or xss of the third kind. *Web Application Security Consortium, Articles*, 4:365–372.

Komiya, R., Paik, I., and Hisada, M. (2011). Classification of malicious web code by machine learning. In *2011 3rd International Conference on Awareness Science and Technology (iCAST)*, pages 406–411. IEEE.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

Kotzur, M. (2022). Privacy protection in the world wide web—legal perspectives on accomplishing a mission impossible. In *Personality and Data Protection Rights on the Internet: Brazilian and German Approaches*, pages 17–34. Springer.

Lee, S., Wi, S., and Son, S. (2022). Link: Black-box detection of cross-site scripting vulnerabilities using reinforcement learning. In *Proceedings of the ACM Web Conference 2022*, pages 743–754.

Li, L., Fan, Y., Tse, M., and Lin, K.-Y. (2020a). A review of applications in federated learning. *Computers & Industrial Engineering*, 149:106854.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2020b). Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450.

Likarish, P., Jung, E., and Jo, I. (2009). Obfuscated malicious javascript detection using classification techniques. In *2009 4th International Conference on Malicious and Unwanted Software (MALWARE)*, pages 47–54. IEEE.

Lin, Y., Han, S., Mao, H., Wang, Y., and Dally, W. J. (2017). Deep gradient compression: Reducing the communication bandwidth for distributed training. *arXiv preprint arXiv:1712.01887*.

Luo, C., Su, S., Sun, Y., Tan, Q., Han, M., and Tian, Z. (2020). A convolution-based system for malicious urls detection. *Computers, Materials & Continua*, 62(1).

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Mereani, F. A. and Howe, J. M. (2018). Detecting cross-site scripting attacks using machine learning. In *International conference on advanced machine learning technologies and applications*, pages 200–210. Springer.

Meyer, R. and Cid, C. (2008). Detecting attacks on web applications from log files. *Sans Institute*.

Ndegwa, A. (2016). What is a web application. *Maxcdn [En línea]*, 31.

Nunan, A. E., Souto, E., Dos Santos, E. M., and Feitosa, E. (2012). Automatic classification of cross-site scripting in web pages using document-based and url-based features. In *2012 IEEE symposium on computers and communications (ISCC)*, pages 000702–000707. IEEE.

OWASP (2017). Owasp top 10 - 2023 rc1. `https://owasp.org`, note = Accessed on [26-9-2023],.

Rocha, T. S. and Souto, E. (2014). Etssdetector: A tool to automatically detect cross-site scripting vulnerabilities. In *2014 IEEE 13th International Symposium on Network Computing and Applications*, pages 306–309. IEEE.

Shamir, O., Srebro, N., and Zhang, T. (2014). Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008. PMLR.

Sharafaldin, I., Lashkari, A. H., Ghorbani, A. A., et al. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, 1:108–116.

Wei-Hong, W., Yin-Jun, L., Hui-Bing, C., and Zhao-Lin, F. (2013). A static malicious javascript detection using svm. In *Conference of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)*, pages 214–217. Atlantis Press.

Wu, D., He, Y., Luo, X., and Zhou, M. (2021a). A latent factor analysis-based approach to online sparse streaming feature selection. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(11):6744–6758.

Wu, D., Luo, X., Shang, M., He, Y., Wang, G., and Zhou, M. (2019). A deep latent factor model for high-dimensional and sparse matrices in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(7):4285–4296.

Wu, D., Shang, M., Luo, X., and Wang, Z. (2021b). An l 1-and-l 2-norm-oriented latent factor model for recommender systems. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5775–5788.

Wu, X., Zheng, W., Chen, X., Zhao, Y., Yu, T., and Mu, D. (2021c). Improving high-impact bug report prediction with combination of interactive machine learning and active learning. *Information and Software Technology*, 133:106530.

Yan, H., Feng, L., Yu, Y., Liao, W., Feng, L., Zhang, J., Liu, D., Zou, Y., Liu, C., Qu, L., et al. (2022). Cross-site scripting attack detection based on a modified convolution neural network. *Frontiers in Computational Neuroscience*, 16:981739.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. (2019). Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19.

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., and Gao, Y. (2021). A survey on federated learning. *Knowledge-Based Systems*, 216:106775.

ZHOU, K., WAN, L., and DING, H.-w. (2019). A cross-site script detection method based on mlp-hmm. *Computer Engineering & Science*, 41(08):1413.

Integrated Evaluation of Semantic Representation Learning, BERT, and Generative AI for Disease Name Estimation Based on Chief Complaints

Ikuo Keshi[1,2], Ryota Daimon[2], Yutaka Takaoka[3,5] and Atsushi Hayashi[4,5]

[1]*AI & IoT Center, Fukui University of Technology, 3-6-1, Gakuen, Fukui, Japan*

[2]*Electrical, Electronic and Computer Engineering Course, Department of Applied Science and Engineering, Fukui University of Technology, 3-6-1, Gakuen, Fukui, Japan*

[3]*Data Science Center for Medicine and Hospital Management, Toyama University Hospital, 2630 Sugitani, Toyama, Japan*

[4]*Department of Ophthalmology, University of Toyama, 2630 Sugitani, Toyama, Japan*

[5]*Center for Data Science and Artificial Intelligence Research Promotion, Toyama University Hospital, 2630 Sugitani, Toyama, Japan*

keshi@fukui-ut.ac.jp, drs0928@gmail.com, ytakaoka@med.u-toyama.ac.jp, ahayashi@med.u-toyama.ac.jp

Keywords: Generative AI, Electronic Medical Record (EMR), Chief Complaints, Disease Name Estimation, Medical AI, Medical Diagnostic Support Tool, Semantic Representation Learning, BERT, GPT-4.

Abstract: This study compared semantic representation learning + machine learning, BERT, and GPT-4 to estimate disease names from chief complaints and evaluate their accuracy. Semantic representation learning + machine learning showed high accuracy for chief complaints of at least 10 characters in the International Classification of Diseases 10th Revision (ICD-10) codes middle categories, slightly surpassing BERT. For GPT-4, the Retrieval Augmented Generation (RAG) method achieved the best performance, with a Top-5 accuracy of 84.5% when all chief complaints, including the evaluation data, were used. Additionally, the latest GPT-4o model further improved the Top-5 accuracy to 90.0%. These results suggest the potential of these methods as diagnostic support tools. Future work aims to enhance disease name estimation through more extensive evaluations by experienced physicians.

1 INTRODUCTION

We developed a method for estimating disease names based on learning semantic representations of medical terms to improve both accuracy and interpretability (Keshi et al., 2022). While semantic representation learning provides high interpretability for discharge summaries, it struggles with texts with poor context, such as a patient's chief complaint. Therefore, we aimed to improve the accuracy and interpretability of disease name estimation by evaluating generative AI techniques like GPT-4.

This study evaluated semantic representation learning to determine the conditions of the chief complaint using generative AI. We conducted a reference evaluation using BERT models (Devlin et al., 2019; Kawazoe et al., 2021), pretrained on Japanese clinical texts, and Wikipedia. Finally, we used an integrated approach to infer disease names from chief complaints, applying zero-shot learning, few-shot learning, and RAG with GPT-4. We comprehensively evaluated these approaches' accuracy and explored their potential application for medical diagnosis.

This study highlights the importance of combining traditional supervised learning and generative AI techniques to improve the accuracy of disease name estimation, especially from minimal contextual data like chief complaints. This combination is crucial to address the challenges of medical diagnosis and enhance accuracy.

2 RELATED RESEARCH

The field of medical AI is rapidly advancing with the application of large language models. Generative AI is being widely adopted in the medical field, and its democratization has the potential to enhance diagnostic accuracy (Chen et al., 2024). Google's Med-PaLM2, fine-tuned with medical texts, has shown high performance in the US medical licensing exam (Singhal et al., 2023). OpenAI's GPT-4 can pass the Japanese national medical exam but still faces challenges in professional medical applica-

Keshi, I., Daimon, R., Takaoka, Y. and Hayashi, A.
Integrated Evaluation of Semantic Representation Learning, BERT, and Generative AI for Disease Name Estimation Based on Chief Complaints.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 294-301
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Table 1: Number of cases in the old EMR corresponding to the top 20 ICD-10 codes in the new EMR.

ICD-10 code	**new EMR**	**old EMR**
C34.1	1127	210
H25.1	929	123
C61	912	2216
C34.3	893	158
C22.0	864	1501
I20.8	698	75
I35.0	690	70
I50.0	545	166
C16.2	536	231
I67.1	515	387
C25.0	503	111
C15.1	483	253
I48	483	253
C34.9	468	1579
P03.4	432	399
C56	393	1276
M48.06	373	845
H35.3	368	1060
H33.0	361	625
C20	357	343

tions (Kasai et al., 2023). In the 2022 National Medical Examination for Physicians (NMLE) in Japan, GPT-4 achieved a correct response rate of 81.5%, significantly higher than GPT-3.5's 42.8%, and exceeded the passing standard of 72%, showing its potential to support diagnostic and therapeutic decisions (Yanagita et al., 2023).

Given these advancements, this study focuses on utilizing these models to establish evaluation criteria for estimating disease names from chief complaints.

3 DATASET

Developing disease estimation AI models using electronic medical records faces the challenge of accuracy drop when applied across different hospitals. This study aims to create models with high accuracy across two types of EMRs with different data distributions.

3.1 Progress Summary Dataset

The training data includes discharge summaries from Toyama University Hospital (2004-2014, 94,083 cases) and the evaluation data from 2015-2019 (61,772 cases). Data cleansing involved excluding cases with missing values, unused fields, rare disease names (less than 0.02%), and short progress summaries (less than 50 words).

Table 1 shows the number of cases in both EMRs for the top 20 disease codes. Despite distribution differences, the top 20 disease codes in the new EMR appear in the old EMR, ensuring sufficient cases for model training and evaluation.

The records include the ICD-10 code, the first 500 characters of the progress summary, department, gender, and age.

Table 2: The number of cases according to different chief complaint conditions.

	old EMR	**new EMR**
Before data cleansing	94,083 cases	61,772 cases
After data cleansing	73,150 cases	48,911 cases
Subcategories with any chief complaint	35,509 cases	28,787 cases
Subcategories with chief complaints of more than 10 characters	8,300 cases	5,876 cases
Middle categories with chief complaints of more than 10 characters	6,766 cases	4,949 cases

Table 3: The number of cases for benchmarks focusing on the top 20 ICD-10 codes.

	old EMR	**new EMR**
Subcategories with any chief complaint	4,205 cases	5,547 cases
Subcategories with chief complaints of more than 10 characters	1,013 cases	1,054 cases
Middle categories with chief complaints of more than 10 characters	1,605 cases	1,715 cases

3.2 Chief Complaint Dataset

Chief complaints were extracted from both EMRs. Table 2 shows the variation in case numbers under different conditions. Table 3 presents benchmarks for the top 20 ICD-10 codes in the new EMR.

In the chief complaint dataset, restricting the number of letters significantly reduces case numbers but retains sufficient data for machine learning. Records include the ICD-10 code, chief complaint, department, gender, and age.

4 PROPOSED METHOD

We developed a model to estimate disease names from chief complaints by extending GPT-4 using EMRs. GPT-4 can pass the Japanese national examination for physicians, but its performance can be improved using the chief complaint dataset from Chapter 3. This study employs supervised learning (semantic representation learning + machine learning) and a BERT model pretrained on medical documents for comparative validation.

4.1 Semantic Representation Learning of Medical Terms

The semantic representation learning process (Figure 1) involves using the first 500 characters of the progress summary. The step of obtaining a weight vector of the progress summary includes generating a paragraph vector (Le and Mikolov, 2014) with initial

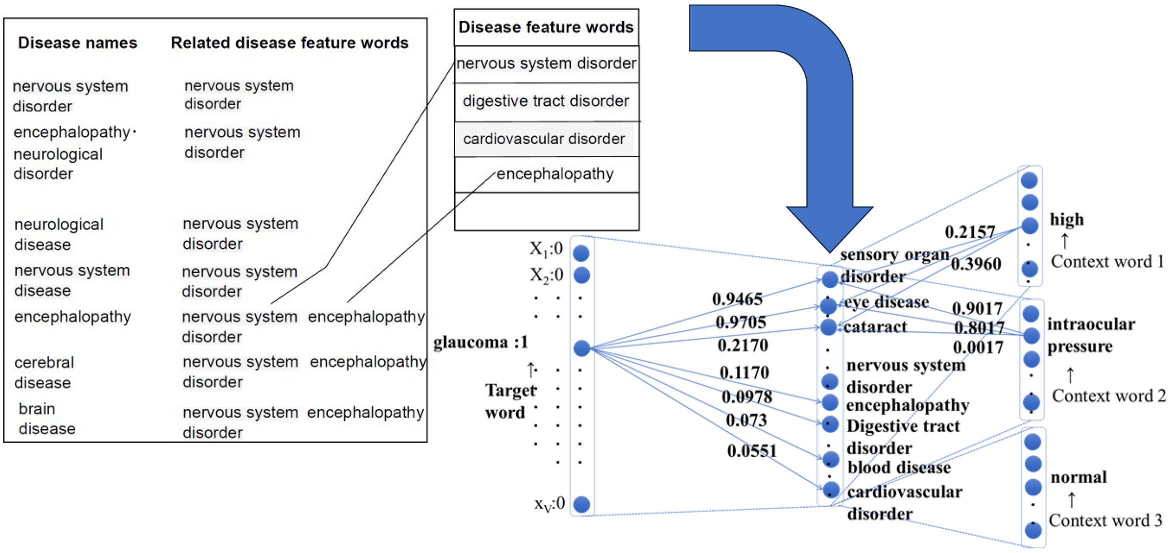


Figure 1: Semantic representation learning process based on the medical-term semantic vector dictionary.

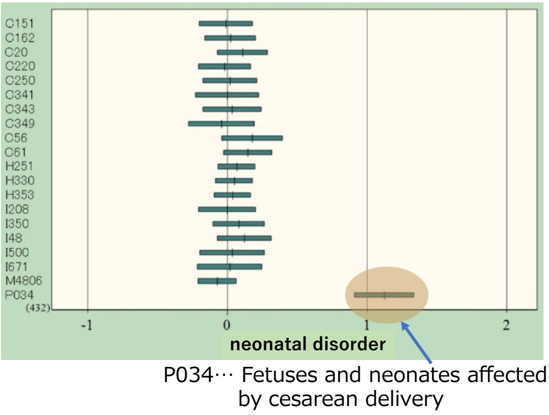


Figure 2: Distribution of weights by ICD-10 code for the disease feature word ”neonatal disorder”.

weights based on the medical-term semantic vector dictionary (Keshi et al., 2022). The resulting paragraph vector, which captures the semantic meaning of the text, is then combined with other explanatory variables such as gender, age, and department. The learning model subsequently uses linear SVM and logistic regression to classify the ICD-10 codes based on these features.

4.1.1 Structure of Medical-Term Semantic Vector Dictionary

The structure of the medical-term semantic vector dictionary is based on the disease thesaurus named T-dictionary[*1]. It associates 299 feature words (264 disease feature words + 35 main symptoms) with basic disease names to provide semantic information for interpretable disease name estimation (Figure 1).

4.1.2 Classification and Visualization

Figure 2 shows the top 20 ICD-10 codes on the vertical axis and the weight distribution of the disease feature word ”neonatal disorder” on the horizontal axis. For ICD-10 code P034, where the mean of the weight distribution is greater than 1.0, it indicates features and neonates affected by cesarean delivery. This visualization facilitates the interpretation of how the model arrived at a particular diagnosis by highlighting the significance of specific disease feature words in the classification process.

4.2 Disease Name Estimation Using BERT

We evaluated a BERT model pretrained on medical documents. The BERT model required pre-training and fine-tuning to achieve accurate disease name estimation.

Table 4 provides information on the BERT models used in the study.

[*1] https://www.tdic.co.jp/products/tdic

[*2] https://github.com/cl-tohoku/bert-japanese

[*3] https://ai-health.m.u-tokyo.ac.jp/home/research/uth-bert

[*4] https://github.com/ou-medinfo/medbertjp

Table 4: Information on the BERT Models Used.

Model Name	Model Size	Training Data
TU-BERT[*2] (Tohoku University BERT)	Base	Japanese Wikipedia (approximately 17 million sentences)
UTH-BERT[*3] (University of Tokyo Hospital BERT)	Base	Clinical texts (120 million records)
MedBERTjp[*4] (Osaka University Graduate School of Medicine BERT)	Base	Japanese Wikipedia + Corpus scraped from "Today's Diagnosis and Treatment: Premium"

4.3 Estimation of Disease Names Using GPT-4

We used GPT-4 (model version: 1106-Preview) from Azure OpenAI Service.[*5], The chief complaint dataset was selected for training and evaluation purposes to avoid personal information. Additionally, we conducted an evaluation using the latest GPT-4o (model version: 2024-05-13) under the same conditions that yielded the best performance in the earlier evaluation.

4.3.1 Zero-Shot Learning

In zero-shot learning, GPT-4 estimated disease names based solely on a system prompt, without any specific training on the target dataset. This approach leverages the model's pre-existing knowledge to make predictions, demonstrating its ability to infer disease names from chief complaints even in the absence of domain-specific data.

4.3.2 Few-Shot Learning

In few-shot learning, one set of chief complaints and corresponding ICD-10 codes for each of the top 20 ICD-10 codes in the new EMR was used from the old EMR, providing 20 sets as example responses to GPT-4.

4.3.3 RAG

The RAG approach used three databases:

- RAG1: A database of chief complaints and ICD-10 codes excluding the chief complaints of the top 20 ICD-10 codes in the new EMR.
- RAG2: A database of chief complaints and ICD-10 codes from the old EMR corresponding to the top 20 ICD-10 codes from the new EMR.
- RAG3: A database linking all chief complaints with corresponding ICD-10 codes, including the evaluation data.

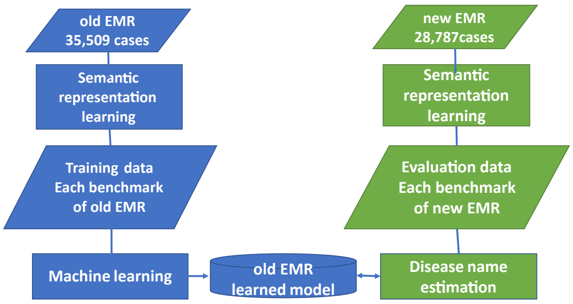


Figure 3: Experimental flow of semantic representation learning.

5 EXPERIMENTAL SETUP

5.1 Semantic Representation Learning + Machine Learning

We used vectors of disease feature words from semantic representation learning to create models using machine learning. Statflex[*6] was employed for interpretability evaluation to graph the variance and mean of the vectors. Figure 3 shows the experimental flow of disease name estimation from chief complaints using semantic representation learning and machine learning.

The datasets of all chief complaints shown in Table 2 (35,509 cases in the old EMR and 28,787 cases in the new EMR) were used for semantic representation learning. We evaluated each benchmark shown in Table 3. Both linear SVM and logistic regression were evaluated due to the shorter text length of chief complaints.

We determined the optimal conditions for chief complaints with the highest accuracy based on overall accuracy and macro-average F1 score of the top 20 ICD-10 codes. These conditions were used in subsequent BERT and GPT-4 experiments.

5.2 BERT

All training data were taken from the progress summary dataset in the old EMR for fine-tuning BERT. The evaluation consisted of two methods:

- Extracting progress summaries related to the top 20 ICD-10 codes from the new EMR and classifying them as evaluation data.
- Extracting chief complaints related to the top 20 ICD-10 codes from the new EMR and classifying them as evaluation data.

[*5] https://portal.azure.com/#view/Microsoft_Azure_ProjectOxford/CognitiveServicesHub/~/OpenAI

[*6] https://www.statflex.net/

5.3 GPT-4

For GPT-4 experiments, we used the chief complaint dataset to avoid personal information.

5.3.1 Zero-Shot Learning

GPT-4 estimated disease names based solely on a system prompt, without any specific training on the target dataset.

System Prompt Example

```
# Role
You are an experienced doctor at a
    ↪ hospital. You will answer
    ↪ questions from young doctors and
    ↪  medical staff in Japanese.
# Objective
Based on the input of the patient's
    ↪ chief complaint, you will
    ↪ perform the following tasks:
- Estimate the patient's disease and
    ↪ provide up to five possible
    ↪ diagnoses along with their ICD
    ↪ -10 codes of middle categories.
# Data Specifications
For each chief complaint, display the
    ↪ ICD-10 code of the middle
    ↪ categories and the top five
    ↪ candidate diagnoses.
# Output Format
The output should be in the following
    ↪ JSON format:
(format details omitted)
```

5.3.2 Few-Shot Learning

Few-shot learning involved providing example sentences to GPT-4 to enable in-context learning.

Few-shot Learning Example

```
{"role": "user", "content": "Loss of
    ↪ appetite, generalized fatigue,
    ↪ pain in dark surroundings"},
{"role": "assistant", "content":"[{"
    ↪ Estimated Disease": "C25", "
    ↪ Diagnosis": "Cancer of the
    ↪ pancreas"]"}
```

5.3.3 RAG

In the experiment, the three configurations RAG1, RAG2, and RAG3 described in the proposed method were used to evaluate the performance of the model. Each configuration was designed to test the model under different conditions, focusing on the availability and relevance of reference data.

RAG External Data Example

```
Diagnosis Code: C34
C34, Back pain, abdominal pain, liver
    ↪ dysfunction
C34, Abnormal sensation in the right
    ↪ upper arm, swelling in the right
    ↪  supraclavicular fossa
```

In the RAG, new and old EMR chief complaints were entered into text files for each ICD-10 code of the middle categories and managed in an Azure storage Blob container. Data was chunked into 512-token segments with 128-token overlap. The search used Azure AI Search's hybrid (keyword + vector) search and semantic ranking features (Berntson et al., 2023).

For evaluation, the Zero-shot learning, Few-shot learning, and RAG methods used the same 200 sets of evaluation data, which consisted of 200 chief complaints randomly selected from the top 20 ICD-10 codes in the new EMR. The results of these evaluations are presented in the following sections. Based on the results of the semantic representation learning experiments, RAG was constructed targeting chief complaints of more than 10 characters in the ICD-10 middle categories. RAG1 and RAG3 included 872 types of ICD-10 codes, while RAG2 focused on the top 20 ICD-10 codes from the new EMR. To align the evaluation with the other two methods, 200 evaluation data sets were constructed by randomly selecting 10 chief complaints from each of the top 20 ICD-10 codes. Each evaluation data set had only one correct ICD-10 code.

6 EVALUATION RESULTS

6.1 Semantic Representation Learning + Machine Learning

The evaluation results of disease name estimation using semantic representation learning and machine learning (logistic regression and linear SVM) based on the chief complaint benchmarks are shown in the first six rows of Table 5. The regularization parameter C was determined using a grid search. The highest overall accuracy was 62.0% when the chief complaint had more than 10 characters and the ICD-10 codes were categorized at the middle level. The highest macro-average F1 score was 51.7 points when the chief complaints had more than 10 characters and the ICD-10 codes were categorized at the subcategory level, using logistic regression. Linear SVM showed the best results (the accuracy: 56.1 %, the F1-score: 49.1) with chief complaints of more than 10 characters and ICD-10 codes categorized at the middle level.

Table 5: Evaluation results of disease name estimation from chief complaints and progress summaries.

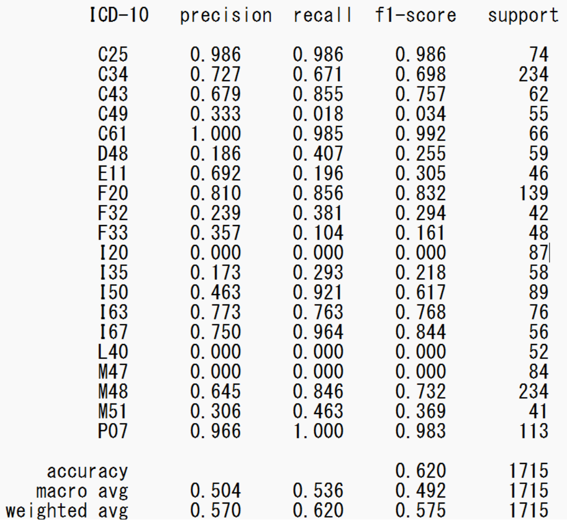
Model Name	Type of Evaluation Data	C value	Accuracy	F1-score
Semantic Representation Learning + Logistic Regression	Chief Complaints (Any chars, Subcategories)	60.0	36.0%	29.5
Semantic Representation Learning + Logistic Regression	Chief Complaints (10+ chars, Subcategories)	49.0	49.4%	51.7
Semantic Representation Learning + Logistic Regression	Chief Complaints (10+ chars, Middle Categories)	34.0	62.0%	49.2
Semantic Representation Learning + Linear SVM	Chief Complaints (Any chars, Subcategories)	250	26.2%	22.7
Semantic Representation Learning + Linear SVM	Chief Complaints (10+ chars, Subcategories)	130	44.5%	48.6
Semantic Representation Learning + Linear SVM	Chief Complaints (10+ chars, Middle Categories)	41.0	56.1%	49.1
Semantic Representation Learning + Linear SVM	Progress Summaries (500 chars, Subcategories)	N/A	69.5%	72.1
TU-BERT	Progress Summaries (500 chars, Subcategories)	N/A	77.5%	80.0
UTH-BERT	Progress Summaries (500 chars, Subcategories)	N/A	83.8%	85.3
MedBERTjp	Progress Summaries (500 chars, Subcategories)	N/A	77.1%	80.4
TU-BERT	Chief Complaints (10+ chars, Middle Categories)	N/A	52.2%	44.1
UTH-BERT	Chief Complaints (10+ chars, Middle Categories)	N/A	61.1%	53.7
MedBERTjp	Chief Complaints (10+ chars, Middle Categories)	N/A	53.4%	45.7

Figures 4 and 5 show the evaluation results of ICD-10 codes categorized at the middle and subcategory levels for chief complaints with more than 10 characters when using logistic regression. For the middle categories, three ICD-10 codes (I20, L40, M47) had an F1 score of 0, while no subcategory disease names had an F1 score of 0. This suggests a higher overfitting risk for subcategories. Therefore, the condition of chief complaints with more than 10 characters at the middle category level will be used for BERT and GPT-4 evaluations.

6.2 BERT

The four rows starting from the middle of Table 5 shows the evaluation results of classifying progress summaries (up to 500 characters) extracted from the top 20 ICD-10 codes (subcategories) in the new EMR as evaluation data. The macro-average F1-score for semantic representation learning was 72.1, while the fine-tuned large language model using UTH-BERT achieved a macro-average F1-score of 85.3, surpassing semantic representation learning by over 10 points.

For the evaluation based on chief complaints, as shown in the last three rows of Table 5, UTH-BERT had the highest accuracy and macro-average F1 score among the BERT models. However, the accuracy of semantic representation learning combined with logistic regression slightly exceeded that of the BERT models.

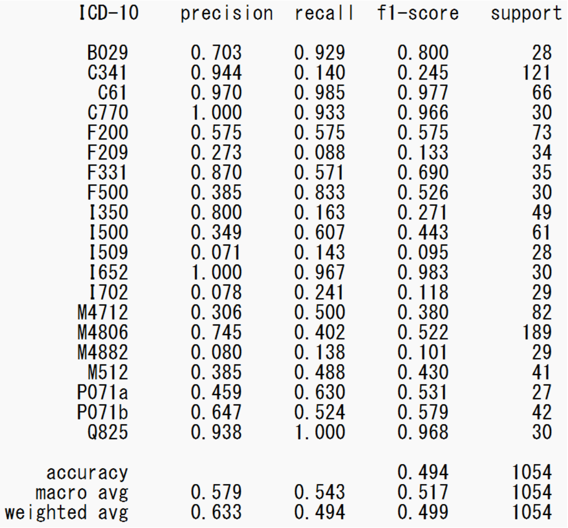


```
      ICD-10   precision   recall   f1-score   support

         C25     0.986      0.986     0.986        74
         C34     0.727      0.671     0.698       234
         C43     0.679      0.855     0.757        62
         C49     0.333      0.018     0.034        55
         C61     1.000      0.985     0.992        66
         D48     0.186      0.407     0.255        59
         E11     0.692      0.196     0.305        46
         F20     0.810      0.856     0.832       139
         F32     0.239      0.381     0.294        42
         F33     0.357      0.104     0.161        48
         I20     0.000      0.000     0.000        87
         I35     0.173      0.293     0.218        58
         I50     0.463      0.921     0.617        89
         I63     0.773      0.763     0.768        76
         I67     0.750      0.964     0.844        56
         L40     0.000      0.000     0.000        52
         M47     0.000      0.000     0.000        84
         M48     0.645      0.846     0.732       234
         M51     0.306      0.463     0.369        41
         P07     0.966      1.000     0.983       113

    accuracy                          0.620      1715
   macro avg     0.504      0.536     0.492      1715
weighted avg     0.570      0.620     0.575      1715
```

Figure 4: Disease name estimation using semantic representation learning and logistic regression for ICD-10 codes categorized at the middle level with chief complaints of more than 10 characters.

6.3 GPT-4

Table 6 shows the evaluation results of GPT-4 in estimating disease names from chief complaints (200 sets of evaluation data). The Top-5 accuracy was measured, considering a result correct if the cor-



```
       ICD-10    precision   recall   f1-score   support

         B029      0.703     0.929     0.800        28
         C341      0.944     0.140     0.245       121
          C61      0.970     0.985     0.977        66
         C770      1.000     0.933     0.966        30
         F200      0.575     0.575     0.575        73
         F209      0.273     0.088     0.133        34
         F331      0.870     0.571     0.690        35
         F500      0.385     0.833     0.526        30
         I350      0.800     0.163     0.271        49
         I500      0.349     0.607     0.443        61
         I509      0.071     0.143     0.095        28
         I652      1.000     0.967     0.983        30
         I702      0.078     0.241     0.118        29
        M4712      0.306     0.500     0.380        82
        M4806      0.745     0.402     0.522       189
        M4882      0.080     0.138     0.101        29
         M512      0.385     0.488     0.430        41
        P071a      0.459     0.630     0.531        27
        P071b      0.647     0.524     0.579        42
         Q825      0.938     1.000     0.968        30

     accuracy                          0.494      1054
    macro avg      0.579     0.543     0.517      1054
 weighted avg      0.633     0.494     0.499      1054
```

Figure 5: Disease name estimation using semantic representation learning and logistic regression for ICD-10 codes categorized at the subcategory level with chief complaints of more than 10 characters.

Table 6: Evaluation results of disease name estimation from chief complaints (200 sets of evaluation data).

	Top-5 Acc.	Top-1 Acc.
Zero-shot Learning	52.5%	22.0%
Few-shot Learning	61.0%	20.0%
RAG1: All cases except the benchmark cases in the new EMR (15 reference documents)	65.5%	19.5%
RAG2: Only the benchmark cases in the old EMR (5 reference documents)	82.5%	24.0%
RAG3: All cases, including the benchmark cases in the new EMR (15 reference documents)	84.5%	25.0%
RAG3: GPT-4o	90.0%	26.5%

rect ICD-10 code was among the top five candidates. Zero-shot learning achieved a Top-5 accuracy of 52.5%, while few-shot learning improved it to 61.0%. RAG1 achieved 65.5% with 15 reference documents, RAG2 reached 82.5% with 5 reference documents, and RAG3 achieved the highest Top-5 accuracy of 84.5% with 15 reference documents.

Additionally, the latest GPT-4o was evaluated under the same conditions as RAG3, achieving the highest Top-5 accuracy of 90.0%. Excluding one chief complaint where a response was not generated due to content filtering, GPT-4o's Top-5 accuracy reached 90.5%.

Figure 6 illustrates the relationship between the number of reference documents and the Top-5 accuracy for RAG1. The accuracy improves as the number of reference documents increases, with the best performance achieved at 15 reference documents.

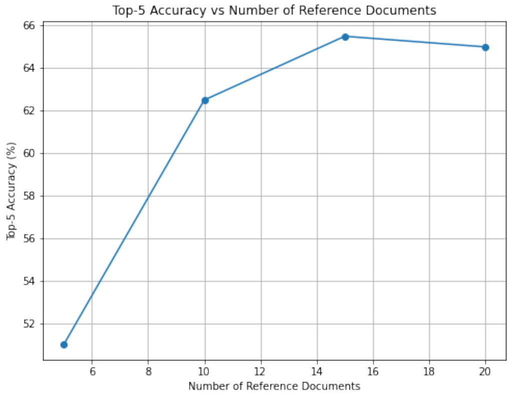


Figure 6: Top-5 Accuracy vs Number of Reference Documents.

7 DISCUSSION

This study confirmed that the accuracy of disease name estimation significantly decreases when changing the target from progress summaries to chief complaints. However, using semantic representation learning, logistic regression achieved an accuracy of 62.0% for chief complaints of more than 10 characters classified at the middle category level. This slightly exceeded the accuracy of UTH-BERT, which was fine-tuned with over 10,000 progress summaries, while semantic representation learning used only 1,605 chief complaints. However, for 3 out of the 20 ICD-10 codes, the estimation accuracy was 0%. This is because chief complaints often consist of general symptoms like "fever" or "dizziness," which do not include disease names registered in the medical-term semantic vector dictionary. If the chief complaint does not include a disease name, the feature vector does not change, leading to estimation failure.

In cases where the data is rich in context, such as progress summaries of up to 500 characters, SVM tends to perform better due to its ability to capture complex relationships within the data. However, for datasets like chief complaints, which are often lacking in context, logistic regression may be more suitable. This is because logistic regression is a simpler model that is less prone to overfitting, making it better suited to handle sparse and less informative data. The results suggest that logistic regression was better suited for the chief complaint dataset due to its simplicity and robustness. Similarly, this may also explain why semantic representation learning slightly outperformed BERT, as the former was better able to handle the limited context and information present in the chief complaints.

GPT-4 showed significant improvement in Top-5 accuracy with few-shot learning, providing 20 sets of example sentences, and RAG, using only the chief complaints and ICD-10 codes from the old EMR as external data. The contextual limitation likely contributed to this improvement. For RAG without correct cases, fewer reference documents resulted in lower accuracy than few-shot learning, highlighting the importance of data quality over quantity.

The evaluation set was limited to the top 20 disease names, and GPT-4 generated 5 candidate disease names. Expanding the evaluation set to a wider range of disease names and conducting evaluations using external data is necessary. Additionally, subjective evaluation of the validity and diagnostic reasons by veteran physicians is important.

8 CONCLUSIONS

This study compared disease name estimation methods using semantic representation learning + machine learning, BERT, and GPT-4, and evaluated their accuracy. Despite being trained on only 1,605 chief complaints, semantic representation learning + machine learning showed slightly higher accuracy than BERT, which was fine-tuned on over 10,000 progress summaries, under certain conditions. However, it was found to have limitations in disease name estimation based on chief complaints.

For GPT-4, evaluation data were created based on the top 20 disease names with the highest occurrence frequency in the new EMR, targeting cases with chief complaints of more than 10 characters. Evaluations using zero-shot learning, few-shot learning, and RAG demonstrated that RAG achieved the highest performance. When all chief complaints, including the evaluation data, were used, the highest Top-5 accuracy of 84.5% was achieved, while excluding the evaluation data decreased the accuracy to 65.5%. The optimal number of reference chunks for RAG was 15. Even when excluding the evaluation data, limiting the database to the 20 diagnostic disease names improved the Top-5 accuracy to 82.5%. Furthermore, the latest GPT-4o model was evaluated under the same conditions as RAG, and it further improved the Top-5 accuracy to 90.0%.

In the future, we aim to expand the benchmark to cover additional middle categories of ICD-10, conduct more extensive evaluations, and perform subjective evaluations by experienced physicians. This aims to implement disease name estimation from chief complaints as a practical diagnostic support tool in medical settings.

ACKNOWLEDGMENTS

Part of this study was conducted by Shuta Asai, Tatsuki Sakata as their graduation research in 2023, and is currently being conducted by Mikio Osaki as part of his ongoing graduation research in 2024, all from Fukui University of Technology. We thank them for their contributions. This work was supported by JSPS KAKENHI Grant Number 24K14964 and 20K11833. This study was approved by the Ethical Review Committee of the Fukui University of Technology and the Toyama University Hospital.

REFERENCES

Berntson, A. et al. (2023). Azure ai search: Outperforming vector search with hybrid retrieval and ranking capabilities. https://techcommunity.microsoft.com/t5/ai-azure-ai-services-blog/azure-ai-search-outperforming-vector-search-with-hybrid/ba-p/3929167. Accessed: 2024-05-18.

Chen, A., Liu, L., and Zhu, T. (2024). Advancing the democratization of generative artificial intelligence in healthcare: a narrative review. *Journal of Hospital Management and Health Policy*, 8(0).

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Kasai, J., Kasai, Y., Sakaguchi, K., Yamada, Y., and Radev, D. (2023). Evaluating gpt-4 and chatgpt on japanese medical licensing examinations.

Kawazoe, Y., Shibata, D., Shinohara, E., Aramaki, E., and Ohe, K. (2021). A clinical specific bert developed using a huge japanese clinical text corpus. *PLoS One*, 16(11)(9).

Keshi, I., Daimon, R., and Hayashi, A. (2022). Interpretable disease name estimation based on learned models using semantic representation learning of medical terms. In Coenen, F., Fred, A. L. N., and Filipe, J., editors, *Proceedings of the 14th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2022, Volume 1: KDIR, Valletta, Malta, October 24-26, 2022*, pages 265–272. SCITEPRESS.

Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proc. of ICML*, pages 1188–1196.

Singhal, K. et al. (2023). Towards expert-level medical question answering with large language models.

Yanagita, Y., Yokokawa, D., Uchida, S., Tawara, J., and Ikusaka, M. (2023). Accuracy of chatgpt on medical questions in the national medical licensing examination in japan: Evaluation study. *JMIR Form Res*, 7:e48023.

RUDEUS: A Machine Learning Classification System to Study DNA-Binding Proteins

David Medina-Ortiz[1,2][a], Gabriel Cabas-Mora[1][b], Iván Moya[1][c], Nicole Soto-García[1][d]
and Roberto Uribe-Paredes[1]

[1]*Departamento de Ingeniería En Computación, Universidad de Magallanes, Avenida Bulnes 01855, Punta Arenas, Chile*
[2]*Centre for Biotechnology and Bioengineering, CeBiB, Universidad de Chile, Beauchef 851, Santiago, Chile*
david.medina@umag.cl, roberto.uribe@umag.cl

Keywords: DNA-Binding Proteins, Single-Stranded and Double-Stranded DNA, Machine Learning, Protein Language Models.

Abstract: DNA-binding proteins play crucial roles in biological processes such as replication, transcription, packaging, and chromatin remodeling. Their study has gained importance across scientific fields, with computational biology complementing traditional methods. While machine learning has advanced bioinformatics, generalizable pipelines for identifying DNA-binding proteins and their specific interactions remain scarce. We present RUDEUS, a Python library with hierarchical classification models to identify DNA-binding proteins and distinguish between single- and double-stranded DNA interactions. RUDEUS integrates protein language models, supervised learning, and Bayesian optimization, achieving 95% precision in DNA-binding identification and 89% accuracy in distinguishing interaction types. The library also includes tools for annotating unknown sequences and validating DNA-protein interactions through molecular docking. RUDEUS delivers competitive performance and is easily integrated into protein engineering workflows. It is available under the MIT License, with the source code and models available on the GitHub repository https://github.com/ProteinEngineering-PESB2/RUDEUS.

1 INTRODUCTION

DNA-protein interactions are fundamental to numerous cellular processes critical for biological functions. Approximately 6-7% of eukaryotic proteins interact with DNA, utilizing specific DNA-binding domains and varying affinities for single- and double-stranded DNA (Attali et al., 2021; Gupta et al., 2021). These interactions are driven by direct base–amino acid recognition and indirect forces from DNA conformational changes (Arora et al., 2023).

DNA-binding proteins (DBPs) play key roles in processes like DNA replication, transcription, packaging, and chromatin remodeling (Kabir et al., 2024). They aid in strand separation, maintain DNA integrity, regulate gene expression, and influence chromatin structure. Understanding DBPs is essential for insights into gene regulation and links between mutations and genetic diseases (Zhang et al., 2022; Kabir et al., 2024). Recent studies on proteins such as TDP-43 and helicase chromodomain proteins have advanced knowledge in fields like neurodegeneration and cancer (Lye and Chen, 2022; Alendar and Berns, 2021; Wang et al., 2022a).

Computational biology, bolstered by AI and machine learning, has enhanced the discovery of DBPs by predicting interaction sites and transcription factor binding hotspots (Wang et al., 2022b). While many machine learning models have been applied, including deep learning, comparing them is difficult due to variations in datasets and validation methods (Shadab et al., 2020; Zhang et al., 2020; Ali et al., 2022; Banjar et al., 2022; Barukab et al., 2022). Recent approaches have employed large protein language models for more robust numerical representations (Medina et al., 2023; Medina-Ortiz et al., 2024; Fernández et al., 2023).

This paper introduces RUDEUS, a Python library designed for DNA-binding classification and distinguishing between single- and double-stranded interactions. RUDEUS combines protein language mod-

[a] https://orcid.org/0000-0002-8369-5746
[b] https://orcid.org/0009-0004-2344-9860
[c] https://orcid.org/0000-0002-0458-378X
[d] https://orcid.org/0009-0001-1438-1938

Medina-Ortiz, D., Cabas-Mora, G., Moya, I., Soto-García, N. and Uribe-Paredes, R.
RUDEUS: A Machine Learning Classification System to Study DNA-Binding Proteins.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 302-310
ISBN: 978-989-758-716-0; ISSN: 2184-3228

els, supervised learning algorithms, and Bayesian hyperparameter tuning to build predictive models. Achieving precision rates of 95% for DNA-binding identification and 89% for interaction type evaluation, RUDEUS demonstrates strong performance. It annotated over 20,000 protein sequences and was validated using structural bioinformatics. The library's flexibility and ease of use make it a valuable tool for exploring latent space and mutation landscapes in DBPs.

2 METHODS

2.1 Collecting and Processing Protein Sequences

All protein sequences were sourced from the literature, including datasets from Hu et al. (2019); Shadab et al. (2020); Sharma et al. (2021); Wang et al. (2017). After collection, a preprocessing step was applied to merge, clean, and remove redundancy and inconsistencies. Filters were then applied to exclude non-canonical sequences and select sequences within a length range of 50 to 1024 amino acids. Additionally, homology redundancy was eliminated using the CDHit library Fu et al. (2012).

2.2 Numerical Representation Strategies

This work explore different pre-trained models based on protein language models, including ProTrans (Elnaggar et al., 2020) and ESM (Rives et al., 2021; Meier et al., 2021). All pre-trained models were applied through the bio-embedding tool, combined with a reduction process to obtain vectors in a $1-D$ dimension (Dallago et al., 2021). Moreover, physicochemical based approaches and Fourier transforms also were explored (Medina-Ortiz et al., 2022, 2020a).

2.3 Training Predictive Models and Tuning Optimization

A classic machine learning pipeline was employed to train predictive models Medina-Ortiz et al. (2020b). The datasets were first split into training (70%), validation (20%), and testing (10%) sets. The models were then trained using the strategies proposed in Medina-Ortiz et al. (2024), which included an exploration phase, statistical methods to select the best combinations of numerical representation strategies and machine learning algorithms, and Bayesian approaches for hyperparameter tuning Akiba et al. (2019). Once the models were trained, the testing datasets were used for benchmarking, and the models were deployed to predict unknown protein sequences (See Figure 3 of Appendix for a schematic representation of the employed pipeline to train the predictive models).

2.4 Structural Bioinformatics Approaches

RUDEUS incorporates a structural bioinformatics pipeline to validate model predictions using DNA-protein molecular docking via LightDock v9.4 Roel-Touris et al. (2020). The pipeline prepares protein structures by applying protonation, hydrogen deletion, structure rebuilding with the Reduce library, and modifying atoms to comply with the AMBER94 force field. After preparation, molecular docking is performed with 400 swarms, 200 glowworms, and 100 steps. The resulting conformers are clustered using the RMSD metric with the BSAS function, and the best pose is selected based on the highest docking score.

2.5 Availability and Implementation Strategies

All source code was implemented under the Python Language programming v3.9.16, including the modules, libraries, and demonstration scripts in RUDEUS. The main libraries employed to develop the predictive models were scikit-learn (Pedregosa et al., 2011) and Optuna (Akiba et al., 2019). Furthermore, to process and compile all datasets, the Pandas library was employed (McKinney et al., 2011). Finally, a conda environment was constructed to facilitate the deployment of the built library, combined with different Jupyter Notebooks, to ensure the replicability of the presented work. All source code, environment configuration, datasets, and created models are available for non-commercial uses in the GitHub repository under the MIT licence https://github.com/ProteinEngineering-PESB2/RUDEUS.

3 RESULTS AND DISCUSSIONS

3.1 RUDEUS Achieves High Performances in Its Classification Models

Two classification tasks were explored in RUDEUS: DNA-binding protein classification and the identification of single- versus double-stranded DNA interactions. For each task, over 10,000 combinations of numerical representation strategies and supervised learning algorithms were evaluated. The models' performance was measured using accuracy, precision, recall, and F-score.

Figure 4 of Appendix displays the recall metric distributions for the training process. On average, the models achieved 83% precision for DNA-binding classification and 82% precision for DNA strand type prediction. The highest-performing DNA-binding models were based on pre-trained ProtTrans Uniref, BDF, and XLU50 models, independent of the learning algorithm. For DNA strand type classification, the best results came from ProtTrans XLU50, Uniref, t5bdf, ESM1B, and ESM1V models. Ensemble methods like Random Forest, Gradient Boosting, ExtraTrees, and KNN consistently delivered the top results for both tasks.

A statistical selection process identified the best combinations of representation strategies and algorithms. Sixteen Bernoulli events were evaluated using two filters: i) top-performing models above the 90th quantile and ii) models with standard deviations below the 10th quantile. A binomial distribution was then applied to detect outliers, with a success threshold of > 12 events, representing a success probability below 0.01. This stringent selection yielded five optimal combinations for DNA-binding classification and four for DNA strand type prediction, as summarized in Table 1. While the selected models exhibited strong performance, overfitting was observed, with differences between training and validation metrics.

Table 1: Selected combinations of supervised learning algorithms and numerical representation approaches for all tasks explored in this work.

Task	Algorithm	Encoder	Recall
DNA-binding classification	ExtraTrees	prot. Uniref	0.93
	ExtraTrees	prot. bdf	0.93
	Gradient B	prot. Uniref	0.91
	KNeighbors	prot. Uniref	0.93
	RandomForest	prot. Uniref	0.93
Single-stranded or double-stranded	ExtraTrees	prot. Uniref	0.90
	ExtraTrees	prot. XLU50	0.90
	Gaussian Pro.	prot. XLU50	0.89
	SVC	prot. XLU50	0.90

All selected combinations of supervised learning algorithms and numerical representation strategies were optimized using the Optuna library (Akiba et al., 2019). Two models were then selected based on the criteria outlined in the pipeline. For DNA-binding prediction, the ExtraTrees algorithm combined with the ProtTrans Uniref model was chosen, while for single-stranded or double-stranded DNA interaction, the same algorithm was used, but the ProtTrans XLU50 model was selected. The DNA-binding model achieved 95% precision with a Matthews correlation coefficient (MCC) of 0.89, and the single-stranded/double-stranded model achieved 89% precision with an MCC of 0.81.

Figure 1 summarizes both models' performance. The confusion matrices (Figure 1 A and 1 C) indicate strong performance in identifying positive and negative classes, with the DNA-binding model outperforming the single/double-stranded model in distinguishing interactions. Precision-recall curves (Figure 1 B and 1 D) showed average precision values of 0.98 and 0.96, respectively, aligning with the confusion matrices and demonstrating the greater difficulty in classifying interaction types. ROC curves, calculated using $k = 5$ cross-validation, revealed area under the curve (AUC) scores of 0.98 for DNA-binding and 0.97 for the interaction model, confirming the models' robust predictive capabilities.

Table 2 compares the RUDEUS models with state-of-the-art methods. For DNA-binding, RUDEUS achieved the highest specificity (95.5%) and MCC (0.89), while the method in (Zhang et al., 2021) had the highest sensitivity, differing only by 0.1% from RUDEUS. For the single-stranded/double-stranded task, RUDEUS achieved the highest MCC (0.81), although other methods reported higher sensitivity (Ali et al., 2020) and specificity (Tan et al., 2019). However, these methods showed signs of overfitting, as indicated by large gaps between sensitivity and specificity and lower MCC values compared to RUDEUS.

Table 2: State-of-the-art comparison for DNA-binding classification models and single-stranded or double-stranded interaction models.

Task	Classifier	SN(%)	SP(%)	MCC	Reference
DNA-binding	RF	79.3	89.0	0.69	(Kumar et al., 2009)
	RF	83.7	90.0	0.72	(Ma et al., 2016)
	SVM	87.0	85.5	0.72	(Zaman et al., 2017)
	SVM	89.1	88.8	0.78	(Ali et al., 2018)
	SVM	94.1	97.6	0.92	(Rahman et al., 2018)
	SVM	91.1	88.8	0.79	(Mishra et al., 2019)
	SVM	91.8	93.0	0.84	(Ali et al., 2019)
	SVM	**93.4**	93.4	0.86	(Zhang et al., 2021)
	ExtraTrees	93.3	**95.5**	**0.89**	**This work**
Single-stranded or double-stranded	RF	90.8	78.8	0.64	(Wang et al., 2017)
	SVM	**94.2**	80.33	0.72	(Ali et al., 2020)
	GTB	78.4	**97.5**	0.79	(Tan et al., 2019)
	HMM	85.3	92.8	0.78	(Sharma et al., 2021)
	ExtraTrees	87.8	91.6	**0.81**	**This work**

3.2 RUDEUS Facilitate the Exploration of Single-Stranded or Double-Stranded Interaction Evaluation

More than 20,000 DNA-binding protein sequences were classified as either single- or double-stranded

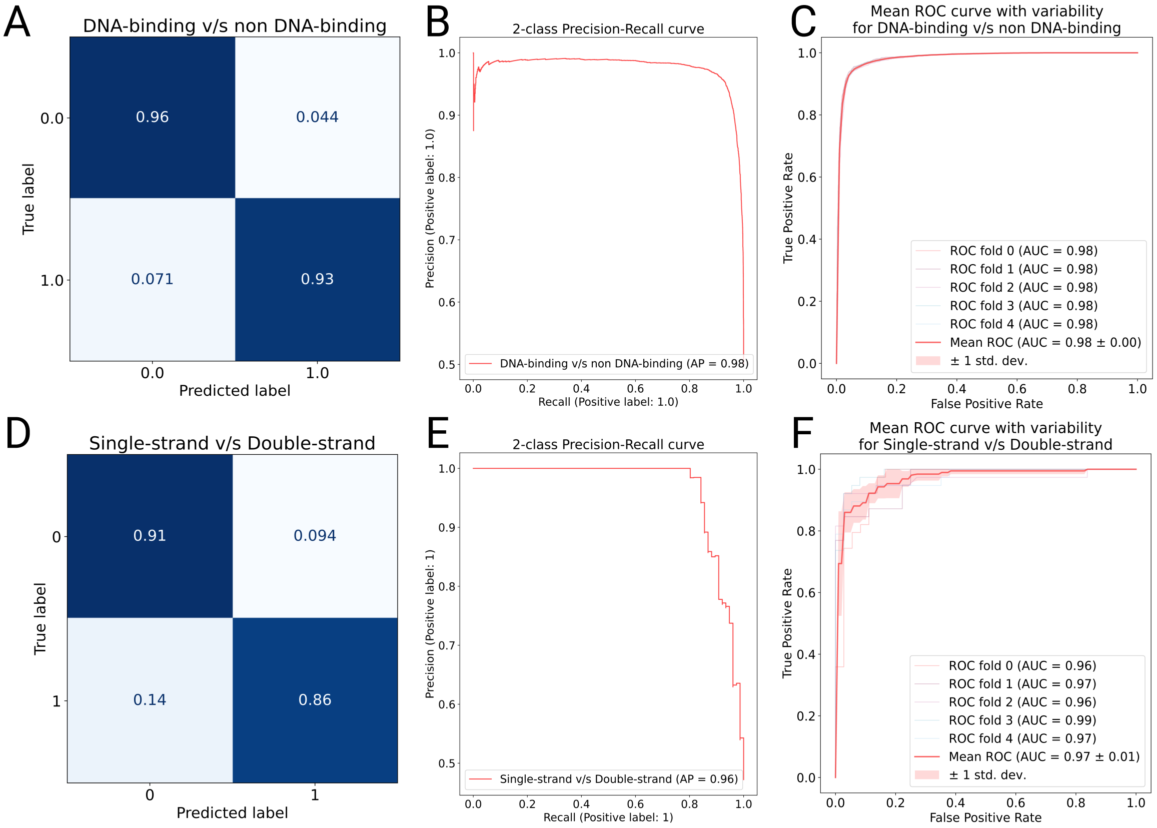


Figure 1: **Description through different performances visualization the selected and optimized models for both tasks explored in this work**. **A-D** Confusion matrix estimated during the validation process for DNA-binding task single-stranded or double-stranded task, respectively. **B-E** Precision-recall curve estimated during the validation process for DNA-binding task single-stranded or double-stranded task, respectively. The average precision (AP) was calculated in both cases, achieving 0.98 and 0.96, respectively. **C-F** Receiver operating characteristic (ROC) curve estimated during the training process for DNA-binding task single-stranded or double-stranded task, respectively. In both cases, the area under the curve (AUC) was estimated to achieve 0.98 and 0.97, respectively.

using the exploration module in RUDEUS. First, the sequences were numerically represented using pre-trained models selected for strand interaction classification. The predictions showed that over 18,000 proteins were classified as double-stranded, while around 2,000 were identified as single-stranded, reflecting proportions similar to the dataset used for model training.

Three DNA-binding proteins with identified strand interactions were further evaluated using the bioinformatics structural pipeline. Figure 2 provides molecular docking visualizations and detailed interaction site analyses for these proteins, all of which were previously reported in the literature.

Figure 2 A illustrates the molecular docking of protein 1BNZ, a hyperthermophile chromosomal protein that binds double-stranded DNA (Gao et al., 1998; Guagliardi et al., 2002). Key hydrophobic residues—TRP24, VAL26, MET29, and ALA45—play a significant role in DNA binding (Figure 2 B). Interactions occur via hydrogen bonds, salt bridges, and van der Waals contacts, consistent with previous reports (Gao et al., 1998).

Similarly, Figure 2 C shows the docking of protein 1HRY, which is involved in sexual differentiation by regulating the gene responsible for Müllerian duct regression in male embryos (Werner et al., 1995). Six residues (ASN10, PHE12, ILE13, SER33, ILE35, SER36, TYR74) interact with DNA bases, forming hydrogen bonds and electrostatic interactions (Figure 2 D), as described in (Werner et al., 1995).

In contrast, Figure 2 E presents the docking of protein 3ULP, known as Pf-SSB, a single-stranded DNA-binding protein crucial for DNA metabolism in the malaria-causing parasite (Antony et al., 2012). The homotetramer structure of 3ULP features identical DNA-contacting residues (S110, N114, T129) across all four subunits (Figure 2 F), which form part

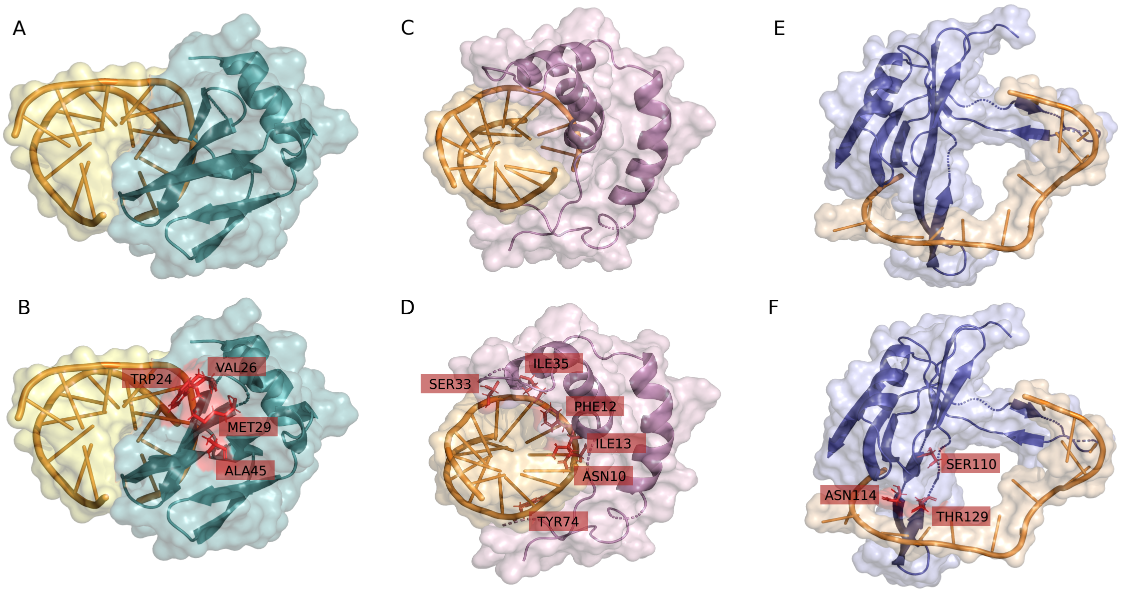


Figure 2: **Structural bioinformatics validation through DNA-protein molecular docking for three DNA-binding proteins and their interaction type identified with the models available in RUDEUS**. **A-B** DNA-protein molecular docking and the most relevant identified residues for the DNA interaction for the protein 1BNZ. **C-D** DNA-protein molecular docking and the most relevant identified residues for the interaction for the protein 1HRY. **E-F** DNA-protein molecular docking and the most relevant identified residues for the interaction for the protein 3ULP.

of the replication and maintenance machinery in the apicoplast (Antony et al., 2012).

4 CONCLUSIONS

This work introduces RUDEUS, a Python library specifically designed for the investigation and classification of DNA-binding proteins, as well as the identification of DNA strand interaction types. The methodology incorporates a flexible pipeline that leverages protein language models, supervised learning algorithms, and Bayesian optimization to train high-performance classification models. These models surpass state-of-the-art benchmarks in sensitivity, specificity, and MCC scores, demonstrating RUDEUS's superiority in this domain, while maintaining the simplicity and replicability of existing methods.

An extensive exploration process highlighted the utility of RUDEUS, enabling the annotation of over 20,000 protein sequences as single- or double-stranded, validated through structural bioinformatic approaches and DNA-protein molecular docking. RUDEUS's intuitive interface and powerful features make it highly applicable for integration into broader protein design pipelines, including landscape reconstruction, directed evolution, and latent space exploration using deep generative models.

COMPETING INTERESTS

The authors declare that the research was conducted without any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS STATEMENT

IM-B and DM-O: conceptualization. DM-O, GC-M, and NS-G: methodology. DM-O and RU-P: validation. IM-B, GC-M, and NS-G: investigation. DM-O, IM-B, RU-P, and GC-M: writing, review, and editing. DM-O and RU-P: supervision and funding resources. DM-O: project administration.

ACKNOWLEDGEMENTS

This research has been financed mainly by the Centre for Biotechnology and Bioengineering - CeBiB (PIA project FB0001, Conicyt, Chile). DM-O acknowledges ANID for the project "SUBVENCIÓN A IN-

STALACIÓN EN LA ACADEMIA CONVOCATORIA AÑO 2022", Folio 85220004. RU-P acknowledges ANID for the grant Fondecyt 1230298.

REFERENCES

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631.

Alendar, A. and Berns, A. (2021). Sentinels of chromatin: chromodomain helicase dna-binding proteins in development and disease. *Genes & Development*, 35(21-22):1403–1430.

Ali, F., Ahmed, S., Swati, Z. N. K., and Akbar, S. (2019). Dp-binder: machine learning model for prediction of dna-binding proteins by fusing evolutionary and physicochemical information. *Journal of Computer-Aided Molecular Design*, 33:645–658.

Ali, F., Arif, M., Khan, Z. U., Kabir, M., Ahmed, S., and Yu, D.-J. (2020). Sdbp-pred: Prediction of single-stranded and double-stranded dna-binding proteins by extending consensus sequence and k-segmentation strategies into pssm. *Analytical biochemistry*, 589:113494.

Ali, F., Kabir, M., Arif, M., Swati, Z. N. K., Khan, Z. U., Ullah, M., and Yu, D.-J. (2018). Dbppred-pdsd: Machine learning approach for prediction of dna-binding proteins using discrete wavelet transform and optimized integrated features space. *Chemometrics and Intelligent Laboratory Systems*, 182:21–30.

Ali, F., Kumar, H., Patil, S., Ahmed, A., Banjar, A., and Daud, A. (2022). Dbp-deepcnn: prediction of dna-binding proteins using wavelet-based denoising and deep learning. *Chemometrics and Intelligent Laboratory Systems*, 229:104639.

Antony, E., Weiland, E. A., Korolev, S., and Lohman, T. M. (2012). Plasmodium falciparum ssb tetramer wraps single-stranded dna with similar topology but opposite polarity to e. coli ssb. *Journal of molecular biology*, 420(4-5):269–283.

Arora, S., Gupta, S., Verma, S., and Malik, I. (2023). Prediction of dna interacting residues. In *2023 International Conference on Computational Intelligence, Communication Technology and Networking (CICTN)*, pages 54–57. IEEE.

Attali, I., Botchan, M. R., and Berger, J. M. (2021). Structural mechanisms for replicating dna in eukaryotes. *Annual review of biochemistry*, 90:77–106.

Banjar, A., Ali, F., Alghushairy, O., and Daud, A. (2022). idbp-pbmd: A machine learning model for detection of dna-binding proteins by extending compression techniques into evolutionary profile. *Chemometrics and Intelligent Laboratory Systems*, 231:104697.

Barukab, O., Ali, F., Alghamdi, W., Bassam, Y., and Khan, S. A. (2022). Dbp-cnn: Deep learning-based prediction of dna-binding proteins by coupling discrete cosine transform with two-dimensional convolutional neural network. *Expert Systems with Applications*, 197:116729.

Dallago, C., Schütze, K., Heinzinger, M., Olenyi, T., Littmann, M., Lu, A. X., Yang, K. K., Min, S., Yoon, S., Morton, J. T., and Rost, B. (2021). Learned embeddings from deep learning to visualize and predict protein sets. *Current Protocols*, 1(5):e113.

Elnaggar, A., Heinzinger, M., Dallago, C., Rihawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. (2020). Prottrans: Towards cracking the language of life's code through self-supervised deep learning and high performance computing.

Fernández, D., Olivera-Nappa, Á., Uribe-Paredes, R., and Medina-Ortiz, D. (2023). Exploring machine learning algorithms and protein language models strategies to develop enzyme classification systems. In *International Work-Conference on Bioinformatics and Biomedical Engineering*, pages 307–319. Springer.

Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. (2012). Cd-hit: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, 28(23):3150–3152.

Gao, Y.-G., Su, S.-Y., Robinson, H., Padmanabhan, S., Lim, L., McCrary, B. S., Edmondson, S. P., Shriver, J. W., and Wang, A. H.-J. (1998). The crystal structure of the hyperthermophile chromosomal protein sso7d bound to dna. *Nature structural biology*, 5(9):782–786.

Guagliardi, A., Cerchia, L., Rossi, M., et al. (2002). The sso7d protein of sulfolobus solfataricus: in vitro relationship among different activities. *Archaea*, 1:87–93.

Gupta, N. K., Wilkinson, E. A., Karuppannan, S. K., Bailey, L., Vilan, A., Zhang, Z., Qi, D.-C., Tadich, A., Tuite, E. M., Pike, A. R., et al. (2021). Role of order in the mechanism of charge transport across single-stranded and double-stranded dna monolayers in tunnel junctions. *Journal of the American Chemical Society*, 143(48):20309–20319.

Hu, S., Ma, R., and Wang, H. (2019). An improved deep learning method for predicting dna-binding proteins based on contextual features in amino acid sequences. *PLoS one*, 14(11):e0225317.

Kabir, A., Bhattarai, M., Rasmussen, K. O., Shehu, A., Bishop, A. R., Alexandrov, B. S., and Usheva, A. (2024). Advancing transcription factor binding site prediction using dna breathing dynamics and sequence transformers via cross attention. *bioRxiv*, pages 2024–01.

Kumar, K. K., Pugalenthi, G., and Suganthan, P. N. (2009). Dna-prot: identification of dna binding proteins from protein sequence information using random forest. *Journal of Biomolecular Structure and Dynamics*, 26(6):679–686.

Lye, Y. S. and Chen, Y.-R. (2022). Tar dna-binding protein 43 oligomers in physiology and pathology. *IUBMB life*, 74(8):794–811.

Ma, X., Guo, J., and Sun, X. (2016). Dnabp: Identification of dna-binding proteins based on feature selection using a random forest and predicting binding residues. *PloS one*, 11(12):e0167345.

McKinney, W. et al. (2011). pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, 14(9):1–9.

Medina, D., Sepulveda-Yanez, J., Alvarez-Saravia, D., Uribe-Paredes, R., Veelken, H., and Navarrete, M. (2023). Artificial intelligence approach for the discovery of autoantigen recognition by b-cell lymphomas. *Blood*, 142:125.

Medina-Ortiz, D., Contreras, S., Amado-Hinojosa, J., Torres-Almonacid, J., Asenjo, J. A., Navarrete, M., and Olivera-Nappa, A. (2020a). Combination of digital signal processing and assembled predictive models facilitates the rational design of proteins. *arXiv preprint arXiv:2010.03516*.

Medina-Ortiz, D., Contreras, S., Amado-Hinojosa, J., Torres-Almonacid, J., Asenjo, J. A., Navarrete, M., and Olivera-Nappa, Á. (2022). Generalized property-based encoders and digital signal processing facilitate predictive tasks in protein engineering. *Frontiers in Molecular Biosciences*, 9.

Medina-Ortiz, D., Contreras, S., Fernández, D., Soto-García, N., Moya, I., Cabas-Mora, G., and Olivera-Nappa, Á. (2024). Protein language models and machine learning facilitate the identification of antimicrobial peptides. *International Journal of Molecular Sciences*, 25(16):8851.

Medina-Ortiz, D., Contreras, S., Quiroz, C., and Olivera-Nappa, Á. (2020b). Development of supervised learning predictive models for highly non-linear biological, biomedical, and general datasets. *Frontiers in molecular biosciences*, 7:13.

Meier, J., Rao, R., Verkuil, R., Liu, J., Sercu, T., and Rives, A. (2021). Language models enable zero-shot prediction of the effects of mutations on protein function. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29287–29303. Curran Associates, Inc.

Mishra, A., Pokhrel, P., and Hoque, M. T. (2019). Stackdppred: a stacking based prediction of dna-binding protein from sequence. *Bioinformatics*, 35(3):433–441.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Rahman, M. S., Shatabda, S., Saha, S., Kaykobad, M., and Rahman, M. S. (2018). Dpp-pseaac: a dna-binding protein prediction model using chou's general pseaac. *Journal of theoretical biology*, 452:22–34.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., and Fergus, R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15).

Roel-Touris, J., Bonvin, A. M., and Jiménez-García, B. (2020). Lightdock goes information-driven. *Bioinformatics*, 36(3):950–952.

Shadab, S., Khan, M. T. A., Neezi, N. A., Adilina, S., and Shatabda, S. (2020). Deepdbp: deep neural networks for identification of dna-binding proteins. *Informatics in Medicine Unlocked*, 19:100318.

Sharma, R., Kumar, S., Tsunoda, T., Kumarevel, T., and Sharma, A. (2021). Single-stranded and double-stranded dna-binding protein prediction using hmm profiles. *Analytical biochemistry*, 612:113954.

Tan, C., Wang, T., Yang, W., and Deng, L. (2019). Predpsd: a gradient tree boosting approach for single-stranded and double-stranded dna binding protein prediction. *Molecules*, 25(1):98.

Wang, W., Sun, L., Zhang, S., Zhang, H., Shi, J., Xu, T., and Li, K. (2017). Analysis and prediction of single-stranded and double-stranded dna binding proteins based on protein sequences. *BMC bioinformatics*, 18:1–10.

Wang, Y., Zhang, L., Huang, T., Wu, G.-R., Zhou, Q., Wang, F.-X., Chen, L.-M., Sun, F., Lv, Y., Xiong, F., et al. (2022a). The methyl-cpg-binding domain 2 facilitates pulmonary fibrosis by orchestrating fibroblast to myofibroblast differentiation. *European Respiratory Journal*, 60(3).

Wang, Z., Gong, M., Liu, Y., Xiong, S., Wang, M., Zhou, J., and Zhang, Y. (2022b). Towards a better understanding of tf-dna binding prediction from genomic features. *Computers in Biology and Medicine*, 149:105993.

Werner, M. H., Huth, J. R., Gronenborn, A. M., and Clore, G. M. (1995). Molecular basis of human 46x, y sex reversal revealed from the three-dimensional solution structure of the human sry-dna complex. *Cell*, 81(5):705–714.

Zaman, R., Chowdhury, S. Y., Rashid, M. A., Sharma, A., Dehzangi, A., Shatabda, S., et al. (2017). Hmmbinder: Dna-binding protein prediction using hmm profile based features. *BioMed research international*, 2017.

Zhang, J., Chen, Q., and Liu, B. (2020). idrbp_mmc: identifying dna-binding proteins and rna-binding proteins based on multi-label learning model and motif-based convolutional neural network. *Journal of molecular biology*, 432(22):5860–5875.

Zhang, Q., Liu, P., Wang, X., Zhang, Y., Han, Y., and Yu, B. (2021). Stackpdb: predicting dna-binding proteins based on xgb-rfe feature optimization and stacked ensemble classifier. *Applied Soft Computing*, 99:106921.

Zhang, Y., Bao, W., Cao, Y., Cong, H., Chen, B., and Chen, Y. (2022). A survey on protein–dna-binding sites in computational biology. *Briefings in Functional Genomics*, 21(5):357–375.

APPENDIX

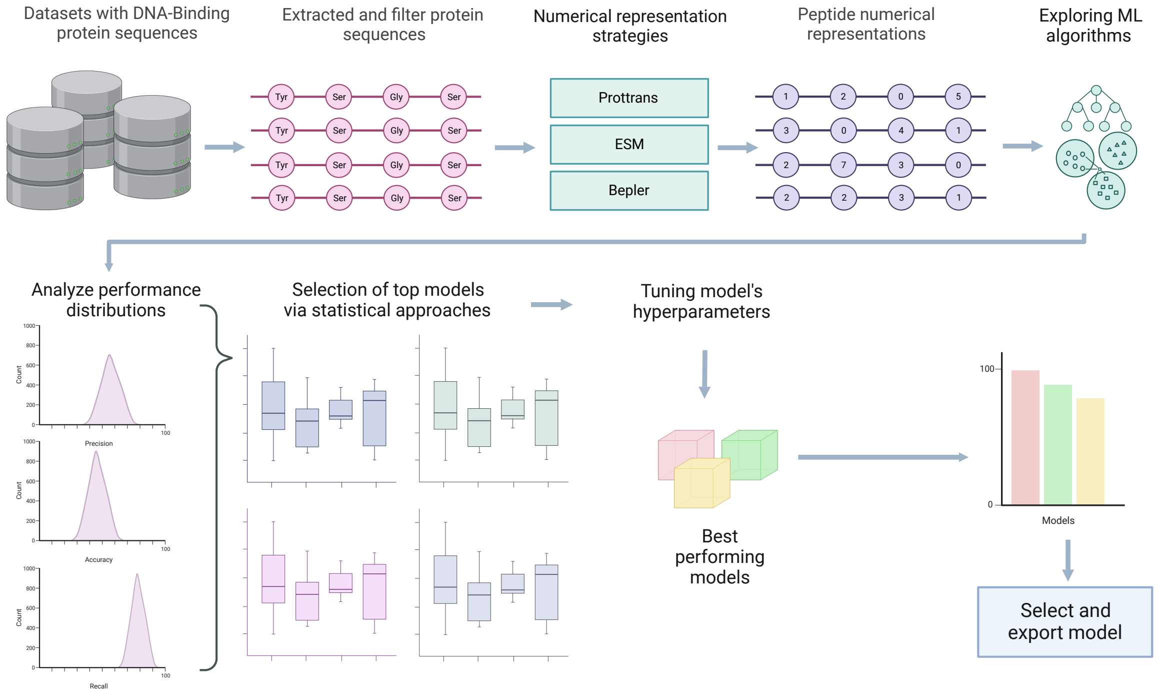

Datasets with DNA-Binding protein sequences
Extracted and filter protein sequences
Tyr
Ser
Gly
Numerical representation strategies
Prottrans
ESM
Bepler
Peptide numerical representations
Exploring ML algorithms
Analyze performance distributions
Count
Precision
Accuracy
Recall
Selection of top models via statistical approaches
Tuning model's hyperparameters
Best performing models
100
0
Models
Select and export model

Figure 3: **The designed e implemented pipeline to train predictive models for DNA-Binding identification incorporated in RUDEUS**. The proposed pipeline first collects and processes the protein sequences by incorporating length filters and removing non-canonical residues. Then, numerical representation strategies are applied to obtain encoded vectors through pre-trained models based on protein language models, including Prottrans family models, ESM family models, Bepler, Glove, and all the different pre-trained models available in the bio-embedding library. Then, different supervised learning algorithms are explored using default hyperparameters employing all generated datasets in the previous step. Then, statistical approaches are applied to filter and select the best combinations of supervised learning algorithms and numerical representation approaches. A Bayesian approach guides the selected combinations tuning hyperparameters process through the Optuna library, and ensemble learning is explored to evaluate different combinations of the individual optimized models. Finally, the best strategy is selected based on the best performances, including training, validation, and overfitting ratio.

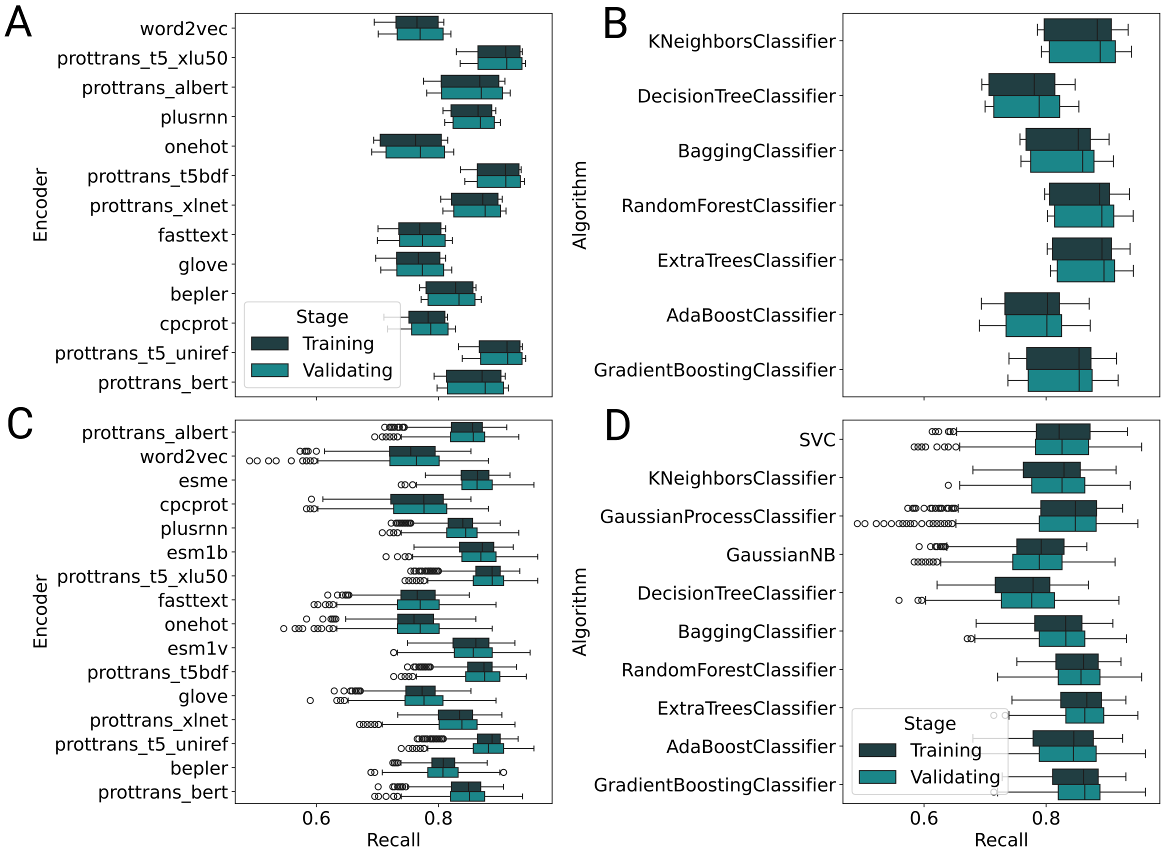


Figure 4: **Recall distribution performances for all explored tasks in this work evaluated by numerical representation strategies and supervised learning algorithms**. **A** Recall distribution for DNA-binding classification task grouped by pre-trained model employed as numerical representation strategy. **B** Recall distribution for DNA-binding classification task grouped by supervised learning algorithm. **C** Recall distribution for single-stranded or double-stranded DNA type interaction task grouped by pre-trained model employed as numerical representation strategy. **D** Recall distribution for single-stranded or double-stranded DNA type interaction task grouped by supervised learning algorithm.

Knowledge Discovery in Optical Music Recognition: Enhancing Information Retrieval with Instance Segmentation

Elona Shatri[a] and György Fazekas[b]
Queen Mary University of London, London, U.K.
{e.shatri, g.fazekas}@qmul.ac.uk

Keywords: OMR, Instance Segmentation, Dense Objects.

Abstract: Optical Music Recognition (OMR) automates the transcription of musical notation from images into machine-readable formats like MusicXML, MEI, or MIDI, significantly reducing the costs and time of manual transcription. This study explores knowledge discovery in OMR by applying instance segmentation using Mask R-CNN to enhance the detection and delineation of musical symbols in sheet music. Unlike Optical Character Recognition (OCR), OMR must handle the intricate semantics of Common Western Music Notation (CWMN), where symbol meanings depend on shape, position, and context. Our approach leverages instance segmentation to manage the density and overlap of musical symbols, facilitating more precise information retrieval from music scores. Evaluations on the DoReMi and MUSCIMA++ datasets demonstrate substantial improvements, with our method achieving a mean Average Precision (mAP) of up to 59.70% in dense symbol environments, achieving comparable results to object detection. Furthermore, using traditional computer vision techniques, we add a parallel step for staff detection to infer the pitch for the recognised symbols. This study emphasises the role of pixel-wise segmentation in advancing accurate music symbol recognition, contributing to knowledge discovery in OMR. Our findings indicate that instance segmentation provides more precise representations of musical symbols, particularly in densely populated scores, advancing OMR technology. We make our implementation, pre-processing scripts, trained models, and evaluation results publicly available to support further research and development.

1 INTRODUCTION

Optical Music Recognition (OMR) is a research subfield within Music Information Retrieval that automates the transcription of music notation from images into machine-readable formats such as MusicXML (Good, 2001), MEI (Roland, 2002), or MIDI[1]. This automation addresses the significant time and cost involved in manually transcribing music scores, a process essential for digital music analysis, editing, and playback. Beyond automation, OMR has profound implications for knowledge discovery and information retrieval within large music databases. By converting analogue musical scores into searchable digital formats, OMR enables researchers, educators, and musicians to efficiently explore vast collections of musical works, facilitating musicological research, comparative studies, and the discovery of patterns and trends across different musical eras and styles. These capabilities underscore OMR's critical role in advancing music information retrieval and digital humanities.

While OMR is often compared to Optical Character Recognition (OCR), which transcribes printed text into digital form, OMR faces unique complexities due to the multifaceted nature of Common Western Music Notation (CWMN). In music scores, the meaning of symbols depends not only on their shapes but also on their precise positions on the staff and their contextual relationships with other symbols. For example, a note's value and pitch are determined by its position on the staff and its interaction with key signatures, time signatures, and other musical elements. Consequently, accurate staff detection is crucial, so our approach incorporates a parallel step using traditional computer vision techniques to address this challenge, as detailed in Subsection 4.3. These intricacies present challenges that standard OCR methods, primarily designed for straightforward text recognition, cannot handle. Therefore, specialised techniques are required to interpret and digitise musical notation ac-

[a] https://orcid.org/0000-0002-1651-5848
[b] https://orcid.org/0000-0003-2580-0007
[1] https://www.midi.org

Shatri, E. and Fazekas, G.
Knowledge Discovery in Optical Music Recognition: Enhancing Information Retrieval with Instance Segmentation.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 311-319
ISBN: 978-989-758-716-0; ISSN: 2184-3228

curately.

Recent advances in deep learning, particularly in the use of Convolutional Neural Networks (CNN) (Pacha et al., 2018; Pacha and Calvo-Zaragoza, 2018), Recurrent Neural Networks (RNN) (Baró et al., 2018), and more recently, transformer-based models (Li et al., 2023; Ríos-Vila et al., 2024a; Ríos-Vila et al., 2024b), have significantly enhanced the capabilities of OMR systems. CNNs excel at recognising patterns and features within images, making them well-suited for identifying and distinguishing musical symbols. RNNs, in contrast, are adept at handling sequential data, enabling them to interpret the temporal and contextual relationships between musical elements. Together, these technologies have improved the accuracy of symbol detection and interpretation in OMR. Despite these advances, significant limitations remain, particularly when dealing with densely packed and overlapping symbols, which can result in errors in detection and interpretation (Pacha and Eidenberger, 2017).

Our study compares object detection with instance segmentation using Mask R-CNN (He et al., 2017), integrating a parallel staff detection stage for pitch inference. This step addresses the challenge of recognising thin, elongated staff lines, which detection methods often struggle with. Instance segmentation enables pixel-level classification to precisely delineate musical symbols, particularly in dense and overlapping notation, enhancing OMR accuracy and supporting reliable information extraction from music scores.

We also emphasise the importance of significantly improving performance by domain-specific pre-training, such as using MUSCIMA++ weights. Our focus on full-page music symbol recognition addresses the complexities of entire music sheets, distinguishing our approach from methods that only handle isolated symbols.

By improving musical symbol recognition, our method facilitates knowledge discovery and information retrieval in large music databases through enhanced metadata extraction, enabling advanced queries and comparative musicology. We evaluate our approach using the DoReMi and MUSCIMA++ datasets and provide our implementation details here here [2].

Our study shows that using models like Mask R-CNN for instance segmentation, along with efficient staff detection algorithms, advances OMR for full-page images. These models are less data-hungry than transformer-based approaches, making them more practical for OMR, where annotated datasets are limited. This combination provides a more accurate and efficient solution for digitising musical scores, supporting enhanced music analysis, retrieval, and knowledge discovery.

2 RELATED WORK

OMR has traditionally been approached through four stages: image pre-processing, musical object detection, reconstruction, and encoding (Rebelo et al., 2012; Shatri and Fazekas, 2020). The primary objective of musical symbol detection is to find the bounding boxes and corresponding classes of musical objects. Undetected musical primitives in this stage can introduce errors into subsequent stages, making detection a critical component of OMR. This is particularly important due to the complexities introduced by artefacts, image quality, and object density. The classified primitive elements are then integrated using graphical or syntactic rules, or more recently, deep learning, to reconstruct the musical notation in the third stage. The final encoding stage converts the reconstructed notation into a machine-readable format mentioned in Section 1.

Traditional object detection in OMR has seen the application of models like Region-based CNNs (R-CNNs) (Jiao et al., 2019), which integrate region proposals with CNNs. Despite their effectiveness in certain domains, these models face limitations in OMR due to their separate training stages and the challenges of handling densely packed musical symbols. Fast R-CNN and Faster R-CNN (Girshick, 2015; Ren et al., 2015) addressed some of these issues by introducing a Region Proposal Network (RPN) that generates region proposals more efficiently, sharing convolution features with the detection network. However, while effectively reducing computational costs and improving detection accuracy, these models still struggle with full-page musical scores and densely populated notation (Pacha and Calvo-Zaragoza, 2018).

Beyond object detection, semantic segmentation has been explored to provide pixel-level classification of similar entities within images (Long et al., 2015; Zhao et al., 2019). Techniques like the Deep Watershed Detector (Tuggener et al., 2018) and U-Net architectures (Jr et al., 2018) have been employed to improve the detection of smaller musical objects and overlapping symbols. However, these methods also have limitations. The Deep Watershed Detector struggles with larger musical objects and rare classes due to its size variation learning limitations. At the same time, U-Net models are sensitive to training data distribution and may lose translation equivalence due

[2] https://github.com/elonashatri/pitch_mask_rcnn

to their down-sampling and up-sampling processes (Johnson and Khoshgoftaar, 2019; Ronneberger et al., 2015). Instance segmentation models, such as Mask R-CNN (He et al., 2017), offers a more refined approach by providing pixel-level classification for each object, allowing for the precise identification and separation of overlapping symbols. This method extends Faster R-CNN by adding a branch that predicts segmentation masks on each Region of Interest (RoI), using RoI Align to avoid the quantisation issues seen in earlier models. By generating pixel-wise masks, instance segmentation improves the detection and analysis of musical symbols and facilitates the exclusion of noise and more accurate pitch identification relative to staff lines. Mask R-CNN has been applied to handwritten 4-part harmonies (De Vega et al., 2022) showing promising results in these types of scores.

Despite the advancements brought by instance segmentation, challenges remain, particularly with class imbalance in musical notation datasets and the variability in handwritten music. Certain symbols, such as noteheads and stems, are more prevalent, skewing the training process. Additionally, detecting thin, closely spaced objects like staff lines and barlines poses significant difficulties.

Our study addresses these challenges by implementing and comparing object detection and instance segmentation techniques for OMR using the DoReMi and MUSCIMA++ datasets. We demonstrate the effectiveness of Mask R-CNN in handling dense and overlapping symbols and provide a comprehensive comparative evaluation highlighting the advantages of instance segmentation over traditional methods. By integrating a parallel staff detection stage, our approach enhances the accuracy and detail of OMR systems and offers a more comprehensive solution for digitising musical notation. Implementation details and evaluation results will be publicly available.

3 DATASETS

This study utilises two prominent datasets in the domain of OMR: MUSCIMA++ (Hajič and Pecina, 2017) and DoReMi (Shatri and Fazekas, 2021). Both datasets play crucial roles in training and evaluating our models, offering diverse challenges and benefits.

MUSCIMA++. The MUSCIMA++ dataset consists of handwritten musical scores and is specifically designed to address the challenges of handwritten OMR. It provides a substantial number of annotated images with detailed metadata files that include bounding boxes and pixel masks for each musical symbol. This dataset is particularly valuable for its representation of handwritten music, which introduces variability and complexity not found in typeset scores.

DoReMi. The DoReMi dataset comprises approximately 6,400 high-resolution images (300 DPI, 2475x3504 pixels) of typeset sheet music, annotated with one of 94 category labels. This dataset is notably larger than MUSCIMA++, making it a robust resource for training deep learning models. The high resolution and detailed annotations allow for precise training and evaluation of models aimed at both object detection and instance segmentation. Both datasets exhibit class imbalance, with a significant portion of the annotated objects being stems and noteheads. This imbalance presents a challenge for training models that need to recognise a diverse set of musical symbols.

The DoReMi dataset is primarily used for training our models due to its larger size and the fact that it consists of typeset images, which are more standardised than handwritten scores. In our object detection experiments, we train different models using varying subsets of the DoReMi dataset: 27%, 45%, 90%, and 100% of the data. This stratified approach serves two key purposes:

- It helps identify the most effective amount of data required for training without overfitting.
- Explore how class balance affects the detection rate by avoiding the predominance of infrequent classes that could skew the training process.

The resulting datasets used for training consist of 94, 71, 71, and 64 classes, respectively. For instance segmentation, we further refine our subsets to include 291 and 1685 images from DoReMi, focusing on limiting computational time during training while maintaining a sufficient number of classes to evaluate model performance effectively. By iteratively refining the training process and expanding the dataset, this study aims to identify key factors that may improve the accuracy and reliability of segmenting musical symbols in sheet music.

4 METHODOLOGY

This section discusses the detailed methodology, covering object detection, instance segmentation, data pre-processing, and training configurations. Our approach leverages full-page images, encompassing more objects with smaller bounding boxes than staff-cropped images. This approach enhances the gran-

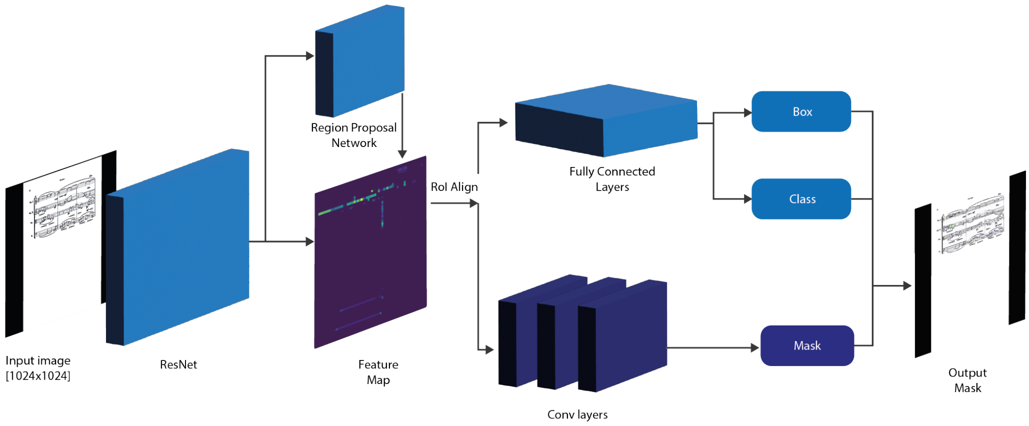


Figure 1: Mask R-CNN architectures applied to OMR.

ularity and accuracy of detection and segmentation within the full-page images.

We conduct two experiments using the DoReMi dataset: one on detecting music notation primitives and another on instance segmentation. Object detection predicts a bounding box (x_1, x_2, y_1, y_2), an associated category, and a confidence score for each musical element, such as stems, noteheads, or dynamics (e.g., *ppp*). Both pipelines include a parallel step for staff detection using a traditional method described in Section 4.3.

4.1 Music Symbol Object Detection

For the object detection task, we employ Faster R-CNN, a well-established model in object detection. Faster R-CNN integrates a Region Proposal Network (RPN) with a Fast R-CNN detector, allowing for efficient and accurate object localisation and classification. The loss function for Faster R-CNN is defined as:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{\text{reg}}}\sum_i p_i^* L_{reg}(t_i, t_i^*), \tag{1}$$

where i is the index of an anchor in a mini-batch and p_i is the predicted probability of anchor i being an object. We use two losses, classification loss L_{cls} and regression loss L_{reg}. Classification loss is log loss over two classes (object vs not object). Regression loss is activated only for positive anchors $p_i^* L_{reg}$ and not activated otherwise. Both terms are then normalised by N_{cls} and N_{reg} and weighted by λ as a balancing parameter. Every bounding-box has the associated set of cells in the feature map computed by a CNN. Our Faster R-CNN implementation follows prior work from (Ren et al., 2015; Pacha et al., 2018).

The Faster R-CNN model configurations employ ResNet50 or Inception ResNet v2 (Szegedy et al., 2017) backbones with Atrous convolutions, optimised for the MUSCIMA++ or COCO (Lin et al., 2014) datasets with image dimensions of 2475×3504 pixels. The model maintains aspect ratios, with image dimensions between 500 and 1000 pixels. The feature extractor uses a stride of 8 in the first stage, and anchor generation is handled by a grid anchor generator with various widths, heights, scales, and aspect ratios. The first-stage Atrous rate is 2, with box predictor parameters using an L2 regulariser and a truncated normal initialiser. Non-maximum suppression (NMS) has an IoU threshold of 0.5 with up to 500 proposals. The initial crop size is 17, and max-pooling has a kernel size and stride of 1. The model is trained with a batch size of 1 using an RMSProp optimiser (initial learning rate of 0.003, decaying by 0.95 every 30,000 steps), with momentum at 0.9 and gradient clipping at 10.0. Data augmentation includes random horizontal flipping, and evaluation metrics follow the COCO standard, using 120 examples for evaluation.

4.2 Music Symbol Instance Segmentation

For instance segmentation, we utilise Mask R-CNN (He et al., 2017), which extends Faster R-CNN by adding a branch for predicting segmentation masks on each Region of Interest (RoI), as shown in Figure 1). This method allows for pixel-level classification of objects, which is crucial for accurately delineating overlapping musical symbols. We define the task of instance segmentation as follows:

Definition 4.1. Music Symbol Instance Segmentation (MSIS) is the task of assigning a music symbol class to each pixel of a sheet music image in addition to predicting bounding boxes with their corresponding confidence scores.

The loss function in instance segmentation is given as:

$$\begin{aligned} TotalLoss = {} & rpn\_class\_loss + rpn\_bbox\_loss \\ & + mrcnn\_class\_loss + mrcnn\_bbox\_loss \\ & + mrcnn\_mask\_loss \end{aligned} \tag{2}$$

Here, the total loss is a combination of several components: RPN Class Loss, which measures the error in classifying the anchors; RPN Bounding Box Loss, which quantifies the error in bounding box predictions by the RPN; Mask R-CNN Class Loss, which evaluates the error in classifying objects within the proposed regions (RoIs); Mask R-CNN Bounding Box Loss, which measures the error in bounding box predictions for the RoIs; and Mask R-CNN Mask Loss, which evaluates the accuracy of the predicted segmentation masks for each RoI. These components collectively ensure accurate detection and segmentation of musical symbols in sheet music images.

We utilise two backbone networks, ResNet50 and ResNet101 (He et al., 2016), which are pre-trained either on the COCO dataset or a subset of the DoReMi dataset. The training process is structured in multiple phases to enhance model performance iteratively. Initially, the model is trained on a dataset of 291 images with weights pre-trained on COCO, focusing only on the head layers for 20 epochs. Subsequently, training is extended to all layers for an additional ten epochs using the same dataset. To further improve performance, the model is trained on an expanded dataset of 1,348 images for another 40 epochs, fine-tuning the weights from the previous phase.

For the best model, the ResNet101 backbone network has strides of [4, 8, 16, 32, 64] and a batch size of 1. Detection parameters included a maximum of 100 instances, a minimum confidence of 0.9, and an NMS threshold of 0.3. The FPN classification fully connected layer size is 1024. The gradient clipping norm is set to 5.0, and we used one image per GPU. The image dimensions are resized to a square shape of [1024, 1024, 3], and the mini mask shape is (56, 56). The learning momentum is 0.9, with a learning rate of 0.0001. The dataset comprised 72 classes. The RoI positive ratio is 0.33, and RPN anchor ratios were [0.5, 1, 2] with scales (32, 64, 128, 256, 512) and stride 1. RPN bounding box standard deviation was [0.1, 0.1, 0.2, 0.2] with an NMS threshold of 0.7 and 256 RPN train anchors per image. Training is re-initiated from epoch 39, using pre-trained weights from the specified path with a learning rate of 0.0001, and the network was trained on all layers.

The experimental setup and results are detailed in Table 2. Initially, the model training started on a relatively small dataset comprising 291 images, focusing exclusively on the network's head layer, employing pre-trained weights for initialisation (detailed in Table 2, 3rd row). Two subsequent stages of fine-tuning followed this phase:

- After the initial focus on the head layer, training was extended to all network layers for an additional ten epochs. This stage aimed to refine the feature extraction capabilities across the entire network.
- To further examine the effects of dataset size on segmentation performance, the model underwent additional training on an expanded dataset of 1,348 images spanning 50 epochs. This stage evaluated how larger datasets influence the model's ability to generalise and improve segmentation accuracy.

This iterative training approach is structured to assess the effectiveness of pre-trained models and explore the potential of increasing dataset sizes to enhance music symbol segmentation accuracy, a notable challenge in OMR.

4.3 Staff Detection

The proposed methodology for detecting staff lines in musical notation involves several key steps. Initially, the input image is converted to grayscale and enhanced using Otsu's thresholding method to create a binary image. Horizontal lines are then detected through morphological operations. Contours representing potential staff lines are identified and analysed based on their geometric properties. Specifically, contours with a high aspect ratio and appropriate height are classified as staff lines, while others are marked as non-staff lines.

The large contours are divided into smaller segments to manage those that span multiple staff lines. The identified staff lines are then drawn on the image for visualisation (see Figure 2 and 3), using different colours to differentiate between staff lines (in green) and non-staff lines (in red). The final processed image is displayed alongside the original for clear comparison in Figures 2 and 3. This method aims to detect staff lines in musical notation, making it easier to output a more structured result by stave and enabling pitch or staffing position information to be captured. However, it is important to note that this approach may not work well with images of low quality. Still, it is robust enough to handle small perspective shifts, rotations, changes in contrast, and similar variations.

Table 1: Object detection results using DoReMi dataset, '*' denotes COCO weights used for initialisation and '+' denotes MUSCIMA++ weights. Mean Average Precision (mAP) given at 0.50 IoU (%). The Large, Medium and Small columns show the mAP for large, medium and small objects respectively. The last two rows present experimental results using InceptionResNetV2 in CollabScore63, a variant of DeepScores with a reduced number of classes, utilising a training set of 1,362 images, alongside their proposed Cascade R-CNN - FocalNet model with 136 classes (Yesilkanat et al., 2023).

Model	Classes	Data (%)	Steps	Large	Medium	Small	mAP
InceptionResNetV2*	94	100	120K	32.53	38.24	15.46	47.37
InceptionResNetV2*	71	90	120K	36.79	47.42	25.55	63.45
InceptionResNetV2*	71	45	120K	33.89	54.08	22.38	65.42
InceptionResNetV2*	67	27	120K	40.97	46.45	27.13	63.70
InceptionResNetV2+	71	90	120K	41.08	49.93	23.18	64.92
ResNet50*	71	90	120K	36.73	45.22	28.98	59.81
ResNet50*	71	45	120K	32.79	46.03	31.76	62.45
ResNet50*	67	27	120K	39.32	46.64	29.62	63.19
ResNet50+	71	90	120K	35.90	44.57	29.17	63.13
ResNet50*	94	100	80K	**93.63**	**75.82**	**41.71**	**80.99**
InceptionResNetV2†	63	CS63	-	-	-	-	**64.1**
CascadeRCNN†	136	DS	-	-	-	-	**70.0**

Table 2: Performance metrics of various configurations of the ResNet model using the Mask R-CNN architecture.

Feature Extractor	Weights	No. of Images	Epochs	Layers	mAP@.50%
ResNet50	COCO	291	20	Heads	16.239
ResNet101	COCO	291	30	Heads	37.151
ResNet101	DoReMi	291	30 + 10	All	46.087
ResNet101	**DoReMi**	1384	**80**	**All**	**59.70**

Another limitation is that this method is based on the relative distance of the objects to the detected staff lines. The method may provide the wrong stave for complex scores, where musical objects such as noteheads are closer to the next stave than their own staff. This can be addressed using CNN trained to distinguish staves (Pacha, 2019).

This method has been applied in parallel with both instance segmentation and object detection models since they struggle with detecting thin, long objects such as staff lines.

5 EVALUATION AND RESULTS

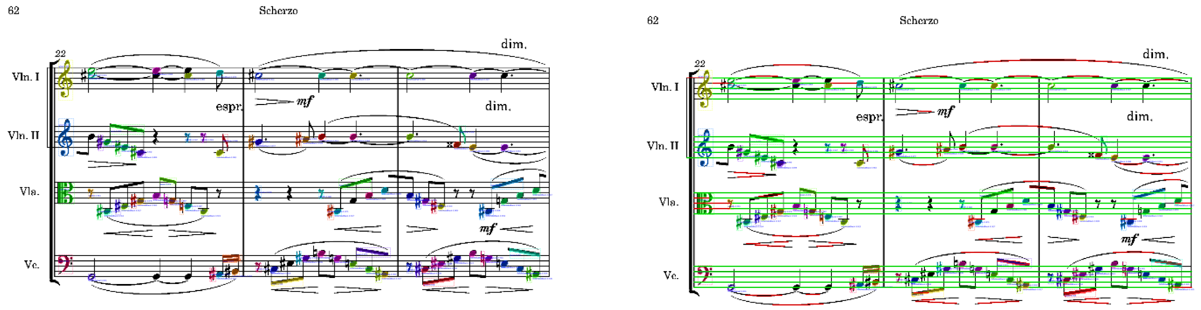
A key contribution of this study is the comparative analysis of object detection and instance segmentation methods for OMR. By evaluating both approaches, we aim to determine which method is better suited to the complexities of musical notation, particularly in handling dense and overlapping symbols. Object detection offers efficient symbol localisation but lacks the pixel-level precision provided by instance segmentation. Our comparison highlights the strengths and weaknesses of each method, giving valuable insights into their applicability in different OMR scenarios.

In this section, we detail the experiments conducted to evaluate the performance of the instance segmentation approach using Mask R-CNN, compared to traditional object detection methods. We used the DoReMi and MUSCIMA++ datasets, which offer diverse challenges due to their handwritten and typeset musical scores, respectively.

The performance of our object detection and instance segmentation models was assessed using the mAP as is standard in the field (Lin et al., 2014). The mAP scores were computed by setting a threshold for Intersection over Union (IoU) to evaluate the accuracy of the predicted bounding boxes against the ground truth.

The trade-off between precision and recall is particularly significant in the domain of OMR, where precise localisation of overlapping musical symbols is crucial. Precision measures the accuracy of the predictions, whereas recall assesses the model's ability to detect all relevant instances. These metrics form the basis of our evaluation using the Average Precision (AP) for each class at an IoU threshold of 0.50.

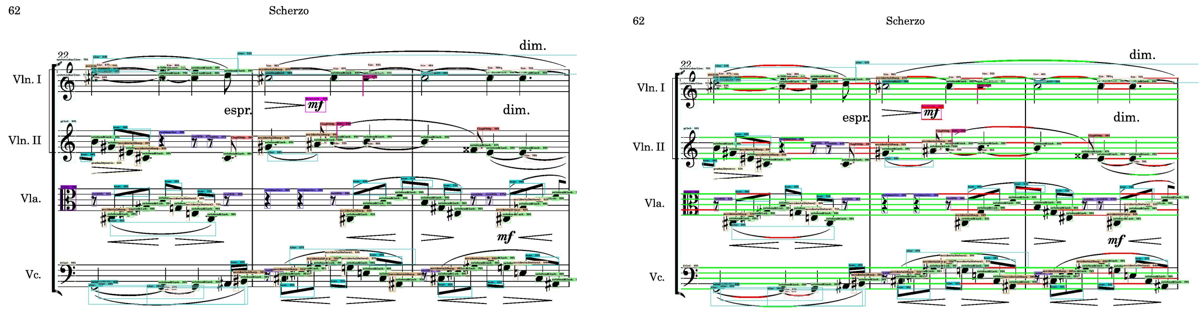
Musical Symbol Object Detection. We first applied Faster R-CNN with both InceptionResNetV2 and ResNet50 backbones for object detection tasks. The evaluation metrics were based on the mean Average Precision (mAP) at an Intersection over Union (IoU) threshold of 0.50. The results are summarised



(a) Output from the Mask R-CNN model, each class is represented by a different colour

(b) Concatenated Mask R-CNN output and staff detection, the green lines represent the detected staff lines

Figure 2: Mask R-CNN model inference in an image.



(a) Output from the Faster R-CNN model, each class is represented by a different colour

(b) Concatenated Faster R-CNN output and staff detection, the green lines represent the detected staff lines

Figure 3: Faster R-CNN model inference in an image.

in Table 1. Using the InceptionResNetV2 backbone, we found that reducing the number of classes from 94 to 71 significantly improved mAP, with a peak mAP of 65.42% achieved with 45% of the data. Interestingly, using 90% of the data resulted in a slightly lower mAP of 64.92%, suggesting that a well-balanced and smaller dataset can sometimes outperform a larger one. Moreover, using domain-specific pre-trained weights from MUSCIMA++ provided nearly a 5% improvement over COCO weights, underscoring the importance of domain adaptation.

Similarly, the ResNet50 backbone demonstrated notable performance, achieving a peak mAP of 63.19% using only 27% of the data. This result highlights the potential efficiency of a well-curated dataset. MUSCIMA++ weights also improved performance, although not as significantly as InceptionResNetV2. These results suggest that while both models are compelling, InceptionResNetV2 outperforms ResNet50, particularly with smaller training sets. However, this dynamic shifts when a larger dataset is used, with ResNet50* achieving an mAP of 80.99% on the full dataset. This is comparable to similar models in OMR (Yesilkanat et al., 2023), which, in contrast, uses high-resolution input images, shown in the last two rows on Table 1. Finally, an investigation into average precision per category revealed that the model struggles to detect less frequent objects, such as rarely used time signatures, dynamics markers, and accidentals, as evident in the inference results shown in Figure 3.

Musical Symbol Instance Segmentation. For instance segmentation, we employed Mask R-CNN with ResNet50 and ResNet101 backbones and evaluated using the Pascal VOC metric (Everingham et al., 2010). The results, detailed in Table 2, demonstrate significant performance improvements. Initially, using the ResNet50 backbone and training on 291 images with COCO pre-trained weights, the model achieved an mAP of 16.239%. Further fine-tuning on the DoReMi dataset for 30 epochs increased the mAP to 46.087%. In contrast, the ResNet101 backbone, when trained on the same 291 images, reached an mAP of 37.151%. Extending the training to 1,384 images for 80 epochs with ResNet101 achieved the highest mAP of 59.70%, demonstrating the benefits of a more extensive and more comprehensive dataset. These results are similar to the mAP achieved by object detection models shown in Table 1 using a similar training set size. Findings from this evaluation indicate that:

- Switching to a more complex network architecture (ResNet101) and training all layers comprehensively significantly improved mAP scores, confirming the architecture's influence on performance.
- Extending training duration and increasing dataset size were crucial for achieving higher mAP scores. This is especially evident in the performance of ResNet101 when trained for 80 epochs on 1,384 images.
- The model faced difficulties predicting thin objects such as staff lines and stems, likely due to downsampling issues. Adjustments in RPN anchor ratios or pre-segmentation strategies might mitigate these challenges.

Beyond Detection and Segmentation As stated in Section 1, the ultimate aim of OMR is to convert music scores into a structured, machine-readable file. While object detection and instance segmentation perform well, their output is somewhat unstructured and lacks musical significance. Therefore, a parallel step is necessary to achieve a more organised format. Given the challenges in accurately detecting thin, elongated objects such as staff lines, which are crucial for determining the pitch of a note, we have opted for a more efficient post-processing approach using traditional computer vision techniques that do not require training. In Figure 2a, you can observe the results of running the Mask R-CNN model on a score image in the top image, and the same image with staff detection in the bottom Figure 2b, similarly for the object detection task shown in Figure 3.

6 CONCLUSIONS

This study uses advanced neural network architectures to evaluate object detection and instance segmentation for OMR. Our findings confirm that instance segmentation, particularly using Mask R-CNN, offers capabilities in delineating detailed and accurate representations of individual musical symbols compared to traditional object detection methods. Mask R-CNN's detailed pixel-level segmentation capability makes it particularly adept at handling the complex visual compositions of sheet music where objects frequently overlap.

By comparing object detection and instance segmentation, we comprehensively evaluate their respective performances in OMR. This comparison is significant as it informs the selection of the most appropriate method depending on the specific requirements of a given task—whether broad localisation of symbols is sufficient or if precise delineation is necessary. Our findings contribute to a deeper understanding of how these techniques can be applied to optimise OMR systems.

One limitation of our approach is handling rare musical symbols, which are underrepresented in the training dataset. This could affect the generalisability of the model. Further research should focus on refining the detection of infrequent musical symbols and enhancing mask predictions for structurally complex objects. Prospective improvements could include optimising RPN anchor ratios and experimenting with larger training datasets to improve detection accuracy.

This study highlights the potential of instance segmentation to advance OMR research, offering a more accurate and detailed method for recognising musical symbols in sheet music. These findings pave the way for further research and practical applications in musicology and digital archiving. Improved precision in symbol recognition directly impacts downstream tasks in knowledge discovery and information retrieval, enabling richer metadata generation for indexing and querying in music information retrieval systems. This facilitates sophisticated analysis, such as identifying patterns across compositions, discovering relationships between works, and conducting large-scale comparative studies. The detailed segmentation provided by Mask R-CNN supports more granular analysis, leading to deeper insights into musical structures and trends.

ACKNOWLEDGEMENTS

The authors acknowledge the support of the AI and Music CDT, funded by UKRI and EPSRC under grant agreement no. EP/S022694/1, and our industry partner Steinberg Media Technologies GmbH for their continuous support.

REFERENCES

Baró, A., Riba, P., Calvo-Zaragoza, J., and Fornés, A. (2018). Optical music recognition by long short-term memory networks. In *Graphics Recognition. Current Trends and Evolutions: 12th IAPR International Workshop, GREC 2017, Kyoto, Japan, November 9-*

10, 2017, Revised Selected Papers 12, pages 81–95. Springer.

De Vega, F. F., Alvarado, J., and Cortez, J. V. (2022). Optical music recognition and deep learning: An application to 4-part harmony. In *2022 IEEE Congress on Evolutionary Computation (CEC)*, pages 01–07. IEEE.

Everingham, M. et al. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*.

Good, M. (2001). Musicxml: An internet-friendly format for sheet music. In *XML Conference and Expo*.

Hajič, J. and Pecina, P. (2017). The muscima++ dataset for handwritten optical music recognition. In *14th IAPR ICDAR*. IEEE.

He, K. et al. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

He, K. et al. (2017). Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*.

Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., and Qu, R. (2019). A survey of deep learning-based object detection. *IEEE Access*.

Johnson, J. M. and Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*.

Jr, J. H., Dorfer, M., Widmer, G., and Pecina, P. (2018). Towards full-pipeline handwritten omr with musical symbol detection by u-nets. In *ISMIR*.

Li, Y., Liu, H., Jin, Q., Cai, M., and Li, P. (2023). Tromr: Transformer-based polyphonic optical music recognition. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Lin, T.-Y. et al. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision*. Springer.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of CVPR*.

Pacha, A. (2019). Incremental supervised staff detection. In *Proceedings of the 2nd international workshop on reading music systems*, pages 16–20.

Pacha, A. and Calvo-Zaragoza, J. (2018). Optical music recognition in mensural notation with region-based convolutional neural networks. In *ISMIR*, pages 240–247.

Pacha, A., Choi, K., Couasnon, B., Ricquebourg, Y., Zanibbi, R., and Eidenberger, H. (2018). Handwritten music object detection: Open issues and baseline results. In *13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 163–168. IEEE.

Pacha, A. and Eidenberger, H. (2017). Towards self-learning optical music recognition. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 795–800. IEEE.

Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marcal, A. R. S., Guedes, C., and Cardoso, J. S. (2012). Optical music recognition: State-of-the-art and open issues. *International Journal of Music Information Retrieval (IJMIR)*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Ríos-Vila, A., Calvo-Zaragoza, J., and Paquet, T. (2024a). Sheet music transformer: End-to-end optical music recognition beyond monophonic transcription. *arXiv preprint arXiv:2402.07596*.

Ríos-Vila, A., Calvo-Zaragoza, J., Rizo, D., and Paquet, T. (2024b). Sheet music transformer++: End-to-end full-page optical music recognition for pianoform sheet music. *arXiv preprint arXiv:2405.12105*.

Roland, P. (2002). The music encoding initiative (mei). In *Proceedings of the First International Conference on Musical Applications Using XML*.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer.

Shatri, E. and Fazekas, G. (2020). Optical music recognition: State of the art and major challenges. In *Proceedings of TENOR'20/21*, Hamburg, Germany. Hamburg University for Music and Theater.

Shatri, E. and Fazekas, G. (2021). Doremi: First glance at a universal omr dataset. In *Proceedings of the 3rd WoRMS*.

Szegedy, C. et al. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Tuggener, L., Elezi, I., Schmidhuber, J., and Stadelmann, T. (2018). Deep watershed detector for music object recognition. *arXiv preprint arXiv:1805.10548*.

Yesilkanat, A., Soullard, Y., Coüasnon, B., and Girard, N. (2023). Full-page music symbols recognition: state-of-the-art deep models comparison for handwritten and printed music scores.

Zhao, Z. et al. (2019). Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*.

Evaluating the Suitability of Long Document Embeddings for Classification Tasks: A Comparative Analysis

Bardia Rafieian[a] and Pere-Pau Vázquez[b]
ViRVIG Group Department of Computer Science, UPC-BarcelonaTECH, C/ Jordi Girona 1-3,
Ed Omega 137, 08034, Barcelona, Spain
{*bardia.rafieian, pere.pau.vazquez*}*@upc.edu*

Keywords: Long Document Classification, Document Embeddings, Doc2vec, Longformer, LLaMA-3, SciBERT, Deep Learning, Machine Learning, Natural Language Processing (NLP).

Abstract: Long documents pose a significant challenge for natural language processing (NLP), which requires high-quality embeddings. Despite the numerous approaches that encompass both deep learning and machine learning methodologies, tackling this task remains hard. In our study, we tackle the issue of long document classification by leveraging recent advancements in machine learning and deep learning. We conduct a comprehensive evaluation of several state-of-the-art models, including Doc2vec, Longformer, LLaMA-3, and SciBERT, focusing on their effectiveness in handling long to very long documents (in number of tokens). Furthermore, we trained a Doc2vec model using a massive dataset, achieving state-of-the-art quality, and surpassing other methods such as Longformer and SciBERT, which are very costly to train. Notably, while LLaMA-3 outperforms our model in certain aspects, Doc2vec remains highly competitive, particularly in speed, as it is the fastest among the evaluated methods. Through experimentation, we thoroughly evaluate the performance of our custom-trained Doc2vec model in classifying documents with an extensive number of tokens, demonstrating its efficacy, especially in handling very long documents. However, our analysis also uncovers inconsistencies in the performance of all models when faced with documents containing larger text volumes.

1 INTRODUCTION

Text embeddings are pivotal in natural language processing (NLP) tasks, such as text classification, where the quality of embeddings significantly affects performance. Traditional methods, such as Word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), have been foundational in generating embeddings at the token and sentence levels. Recent advancements include transformer-based models like BERT (Devlin et al., 2018) and ELMO (Peters et al., 2018), which have improved the quality of embeddings through contextualized representations.

Despite these advancements, handling very long documents presents substantial challenges. Models like BERT are limited by maximum sequence lengths, which restricts their ability to generate embeddings for extensive texts. While recent models such as Longformer (Beltagy et al., 2020) and large language models offer better performance, they come with high computational costs and resource demands (Samsi et al., 2023). These models are expensive to train and deploy, and there is a lack of standardized evaluations across various benchmarks (Tay et al., 2021).

On the other side, the scarcity of datasets with very long documents further complicates the issue, where existing labeled datasets consist only of short articles (up to 800 tokens per document), yet training a classifier for long texts requires a labeled dataset consisting of long documents. This gap in the literature highlights the need for more effective methods to handle lengthy texts while considering computational efficiency. This paper provides an evaluation of several state-of-the-art models, including Doc2vec, Longformer, LLaMA-3, and SciBERT, focusing on their effectiveness in handling long to very long document tokens. The study specifically aims to assess how well these models generate embeddings from documents that are exceptionally long in terms of token count. The key question is how agnostic these models are to document length, and how the quality of the generated embeddings influences their performance in downstream text classification tasks.

We also trained a Doc2vec model on a large

[a] https://orcid.org/0000-0003-4591-8934
[b] https://orcid.org/0000-0003-4638-4065

Rafieian, B. and Vázquez, P-P.
Evaluating the Suitability of Long Document Embeddings for Classification Tasks: A Comparative Analysis.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 320-327
ISBN: 978-989-758-716-0; ISSN: 2184-3228

dataset to evaluate its capability against BERT-based models and large language models (LLMs) in generating embeddings for very long documents. The goal was to assess how well the Doc2vec model performs compared to these advanced models in terms of both the quality of the embeddings produced and their effectiveness in a text classification task. This comparison provides insights into the relative strengths and limitations of traditional embedding methods like Doc2vec versus modern transformer-based approaches when handling lengthy documents. Given the scarcity of very long document datasets, our evaluation utilizes both public datasets and newly created datasets with over 4,000 tokens from arXiv and bioRxiv documents. The paper is structured as follows: Section 2 reviews related work. Section 3 details data preparation and preprocessing steps and the preparation of our pretrained Doc2vec model. In section 4 we study our experiments and finally, we conclude and discuss future directions.

2 RELATED WORKS

With the introduction of Word2Vec and GloVe, various methods have emerged to encode sentences, paragraphs, and longer texts into embeddings. Among these methods, Doc2vec (Le and Mikolov, 2014) stands out as a Paragraph Vector, an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of text. Empirical results have shown that Doc2vec outperforms bag-of-words models and other text representation techniques.

The advent of transformer models brought significant improvements in text encoders. BERT-based models, in particular, demonstrated substantial performance gains. The first application of BERT to document classification, as presented in ”DocBERT: BERT for Document Classification” (Adhikari et al., 2019), improved baseline results by fine-tuning BERT, achieving higher classification accuracy across various datasets. However, BERT-based models were limited by a fixed input sequence length of 512 tokens. To address this, models like SciBERT extended the number of tokens to 768 through fine-tuning. In SciBERT that follows the BERT architecture, which uses the Transformer model for encoding text, the process of generating embeddings can be described as follows:

The input is a tokenized text sequence:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]$$

The tokens are then converted to embeddings:

$$\mathbf{E} = [E(x_1), E(x_2), \ldots, E(x_n)]$$

Next, these embeddings pass through multiple transformer layers. Each transformer layer applies self-attention:

$$\mathbf{H}^{(l)} = \text{TransformerLayer}(\mathbf{H}^{(l-1)})$$

where $\mathbf{H}^{(0)} = \mathbf{E}$, and l is the layer number.

The final hidden states from the last transformer layer are used for downstream tasks:

$$\mathbf{H}^{(L)} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]$$

Despite these advancements, transformer-based models struggle with processing long sequences due to the computational complexity of their self-attention mechanism, which can lead to information loss in documents with more than 1,000 tokens. To overcome this limitation, the Longformer was introduced. It features an attention mechanism that scales linearly with sequence length, allowing it to handle documents with thousands of tokens. The Longformer achieves this by sparsifying the full self-attention matrix according to an ”attention pattern” that specifies which input locations attend to each other. This makes the model efficient for longer sequences. At the time of its introduction, the Longformer consistently outperformed RoBERTa on long document tasks, setting new state-of-the-art results on datasets like WikiHop and TriviaQA. Here is the process of generating embeddings using longformer:

The input is a tokenized sequence:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]$$

The attention mechanism is restricted to a fixed-size window:

$$\mathbf{A}_i = \text{Attention}\left(\mathbf{H}_i^{(l-1)}, \mathbf{H}_{i-w:i+w}^{(l-1)}\right)$$

where w is the window size. Global attention is applied to selected important tokens across the entire sequence. The embeddings are passed through multiple layers of this modified attention mechanism. The final hidden states are used for downstream tasks:

$$\mathbf{H}^{(L)} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]$$

More recently, significant efforts have been made to improve the performance of text encoders on long texts. Notable examples include the LLaMA-2 (Touvron and Lavril, 2023) and LLaMa-3 models. Although detailed technical information about these proprietary models is limited, they propose novel methods for generating embeddings from long texts, further advancing the field of NLP. Since we used LLaMA-3 and GEMMA-2B, we describe the process of generating embeddings as below:

LLaMA follows a standard transformer architecture with self-attention and feedforward networks.

The input is a tokenized text sequence:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]$$

Each transformer layer consists of multi-head self-attention and feedforward networks:

$$\mathbf{H}^{(l)} = \text{MultiHeadAttention}(\mathbf{H}^{(l-1)}) + \text{FFN}(\mathbf{H}^{(l-1)})$$

The final hidden states are used for downstream tasks:

$$\mathbf{H}^{(L)} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_n]$$

A recent study on transformer-based models (Fields et al., 2024) addresses key questions such as ”How Wide, How Large, How Long, How Accurate, How Expensive, and How Safe are they?” The study emphasizes the latest advancements in large language models (LLMs) by evaluating their accuracy across 358 datasets spanning 20 different applications. The findings challenge the assumption that LLMs are universally superior, revealing unexpected results related to accuracy, cost, and safety. LLMs now encompass both unimodal and multimodal tasks, where unimodal models use only textual information, and multimodal models incorporate text, video, signals, images, audio, and columnar data for classification. The paper highlights that while recent models like GPT-4 and Longformer can handle input text lengths of up to 8,192 tokens with high accuracy in classification tasks, the cost of training these LLMs, along with the associated economic and environmental concerns, has become a significant issue in recent years. Another notable study by (Wagh et al., 2021) examines the classification of long documents. The authors reaffirm that while BERT-based models can perform well across various datasets and are suitable for document classification tasks, they come with a high computational cost. They also point out that long document classification is a relatively simpler task, and even basic algorithms can achieve competitive performance compared to BERT-based approaches on most datasets.

3 METHODOLOGY

In our paper, we compare and discuss the capabilities of state-of-the-art models in generating high-quality embeddings for very long texts. Subsequently, we evaluate the generated embeddings using various methods, including Doc2vec, in the context of document classification tasks.

3.1 Datasets

Given the ongoing challenge of benchmarking very long texts due to the lack of agreement on datasets and baselines (Tay et al., 2021), we have prepared and introduced datasets with more than 1,000 tokens per text to evaluate embedding quality. table 1 shows the detailed information about each dataset.

Table 1: Dataset information including token size, sample size, and number of labels. Note*: 20 news and arxiv_100 information on section appendix 6.

Dataset	# Avg. Tokens	Size	Labels
Dataset#1	7630	554	11
Dataset#2	11305	1101	11
s2orc	3450	58905	4
20 news*	149	11297	20
arxiv_100*	121	100004	10

S2ORC. Semantic Scholar Open Research Corpus (Lo et al., 2020) is a comprehensive corpus designed for natural language processing and text mining on scientific papers. It includes over 136 million paper nodes, with more than 12.7 million full-text papers connected by approximately 467 million citation edges, derived from various sources and academic disciplines. The number of tokens in our selected dataset ranges from 1 to 287,400. We chose documents with at least 200 tokens in two classes of computer science and physics to ensure they are not smaller than the shortest document in our test set.

arxiv + Biorxiv. This dataset includes documents from the years 2022 and 2023 in both combination of arxiv and biorxiv, containing 550 and 1,000 documents respectively. Each document includes the full text of the papers, with an average of more than 7,000 tokens after preprocessing (tokenization, lemmatization, stop words removal and extra phrases removal). These datasets encompass multiple classes where for arxiv+biorxiv 2022 (labeled as **Dataset#1**) includes Evolutionary Biology, Paleontology, Mathematics, Computer Science, Zoology, Statistics, Pharmacology and Toxicology, Biochemistry, Economics, Physics and Electrical Engineering. On the other side, the arxiv+biorxiv 2023 (labeled as **Dataset#2**) dataset contain Biochemistry, Paleontology, Genomics, Quantitative Biology, Quantitative Finance, Statistics, Computer Science, Electrical Engineering and Systems Science, Mathematics, Physics and Zoology labels.

To prepare them, we first converted PDF documents to text format and then removed author names, images, tables, captions, references, acknowledgments, and formulas. Furthermore, we eliminated sentences with fewer than three tokens. All preprocessing steps, as well as subsequent operations, were executed using Python 3.

3.2 Embeddings

Doc2vec. Given Doc2vec's scalability with large datasets, we explored its functionalities by training it on extensive technical corpora. For training purposes, we focused on technical documents of S2ORC and collected 341,891 documents, totaling approximately 10GB, from fields including Engineering, Computer Science, Physics, and Math. It is important to note that we excluded the test set from the training set to train the Doc2vec model effectively. Finally, we generated the embeddings using Doc2vec.

SciBERT. (Beltagy et al., 2019), is a BERT-based model pre-trained on a large corpus of scientific text, which includes papers from the corpus of Semantic Scholar. The model aims to address the unique challenges posed by scientific text, such as specialized terminology and longer sentence structures. By leveraging this specialized pre-training, SciBERT achieves better performance on downstream scientific NLP tasks compared to the vanilla BERT model, particularly in domains like biomedical and computer science literature. In this model, full-text documents are encoded into chunks of 512 tokens. We generated text embeddings using the SciBERT model *SciBERT_scivocab_uncased* with a maximum sequence length of 512 tokens. The final layer's hidden states were used as embeddings, with mean pooling applied to obtain sentence-level embeddings. We utilized the Hugging Face ‘transformers‘ library (version 4.x.x) for model loading and inference.

Longformer. Introduced by (Beltagy et al., 2020), addresses the challenge of processing long documents by extending the input sequence token size up to 4096 tokens, significantly more than BERT's 512-token limit. Longformer employs a combination of local and global attention mechanisms that scale linearly with the sequence length, allowing it to handle much longer documents efficiently. This model is specifically designed to mitigate the computational inefficiencies of the quadratic complexity of the standard self-attention mechanism in BERT. In our experiments, we utilized the Longformer-large model *allenai/longformer_large_4096* to generate document embeddings. This model comprises 24 layers, each with a hidden size of 1024, and uses 16 attention heads. It is capable of processing sequences up to 4096 tokens in length, leveraging a sliding window attention mechanism with a window size of 512 tokens and supporting global attention for key tokens. Embeddings were generated by extracting the CLS token's output from the last hidden layer, optionally followed by mean pooling for a fixed-size representation.

LLaMA-3 and GEMMA-2B. Large Language Model for AI Assistance (Touvron and Lavril, 2023) represents a significant advancement in the realm of large-scale language models. Unlike earlier models like BERT or even Longformer, which are constrained by their maximum input sequence lengths, LLaMA-3 is designed to handle extremely large contexts, accommodating up to 16,000 tokens per sequence. This makes it particularly suitable for tasks involving extensive documents, such as entire books, comprehensive reports, and complex dialogues. Moreover, GEMMA-2B (Team, 2024) (Generative Embedding Model with Multi-headed Attention) distinguishes itself with a focus on generating high-quality embeddings for downstream NLP tasks. This model operates with a maximum input sequence length of 2048 tokens, striking a balance between the extensive context capabilities of models like LLaMA-3 and the more focused scope of traditional models. We generated text embeddings using the LLaMA 3 8B model provided by Ollama (Ollama, 2024). This model, which has 8.03 billion parameters, is optimized for instruction-following tasks and operates efficiently through quantization techniques, such as Q4_0. The embedding generation process utilizes the output from the model's last hidden layer, ensuring rich contextual representations of the input text. Ollama's quantization reduces the model's size to 5.5GB, allowing for effective deployment on local hardware while maintaining high-quality performance. Table 2 illustrates detailed information of each model.

Table 2: Characteristics of different models including vocabulary size, corpus size, maximum length, and embedding size.

Model	Vocab	Corpus	Max Len	Embedding
SciBERT	30K	1.14M	512	768
Doc2vec	33K	1.2M	10k	400
LLaMA-3	128K	15 Tn	8k	4096
GEMMA-2B	256K	6 Tn	8k	2048
Longformer	30K	33 Tn	4k	768

4 EXPERIMENTS AND RESULTS

In this section, we present the results of our experiments on several datasets using state-of-the-art models to generate high-quality embeddings for text classification. The models evaluated include Doc2vec, LLaMA-3, Longformer, SciBERT, and GEMMA-2B. We utilized both SVM and MLP classifiers to assess the performance of these embeddings. The evaluation metrics include accuracy, precision, recall, and F1 score. The reason we selected these classifiers, rather than model-based ones like LongformerClassifier, is to remain agnostic regarding classifier selection. This approach allows us to reuse the embeddings for other NLP tasks, providing greater flexibility and utility. Below we give more information on each:

SVM. We utilized a Support Vector Machine (SVM) classifier with a linear kernel to perform the classification tasks. The model was configured with a regularization parameter, C, set to 1.0 to balance the trade-off between minimizing training error and achieving low testing error. The SVM classifier was trained on the given feature set and corresponding labels, facilitating effective class separation within the feature space. **MLP.** We utilized the MLP Classifier from scikit-learn to build a neural network classifier for our dataset. The model features two hidden layers with 100 and 50 neurons, respectively, and was trained for a maximum of 60 iterations. We set the random seed to 42 for reproducibility.

4.1 Results

We observed Doc2vec consistently demonstrated robust performance on the Dataset#2 dataset, achieving an MLP accuracy of 0.67, and an F1 score of 0.65. Longformer also delivered competitive results, with SVM accuracy of 0.64, and an F1 score of 0.65. In contrast, SciBERT and LLaMA-3 showed slightly lower performance, with SVM accuracies of 0.61 and 0.56, and MLP accuracies of 0.64 and 0.60. The GEMMA-2B model, however, had the least favorable outcomes. We can express the lower results of GEMMA-2B model comparing with LLaMA-3 due to its lower embedding dimension and model parameters size. We were surprised by the strong performance of TF-IDF embeddings, which outperformed all other models, likely due to its effectiveness in handling massive documents. On Dataset#1 Doc2vec emerged as the top performer, achieving an SVM accuracy of 0.7590, an MLP accuracy of 0.71, and an F1 score of 0.78. SciBERT followed closely, with an SVM accuracy of 0.72, an MLP accuracy of 0.71, and an F1 score of 0.72. Longformer, however, showed a decline in performance, reflected by an SVM accuracy of 0.5500, an MLP accuracy of 0.5833, and an F1 score of 0.5550. LLaMA-3 provided moderate results with an SVM accuracy of 0.4940, an MLP accuracy of 0.3976, and an F1 score of 0.7804. Meanwhile, GEMMA-2B continued to struggle, recording the lowest performance metrics with an SVM accuracy of 0.4700, an MLP accuracy of 0.4600, and an F1 score of 0.4500.

Finaly LLaMA-3 demonstrated superior performance on the S2ORC dataset with two classes, achieving nearly perfect scores with an SVM accuracy, MLP accuracy, and F1 score all at 0.99. Doc2vec also showed strong results, with an SVM accuracy of 0.97, an MLP accuracy of 0.99, and an F1 score of 0.9776. Both Longformer and SciBERT maintained high levels of accuracy, with SVM scores of 0.96 and 0.97, and MLP accuracies of 0.99 and 0.97, respectively, complemented by high F1 scores. These findings highlight the remarkable efficiency of LLaMA-3 and Doc2vec in managing large-scale scientific documents.

The results in table 3 indicates that LLaMA-3 consistently outperforms other models across various datasets, particularly on the 20 news (6) and S2ORC datasets, demonstrating its robustness and effectiveness in handling long and shorter documents by generating high-quality embeddings. Doc2vec also shows competitive performance, especially on the S2ORC dataset. Longformer and SciBERT exhibit moderate performance, with SciBERT performing better on the arxiv_100 dataset (APPENDIX). GEMMA-2B, while a powerful model for embedding generation, did not perform as well in this classification task, suggesting that its embeddings might need further fine-tuning for specific tasks or datasets. To further analyze the effectiveness of the embeddings generated by the different models, we projected the high-dimensional embeddings into a 2D space using the PACMAP dimensionality reduction technique (Wang et al., 2021). This visualization allows for a deep understanding of how well the models differentiate between classes in various datasets (see Appendix 6).

4.2 Training and Inference Time

Transformer-based models, such as SciBERT, LLMs, and Longformer, possess a complex architecture involving multi-head self-attention mechanisms and multiple layers, which enable them to capture entangled dependencies and contextual information. These models typically require massive datasets for pre-training where depending on the model size and hardware, the training time can range from several days to months, although fine-tuning usually takes a few hours to a few days on powerful GPUs (Devlin et al., 2018). In contrast, simpler models like Word2Vec and Doc2vec use much less complex architectures. Word2Vec, for example, leverages shallow neural networks with a single hidden layer, while Doc2vec extends Word2Vec by considering document context but remains relatively straightforward. These models also utilize large datasets but not to the extent required for transformer models, typically training on corpora

Table 3: Evaluation metrics Macro average of (Precision, Recall, F1 Score) and SVM Classification/MLP classification accuracy for different models across bioarxiv 2022, bioarxiv 2023, and s2orc datasets.

Dataset	Model	Macro avg. P	Macro avg. R	Macro avg. F1	SVM acc	MLP acc
Dataset#2	Doc2vec	0.6702	**0.6545**	**0.6593**	**0.6545**	**0.6780**
	GEMMA-2B	0.5100	0.5100	0.5000	0.5000	0.5000
	LLaMA-3	0.5964	0.5697	0.5744	0.5697	0.6000
	Longformer	**0.6919**	0.6424	0.6519	0.6424	0.6420
	SciBERT	0.6295	0.6182	0.6213	0.6180	0.6400
	TF-IDF	**0.8900**	**0.8800**	**0.8800**	**0.8800**	**0.8900**
Dataset#1	Doc2vec	**0.8181**	**0.7711**	**0.7825**	**0.7590**	0.7100
	GEMMA-2B	0.4800	0.4700	0.4500	0.4700	0.4600
	LLaMA-3	0.7819	0.7819	0.7804	0.4940	0.3976
	Longformer	0.6789	0.5500	0.5550	0.5500	0.5833
	SciBERT	0.7597	0.7229	0.7279	0.7229	**0.7100**
	TF-IDF	0.7400	0.7200	0.7200	**0.7600**	**0.7800**
s2orc	Doc2vec	0.9775	0.9777	0.9776	0.9778	**0.9998**
	LLaMA-3	**0.9976**	**0.9976**	**0.9976**	**0.9976**	0.9976
	Longformer	0.9674	0.9677	0.9675	0.9678	0.9993
	SciBERT	0.9749	0.9749	0.9749	0.9749	0.9797
	TF-IDF	0.9700	0.9700	0.9700	0.9800	0.9800

containing millions to billions of words. Training these models is much faster, with Doc2vec being trainable on a large corpus in a matter of hours using a few CPUs or a single GPU, still considerably quicker than transformer models. Figures 1a and 1b show the comparison of fine-tuning—training/ time and memory/time of full self-attention and different implementations of Longformer's methods vs Doc2vec.

As shown in 2, Doc2vec can perform competitively while offering significant advantages in terms of inference time, resource requirements, and energy consumption. Specifically, Doc2vec demonstrates much faster average embedding inference times per second on a CPU, needing significantly less computational resources and consuming less energy compared to other models.

5 LIMITATIONS

One of the significant challenges encountered in this study was finding datasets with tokens exceeding 1,000 to effectively compare the models' ability to extract embeddings from very long texts. Such datasets are crucial for evaluating model performance on extended sequences.

Additionally, inferring heavy models like LLaMA-3 and GEMMA-2B required substantial time, effort, and computational resources. These models have considerable demands, and their inference process was constrained by the limitations of available libraries and computing environments.

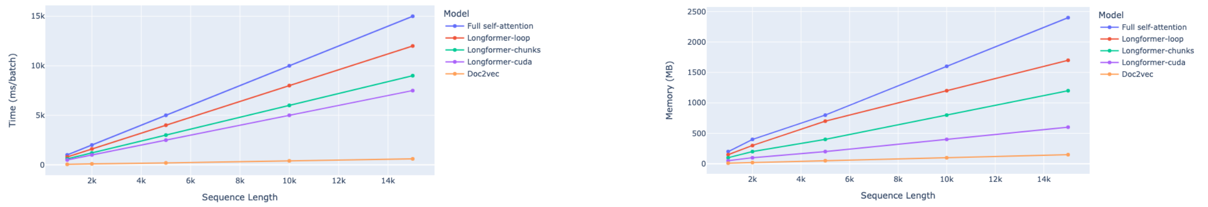
6 CONCLUSIONS

In this study, we have evaluated the performance of various state-of-the-art models, including Doc2vec, SciBERT, Longformer, LLaMA-3, and GEMMA-2B, on the task of generating high-quality embeddings for text classification. Our experiments spanned multiple datasets such as 20 news, arxiv_100, Dataset#1, Dataset#2, and S2ORC, providing a comprehensive analysis of each model's strengths and limitations.

The results indicate that LLaMA-3 consistently outperforms other models across different datasets, particularly excelling in the 20 news and S2ORC datasets with superior accuracy and F1 scores. SciBERT also demonstrated robust performance, especially with the arxiv_100 dataset. Notably, Doc2vec, while slightly behind in absolute performance metrics, offers competitive results with significantly better computational efficiency, making it an excellent choice for applications requiring faster inference times and lower resource consumption. This balance between performance and efficiency is critical for practical deployment in real-world scenarios.

Additionally, our study highlighted the challenges associated with handling very long documents, where models like Longformer and LLaMA-3, designed for extended context processing, showed significant advantages. However, GEMMA-2B, despite its powerful embedding capabilities, requires further fine-tuning.

In future, we aim to investigate the quality of embeddings in additional NLP tasks, such as question



(a) Fine-tuning time of self-attention Longformers and Doc2vec (Beltagy et al., 2020).

(b) Memory usage of self-attention Longformers and Doc2vec (Beltagy et al., 2020).

Figure 1: Performance comparison of Longformers and Doc2vec models.

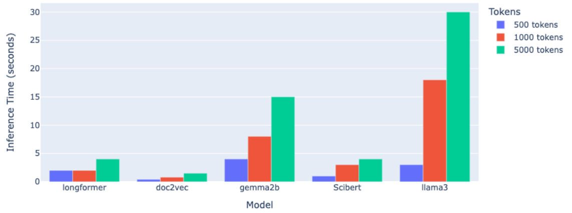


Figure 2: Inference time of different models for embedding extraction.

answering and summarization on very long texts. We will also review the tuned combinations of embeddings for specific tasks and domains.

ACKNOWLEDGEMENTS

This project has been supported by PID2021-122136OB-C21 from the Ministerio de Ciencia e Innovación, by 839 FEDER (EU) funds.

REFERENCES

Adhikari, A., Ram, A., Tang, R., and Lin, J. (2019). Docbert: BERT for document classification. *CoRR*, abs/1904.08398.

Beltagy, I., Lo, K., and Cohan, A. (2019). Scibert: A pretrained language model for scientific text.

Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Fields, J., Chovanec, K., and Madiraju, P. (2024). A survey of text classification with transformers: How wide? how large? how long? how accurate? how expensive? how safe? *IEEE Access*, 12:6518–6531.

Lang, K. (1995). Newsweeder: learning to filter netnews. In *Proceedings of the Twelfth International Conference on International Conference on Machine Learning*, ICML'95, page 331–339, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents. *CoRR*, abs/1405.4053.

Lo, K., Wang, L. L., Neumann, M., Kinney, R., and Weld, D. (2020). S2ORC: The semantic scholar open research corpus. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Ollama (2024). Ollama: Ai models locally. Accessed: July 26, 2024.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global vectors for word representation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *CoRR*, abs/1802.05365.

Samsi, S., Zhao, D., McDonald, J., Li, B., Michaleas, A., Jones, M., Bergeron, W., Kepner, J., Tiwari, D., and Gadepally, V. (2023). From Words to Watts: Benchmarking the Energy Costs of Large Language Model Inference. *arXiv e-prints*, page arXiv:2310.03003.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., and Metzler, D. (2021). Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*.

Team, G. (2024). Gemma: Open models based on gemini research and technology.

Touvron, H. and Lavril, T. (2023). Llama: Open and efficient foundation language models.

Wagh, V., Khandve, S. I., Joshi, I., Wani, A., Kale, G., and Joshi, R. (2021). Comparative study of long document classification. *CoRR*, abs/2111.00702.

Wang, Y., Huang, H., Rudin, C., and Shaposhnik, Y. (2021). Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22(201):1–73.

APPENDIX

Datasets 20 News: (Lang, 1995) is widely used for text classification and natural language processing (NLP) tasks. It contains approximately 20,000 newsgroup documents, divided into 20 different newsgroups.

arxiv_100: dataset comprises 100,000 arXiv paper abstracts and averages 121 tokens per document, covering subjects such as Electrical Engineering and Systems Science, Statistics, Computer Science, Physics, Quantum Physics, Mathematics, High Energy Physics - Theory, High Energy Physics, Condensed Matter Physics, and Astrophysics.

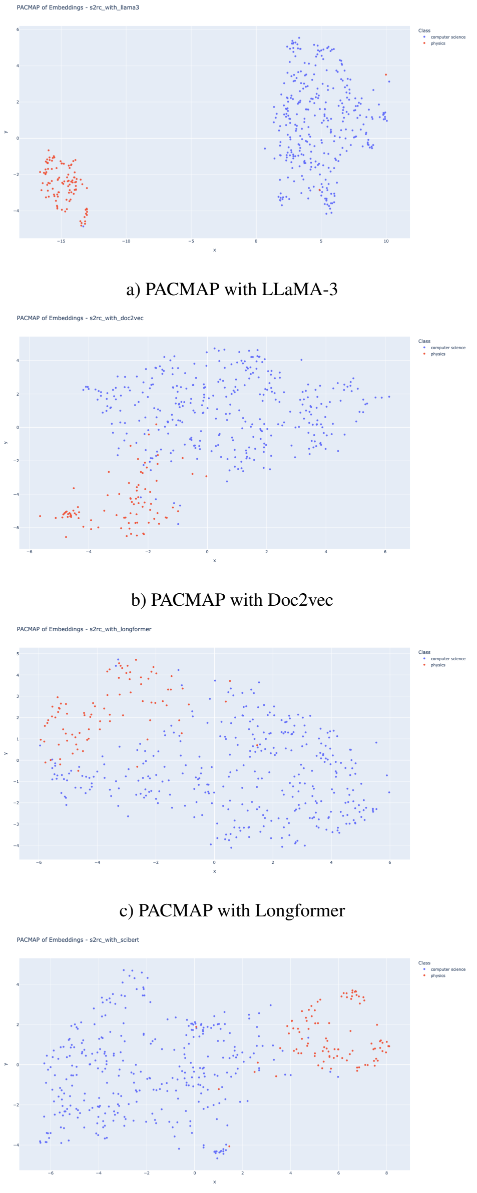
Results: On 20 news dataset, LLaMA-3 significantly outperformed other models, achieving an SVM accuracy of 0.97 and an F1 score of 0.97. Doc2vec showed decent performance with an SVM accuracy of 0.75, while its F1 score was 0.67. Longformer and SciBERT demonstrated moderate results, with SVM accuracies of 0.75 and 0.66, and MLP accuracies of 0.65 and 0.65, respectively. LLaMA-3's results reflect its superior ability to handle the complexity of the newsgroup data.

On arxiv_100, SciBERT led with an SVM accuracy of 0.81, both with an F1 score of 0.81. Doc2vec followed closely, with SVM and MLP accuracies of 0.81 and 0.76. LLaMA-3 also performed well, showing an SVM accuracy of 0.78, and an F1 score of 0.78. Longformer lagged behind with an SVM accuracy of 0.72 and an MLP accuracy of 0.74, with an F1 score of 0.72. These results underscore SciBERT's effectiveness in handling scientific abstracts and technical documents. Table 4 summarizes the classification and F-score results on these datasets.

Dimensionality Reduction and Embedding Analysis: We applied the PACMAP dimensionality reduction method (Wang et al., 2021) to embeddings extracted from various models on the S2ORC test set. As illustrated in Figure 3, LLaMA-3 effectively separated the embeddings in the 2D space, demonstrating distinct class separation. While Doc2Vec and SciBERT also achieved some degree of separation between classes, the resulting data points remained in close proximity within the 2D space. Finally, Longformer, despite distinguishing the classes, performed the weakest separation performance among the others.

Table 4: Evaluation metrics (Precision, Recall, F1 Score) and SVM/MLP classification results for different models across arxiv_100 and 20 news datasets.

Data	Model	P	R	F1	SVM	MLP
20n	Doc2vec	0.6700	0.6665	0.6665	0.747	0.694
	LLaMA-3	0.9775	0.9741	0.9749	0.974	0.971
	Longf	0.6481	0.6324	0.6303	0.746	0.646
	SciBERT	0.6628	0.6581	0.6589	0.658	0.653
arxiv_100	Doc2vec	0.8009	0.8007	0.8007	0.805	0.756
	LLaMA-3	0.7819	0.7819	0.7804	0.781	0.780
	Longf	0.7183	0.7173	0.7171	0.716	0.739
	SciBERT	0.8094	0.8093	0.8093	0.809	0.785



a) PACMAP with LLaMA-3

b) PACMAP with Doc2vec

c) PACMAP with Longformer

d) PACMAP with SciBERT

Figure 3: 2D embedding visualization on S2ORC dataset(tests) , extracted from a)LLaMA-3, b)Longformer, c)SciBERT and d)Doc2vec, results showing a great performance of LLaMA-3 on class separations.

Prompt Distillation for Emotion Analysis

Andrew L. Mackey, Susan Gauch and Israel Cuevas
Deparment of Electrical Engineering and Computer Science, University of Arkansas , Fayetteville, Arkansas, U.S.A.
{*almackey, sgauch, ibcuevas*}*@uark.edu*

Keywords: Emotion Analysis, Natural Language Processing.

Abstract: Emotion Analysis (EA) is a field of study closely aligned with sentiment analysis whereby a discrete set of emotions are extracted from a given document. Existing methods of EA have traditionally explored both lexicon and machine learning techniques for this task. Recent advancements in large language models have achieved success in a wide range of tasks, including language, images, speech, and videos. In this work, we construct a model that applies knowledge distillation techniques to extract information from a large language model which instructs a lightweight student model to improve its performance with the EA task. Specifically, the teacher model, which is much larger in terms of parameters and training inputs, performs an analysis of the document and shares this information with the student model to predict the target emotions for a given document. Experimental results demonstrate the efficacy of our proposed prompt-based knowledge distillation approach for EA.

1 INTRODUCTION

Sentiment analysis (SA) is a prominent subfield of natural language processing (NLP) with the goal of analyzing text documents from which the document's polarity is obtained. Emotion analysis (EA) establishes additional granularity for classes beyond polarity from SA by focusing on the alignment of language with various emotional categories. For example, the Paul Ekman model for emotions defines six primary emotion categories: anger, disgust, fear, joy, sadness, and surprise (Ekman and Friesen, 1971). Another approach to illustrate the various emotional dimensions was proposed as the Robert Plutchik model with eight primary bipolar emotions: anger versus fear, joy versus sadness, anticipation versus surprise, and trust versus disgust (Plutchik, 1982). Additional models have been proposed that projects emotions into a dimensional space, such as for valence, arousal, and dominance (Russell and Mehrabian, 1977).

Various techniques have been proposed for the task of emotion analysis. The first major area of emotion analysis involves lexicon-based techniques where the techniques are focused on aligning the emotional categories of language with the specific words that were used (Baccianella et al., 2010) (Staiano and Guerini, 2014). The next major area of emotion analysis includes various machine learning techniques that discover latent patterns or representations for the detection of different emotional categories (Agrawal and An, 2012) (Calefato et al., 2018) (Hasan et al., 2019). Some researchers have investigated emotion representations that seek to achieve emotion representations that transcend multiple lexicons and datasets (Buechel et al., 2020). Some work in emotion classification has concentrated on aligning transformer-based architectures with emotional categories through deep contextual representations. Pretrained language models (PLM) have demonstrated various successes in outperforming many state-of-the-art techniques in the field. As the parameters and training data continued to scale for PLMs, large language models (LLM) emerged and demonstrated capabilities not seen in prior work, such as prompt-based learning and reasoning.

In this paper, we introduce a prompt-based knowledge distillation model for emotion analysis where the prompt serves as source of knowledge through which we distill that information for a student model under the supervision of a much larger teacher model. The first phase of our model involves a prompt-based teacher model followed by a knowledge distillation student training model. The teacher model uses prompt-based techniques to extract information from the LLM. The student model uses a transformer-based PLM where probabilities from both teacher and student models are aligned so that the student model is capable of generating similar probability distributions as the teacher model.

Mackey, A., Gauch, S. and Cuevas, I.
Prompt Distillation for Emotion Analysis.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 328-334
ISBN: 978-989-758-716-0; ISSN: 2184-3228

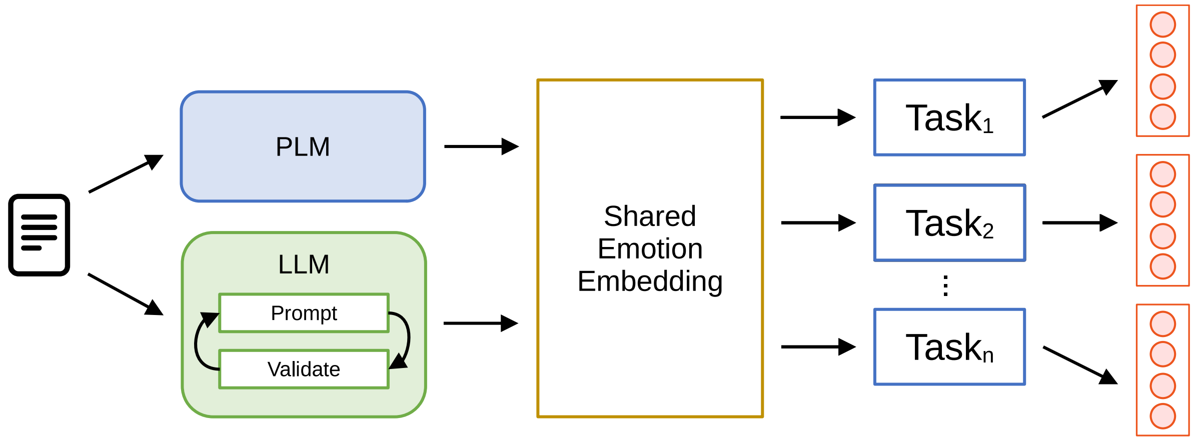


Figure 1: Overview architecture of the model. We combine a pre-trained language model with a large language model to extract the emotion embeddings cross-corpus to perform a classification of the emotions. For the final prediction y, we localize the classification head to a set of possible classes for the respective datatset.

2 RELATED WORK

Recent work in the research community has focused on tasks involving emotion analysis has concentrated primarily on PLMs for learning contextual representations using neural networks (Demszky et al., 2020) (Turcan et al., 2021) (Alhuzali and Ananiadou, 2021) (Wullach et al., 2021) (Mackey et al., 2021) (Toraman et al., 2022) (Rahman et al., 2024). PLMs undergo various training methods which enables them to learn latent contextual representations of text. These models are generally fine-tuned in order to adapt to task-specific objectives, such as emotion classification. Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based architecture that bidirectionally encoded embeddings to learn contextual information in textual data where the model was pre-trained simultaneously on the tasks of masked language modeling (MLM) and next sentence prediction (NSP) (Devlin et al., 2019). XL-Net improves upon BERT by introducing permutation language modeling where tokens are predicted in a random order (Yang et al., 2019). RoBERTa improved upon BERT by modifying the training approach where the NSP task was removed and dynamic masking was introduced, and increasing the amount of training data that was used (Liu et al., 2019).

Other work with LMs has resulted in different techniques for training methodologies. Knowledge distillation techniques, where a teacher model transfers knowledge from a complex model to a much simpler model, how shown promising results across different studies (Hinton et al., 2015) (Lukasik et al., 2022). Brown et al. define various levels of data used for in-context learning, such as fine-tuning (updating weights of a pretrained model), few-shot (models are provided a few demonstrations of a task with no additional weight updates to the model), one-shot (only one demonstration is permitted), and zero-shot (no demonstrations are permitted) (Brown et al., 2020). Brown et al. also demonstrate that as LMs increase in scale, their task-agnostic few-shot performance also increases (Brown et al., 2020). In addition, Halder et al. acknowledged that tranformer-based LMs fine-tuned to task-specific objectives curtail their ability to perform well in zero-shot, one-shot, or few-shot scenarios (Halder et al., 2020).

Work involving large language models continues to demonstrate their task-agnostic capabilities. One study demonstrated a technique of applying a series of reasoning steps named *chain of thought* where an LLM utilized chain-of-thought prompting that demonstrated reasoning abilities provided the LLM is adequately large (Wei et al., 2022). Adversarial distillation frameworks have also been proposed in research literature for improved knowledge distillation and transfer learning (Jiang et al., 2023).

3 PROBLEM DEFINITION

Let $\mathcal{D}$ represent a dataset comprised of N documents, where each document in $\mathcal{D}$ consists of textual information and emotion labels. We observe the following for each $\mathcal{D}$: **(1)** the set of text documents in dataset $\mathcal{D}$ is represented as $\mathcal{X}_D$ such that $|\mathcal{X}_D| = N$; **(2)** the set of possible target labels for dataset $\mathcal{D}$ is represented as $\mathcal{Y}_D$ where $|\mathcal{Y}_D| = C$ different emotions; and **(3)** $\mathcal{D}$ is represented as the following set in the single-label setting:

$$\mathcal{D} = \{(x, y) \mid x \in \mathcal{X}_D \text{ and } y \in \mathcal{Y}_D\} \tag{1}$$

and the following serves as the representation for a multi-label setting:

$$\mathcal{D} = \{(x, y) \mid x \in \mathcal{X}_D \text{ and } y \in \mathcal{P}(\mathcal{Y}_D)\} \tag{2}$$

Let $\mathbf{D}$ represent the input text corpora where $\mathbf{D} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_n\}$. The task presented in this work is to train and align a model to recognize the latent emotion representations in a cross-corpus setting using $\mathbf{D}$ for the purpose of single-class and multi-class emotion classification of an emotion label (or set) y from a given input document x:

$$\hat{\mathbf{y}} = \arg\max_{c} \left[\Pr(y = c \mid x; \Theta) \right] \tag{3}$$

4 PROPOSED APPROACH

We present our proposed solution in this section for the single-class and multi-class cross-corpora emotion classification task. In Figure 2, we provide an overview of our framework for learning the latent emotion distribution of text documents. There are three major components to our approach: **(1)** a prompt-based knowledge distillation paradigm for extracting information from an LLM to facilitate the alignment of a task-specific model; **(2)** a task-specific, emotion classification model that leverages a pre-trained, transformer-based language model, which is fine-tuned for the emotion classification task; and **(3)** a cross-corpora framework for learning latent emotion representations.

4.1 Prompt-Based Methodology

For a given dataset $\mathcal{D} = (\mathcal{X}_D, \mathcal{Y}_D)$, each input and target is represented as $(\mathbf{x}^{\text{doc}}, y^{\text{emo}})$ such that $(\mathbf{x}^{\text{doc}}, y^{\text{emo}}) \in \mathcal{D}$. The target y^{emo} of the model is the emotion class for each document where $y^{\text{emo}} \in \mathcal{Y}_D$ (i.e. anger, grief, disgust, etc.) in the respective dataset $\mathcal{D}$. To facilitate knowledge distillation from an LLM, we define $(\mathbf{x}^{\text{prompt}}, \mathbf{y}^{\text{llm}})$ to represent the prompt-based input and label generated from an LLM for each $(\mathbf{x}^{\text{doc}}, y^{\text{emo}}) \in \mathcal{D}$.

Prompt Template. The following template is used to the generate each $\mathbf{x}^{\text{prompt}}$:

You will be given a human written sentence. Classify the sentence into one of the following categories: $\langle \mathbf{y}_0^{emo}, \mathbf{y}_1^{emo}, ... \rangle$. Return the following format only for each category as a probability distribution (the sum should be 1): $\langle \mathbf{y}_i^{emo}$, probability$\rangle$.

The following is the document: $\mathbf{x}_i$.

The target $\mathbf{y}^{\text{llm}}$ represents the emotion distribution produced by the LLM for the given input prompt $\mathbf{x}^{\text{prompt}}$, which is modeled as follows:

$$\hat{\mathbf{y}}_i^{\text{llm}} = \Pr(y^{\text{emo}} \mid \mathbf{x}_i^{\text{prompt}}) \tag{4}$$

$$= \text{LLM}(\mathbf{x}^{\text{prompt}}) \tag{5}$$

Hallucinations are a known problem in research literature where an LLM produces a response that is either factually incorrect or unaligned with the input prompt it was provided (Farquhar et al., 2024). To address the problem of hallucinations, we conduct a validation step for $\hat{\mathbf{y}}^{\text{llm}}$ to ensure the format of the output is aligned with the targets in the training data. Documents failing the validation step will undergo a fixed interval of reprompting where the input and interactions are returned to the LLM for further processing in the form:

$$\hat{\mathbf{y}}^{\text{llm}'} = \text{LLM}(\ \langle \mathbf{x}^{\text{prompt}'}, \langle \mathbf{x}^{\text{prompt}}, \mathbf{y}^{\text{llm}} \rangle \rangle\ ) \tag{6}$$

4.2 Emotion Classification Model

The task-specific emotion classification model begins by employing the use of a transformer-based language model to provide contextual representations $\mathbf{h}^{\text{emo}} = \langle \mathbf{h}_1^{\text{emo}}, \mathbf{h}_2^{\text{emo}}, ..., \mathbf{h}_k^{\text{emo}} \rangle$ for input tokens $\mathbf{x}^{\text{doc}}$ where k represents the number of time steps. The transformer-based encoder LM is parameterized with ϕ for all datasets $\mathcal{D} \in \mathbf{D}$ to generate the contextualized word representations $\mathbf{h}_i^{\text{emo}}$ for each time step i:

$$\mathbf{h}_i^{\text{emo}} = \text{LM}_\phi(\mathbf{x}_i^{\text{doc}}) \tag{7}$$

The last layer of $\mathbf{h}_i^{\text{emo}}$ is used to compute the distribution for the emotion classes, where it is parameterized by ϕ_d for each $\mathcal{D}_d \in \mathbf{D}$ to obtain the target prediction distribution $\hat{\mathbf{y}}_i^{\text{emo}}$ and the softmax layer is applied to normalize the logits:

$$\hat{\mathbf{y}}_i^{\text{emo}} = \Pr(y_i^{\text{emo}} \mid \mathbf{h}_i^{\text{emo}}) \tag{8}$$

$$= \text{Softmax}(\mathbf{W}_{\phi_d} \mathbf{h}_i^{\text{emo}} + b_{\phi_d}) \tag{9}$$

The model shares a common set of parameters ϕ between all members of $\mathbf{D}$ to facilitate latent emotion representation learning in a cross-domain environment, while the task-specific classification head maintains a specific set of a parameters ϕ_d.

4.3 Knowledge Distillation

The goal of a prompt-based teacher model is to extract knowledge from an LLM and transfer it to the task-specific student model, which is responsible for fine-grained emotion classification. The prompt-based model instructs the emotion classification model to enable the smaller model to generalize in a manner that resembles the teacher model. The student model minimizes a loss function which focuses on both correctly predicting the target label y^{emo} while simultaneously aligning the model with the teacher model's responses $\mathbf{y}^{\text{llm}}$.

The model utilizes a cross-entropy loss function for the single-class emotion classification task

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N}\sum_{i}^{C} y_i^{\text{emo}} \log(\hat{y}_i^{\text{emo}}) \tag{10}$$

and a binary cross-entropy loss function for multi-class emotion classification

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i}^{C}\left[y_i^{\text{emo}}\log(\hat{y}_i^{\text{emo}}) + (1-y_i^{\text{emo}})\log(1-\hat{y}_i^{\text{emo}})\right] \tag{11}$$

for when there exists multiple emotion labels for a given document.

We use τ to represent the temperature rate hyperparameter to produce a softer probability distribution over all possible classes for class imbalances through knowledge distillation techniques. For these models, the losses from the emotion detection task and the prompt-based alignment model are summed together after each batch by using the adjustable hyperparameter λ, which balances the terms below:

$$\mathcal{L}_{\phi} = \lambda\mathcal{L}_{emo} + (1-\lambda)\tau^2\mathcal{L}_{llm} \tag{12}$$

5 EXPERIMENTS

In this section, we provide an empirical analysis of our proposed model and investigate the following research questions:

- **RQ1:** What is the effectiveness of the proposed model for the emotion classification task in terms of model performance metrics?
- **RQ2:** Does the choice of LM contribute to the performance of the proposed model?
- **RQ3:** How does the knowledge distillation from an LLM to the proposed model contribute to the overall performance?

5.1 Data

Our experiments are conducted on two benchmark datasets: WASSA-21 dataset and Real World Worry dataset (Buechel et al., 2018) (Kleinberg et al., 2020). The WASSA-21 dataset was provided in the 11th Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media Analysis (WASSA) Shared Task: Empathy Detection and Emotion Classification (Tafreshi et al., 2021). The dataset consists of $n = 1860$ reactions to news stories indicating that there is harm to a person, group, or other. The labels for each record are mapped to seven emotion categories, which include a *neutral* category and Ekman's basic emotion categories: *anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise*. This label represents the dominant emotion for the text.

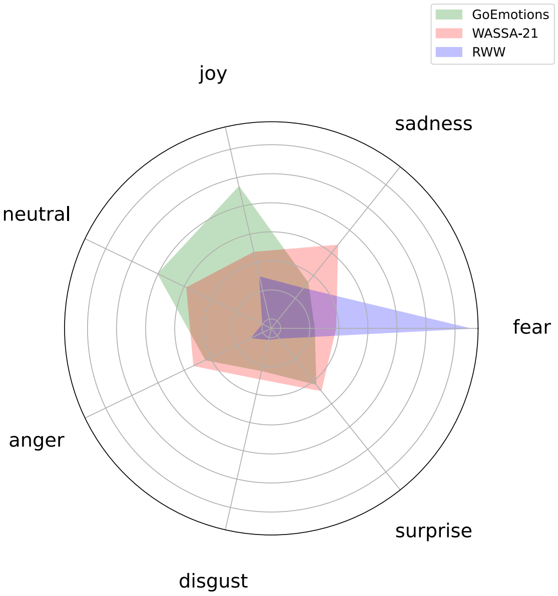


Figure 2: Distribution of the emotion labels by dataset. The RWW dataset emphasized *fear* and *sadness* labels. The GoEmotions dataset had a stronger presence of documents labeled as *neutral* and *joy*. The WASSA dataset contained more labels with the *sadness* and *surprise* labels in comparison to other datasets.

Table 1: GoEmotions emotion mapping to Ekman emotions.

Emotion	Association
anger	anger, annoyance, disapproval
disgust	disgust
fear	fear, nervousness
joy	joy, amusement, approval, excitement, gratitude, love, optimism, relief, pride, admiration, desire, caring
sadness	sadness, disappointment, embarrassment, grief, remorse
surprise	surprise, realization, confusion, curiosity

The second dataset used in our experiments is the COVID-19 Real World Worry dataset (Kleinberg

Table 2: COVID-19 emotion mapping to Ekman emotions.

Emotion	Association
anger	anger
disgust	disgust
fear	fear, anxiety
joy	happiness, relaxation
sadness	sadness
surprise	desire

et al., 2020). The dataset contains $n = 2491$ records that were extracted by surveying participants and ask them to express their emotional feelings towards the COVID-19 pandemic. Participants were asked to construct two different forms of text. The first document they were asked to author included instructions to express their feelings towards the then current COVID-19 situation with a minimum of 500 characters. The second document expressed them to convey the same feelings in the form of a social media post that had a maximum of 240 characters. Participants were asked to rate their emotions toward the situation and select one of the following emotions that best represented their feelings: anger, anxiety, desire, disgust, fear, happiness, relaxation, and sadness. We used the emotion definitions from (Demszky et al., 2020) as indicated in Table 1 to map perform the emotion mappings as indicated in Table 2.

5.2 Baseline Experiments

To evaluate the efficacy of our proposed prompt-based knowledge distillation model, we use PLMs as the baseline for our experiments. We benchmark our model using the BERT, RoBERTa, and XLNet PLMs where the input will only be the document and target emotion(s). We evaluate the model performance of each dataset and report the mean precision, recall, and F_1-scores after 3 runs using macro averaging.

5.3 Experimental Settings

Our model was constructed using the PyTorch framework along with the HuggingFace `transformers` library for the pretrained language model implementations.[1] We followed similar experimental settings as provided in (Demszky et al., 2020). Our model uses the AdamW optimizer (Loshchilov and Hutter, 2017) while setting the learning rate to $5e^{-5}$, batch size to 16, and maximum sequence length of 512. Since previous research literature demonstrated overfitting beyond four epochs, we limited our the number of epochs during the fine-tuning step to four (Demszky et al., 2020). For the large language model, we utilized the `GPT-4o` model provided through the API.

5.4 Experimental Results

Table 3 reflects the results from the experiments conducted in this paper. Each experiment was executed independently of other datasets. The best results are indicated in bold. As reflected in the results, our method is able to demonstrate increased performance above the baseline methods for the WASSA-21 and RWW datasets. This demonstrates that the PLM acquires additional knowledge through transfer learning and knowledge distillation through this technique that it did not acquire through the data alone. Furthermore, we also discover that the RoBERTa PLM is able to achieve superior performance over the other PLMs evaluated in the tests we conducted. Despite the extreme differences in the distribution of the labels between the datasets as evidenced in Figure 2, we observe that the proposed technique is able to work given the task-agnostic knowledge provided from the teacher model. When RoBERTa was used as the underlying PLM for our technique, we were able to achieve a gain of $\Delta = +2.18$ increase in performance for the F_1 score for the WASSA-21 dataset and $\Delta = +1.86$ for the RWW dataset.

It should also be noted that the largest gain in performance was achieved through the prompt-based knowledge distillation approach with the BERT PLM in the RWW dataset. We observe an increase of $\Delta = +2.37$ in the F_1 score under these settings.

6 CONCLUSIONS

Throughout our work in this paper, we investigated the task of emotion analysis under a prompt-based knowledge distillation setting where we trained a student model by aligning it with a teacher model which provides instruction on how to generate similar probability distributions in a task-specific objective. Future directions for this work can involve exploring other techniques, such as chain-of-thought or other reasoning approaches, or augmented LLM approaches to improve the teacher model through prompting strategies. The proposed methodology can be extended to consider additional modalities of data.

REFERENCES

Agrawal, A. and An, A. (2012). Unsupervised emotion detection from text using semantic and syntactic rela-

[1] https://huggingface.co/docs/transformers/en/index

Table 3: Comparison of baselines with experimental settings. Our proposed prompt-based knowledge distillation models outperform the baseline models.

Type	**Model**	**WASSA-21**			**RWW**		
		Precision	Recall	F1	Precision	Recall	F1
Baseline	BERT	68.52	68.67	67.70	18.81	19.33	18.80
	RoBERTa	72.39	73.84	71.74	23.20	21.97	20.61
	XLNet	60.74	63.18	60.92	20.40	20.26	18.52
Experiment	BERT+PKD	69.02	71.15	68.58	23.52	23.32	21.17
	RoBERTa+PKD	**73.85**	**75.16**	**73.92**	**24.63**	**22.69**	**22.47**
	XLNet+PKD	62.55	64.29	61.75	22.52	21.41	19.28
	Δ Change	+1.46	+1.32	+2.18	+1.43	+0.72	+1.86

tions. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 346–353.

Alhuzali, H. and Ananiadou, S. (2021). SpanEmo: Casting multi-label emotion classification as span-prediction. In Merlo, P., Tiedemann, J., and Tsarfaty, R., editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1573–1584, Online. Association for Computational Linguistics.

Baccianella, S., Esuli, A., and Sebastiani, F. (2010). SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In Calzolari, N., Choukri, K., Maegaard, B., Mariani, J., Odijk, J., Piperidis, S., Rosner, M., and Tapias, D., editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners.

Buechel, S., Buffone, A., Slaff, B., Ungar, L., and Sedoc, J. (2018). Modeling empathy and distress in reaction to news stories. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Buechel, S., Modersohn, L., and Hahn, U. (2020). Towards label-agnostic emotion embeddings.

Calefato, F., Lanubile, F., and Novielli, N. (2018). Emotxt: A toolkit for emotion recognition from text.

Demszky, D., Movshovitz-Attias, D., Ko, J., Cowen, A., Nemade, G., and Ravi, S. (2020). Goemotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North*. Association for Computational Linguistics.

Ekman, P. and Friesen, W. V. (1971). Constants across cultures in the face and emotion. *Journal of personality and social psychology*, 17(2):124.

Farquhar, S., Kossen, J., Kuhn, L., and Gal, Y. (2024). Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Halder, K., Akbik, A., Krapac, J., and Vollgraf, R. (2020). Task-aware representation of sentences for generic text classification. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hasan, M., Rundensteiner, E., and Agu, E. (2019). Automatic emotion detection in text streams by analyzing twitter data. *International Journal of Data Science and Analytics*, 7.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Jiang, Y., Chan, C., Chen, M., and Wang, W. (2023). Lion: Adversarial distillation of proprietary large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Kleinberg, B., van der Vegt, I., and Mozes, M. (2020). Measuring emotions in the covid-19 real world worry dataset.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Lukasik, M., Bhojanapalli, S., Menon, A. K., and Kumar, S. (2022). Teacher's pet: understanding and mitigating biases in distillation. *Transactions on Machine Learning Research*.

Mackey, A., Gauch, S., and Labille, K. (2021). Detecting fake news through emotion analysis.

Plutchik, R. (1982). A psychoevolutionary theory of emotions.

Rahman, A. B. S., Ta, H.-T., Najjar, L., Azadmanesh, A., and Gönül, A. S. (2024). Depressionemo: A novel dataset for multilabel classification of depression emotions.

Russell, J. A. and Mehrabian, A. (1977). Evidence for a three-factor theory of emotions. *Journal of research in Personality*, 11(3):273–294.

Staiano, J. and Guerini, M. (2014). Depeche mood: a lexicon for emotion analysis from crowd annotated news. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 427–433.

Tafreshi, S., De Clercq, O., Barriere, V., Buechel, S., Sedoc, J., and Balahur, A. (2021). WASSA 2021 shared task: Predicting empathy and emotion in reaction to news stories. In De Clercq, O., Balahur, A., Sedoc, J., Barriere, V., Tafreshi, S., Buechel, S., and Hoste, V., editors, *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 92–104, Online. Association for Computational Linguistics.

Toraman, C., Şahinuç, F., and Yilmaz, E. (2022). Large-scale hate speech detection with cross-domain transfer. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2215–2225, Marseille, France. European Language Resources Association.

Turcan, E., Muresan, S., and McKeown, K. (2021). Emotion-infused models for explainable psychological stress detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models.

Wullach, T., Adler, A., and Minkov, E. (2021). Fight fire with fire: Fine-tuning hate detectors using large samples of generated hate speech. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Comparing Human and Machine Generated Text for Sentiment

WingYin Ha and Diarmuid P. O'Donoghue[a]
Department of Computer Science, Maynooth University, Co. Kildare, Ireland
wingyin.ha.2019@mumail.ie, diarmuid.odonoghue@mu.ie

Keywords: Large Language Model (LLM), Parallel Corpus, Sentiment, Human Machine Comparison, Evaluation.

Abstract: This paper compares human and machine generated texts, focusing on a comparison of their sentiment. We use two corpora; the first being the HC3 question and answer texts. We present a second corpus focused on human written text-materials sourced from psychology experiments and we used a language model to generate stories analogous to the presented information. Two sentiment analysis tools generated sentiment results, showing that there was a frequent occurrence of statistically significant differences between the sentiment scores on the individual sub-collections within these corpora. Generally speaking, machine generated text tended to have a slightly more positive sentiment than the human authored equivalent. However, we also found low levels of agreement between the Vader and TextBlob sentiment-analysis systems used. Any proposed use of LLM generated content in the place of retrieved information needs to carefully consider subtle differences between the two – and the implications these differences may have on down-stream tasks.

1 INTRODUCTION

The abilities of Large Language Models (LLM) like ChatGPT are still poorly understood and greater understanding is essential in the face of widespread adoption, to ensure safe and reliable utilization. This paper uses two parallel corpora of human and machine originated text to find any similarities and notable differences between them. This paper focuses on the sentiment of these parallel texts, using five distinct parallel collections.

(Yiu *et al*, 2023) argue that LLM are cultural technologies that enhance cultural transmission. (Connell and Lynott, 2024) discussed the strengths and weaknesses of large language models to foster better understanding of human cognition. (Gibney, 2024) note that statements written in the African American English (AAE) dialect (widely spoken in the United States) have revealed strong racial biases in ChatGPT, making it more likely to associate fictionalised speakers with less-prestigious jobs and even more like to recommend the death penalty for a fictional defendant. (Mitchell, 2021) critiqued the ability of LLM to form concepts, abstractions and even make analogies.

Some application may consider machine generated texts as an alternative to retrieving text from a corpus. But this approach assumes equivalence between generated and human when text. This putative equivalence is put to the test in this paper by analysing two corpora of aligned human and machine generated texts. This work also contributes to ongoing work on model collapse (Feng *et al*, 2024) and the impact of machine generated data in training LLM.

This paper evaluates one pre-existing corpus and presents a novel corpus composed of analogous story pairs. We shall argue that these generated analogous stories offer a better mechanism to explore the innate bias contained within LLM.

We use existing technologies to investigate the output of LLM for any sentiment bias and differences between human and LLM originated text. For a comparison of Vader and TextBlob for sentiment analysis see (Bonta *et al*, 2019).

There has been a sigificant amount of recent work on comparing human and LLM generated text. (Katib *et al*, 2023; Liao et al, 2023), with some of this work focusing on detecting machine generated text in the context of plagiarism (Khalil and Erkan, 2023; Cotton *et al*, 2024). This paper differs from previous work in several regards. Firstly, this paper focuses on comparing the sentiment of texts. Secondly, we are not aware of any revious work on using the analogy

[a] https://orcid.org/0000-0002-3680-4217

Ha, W. and O'Donoghue, D.
Comparing Human and Machine Generated Text for Sentiment.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 335-342
ISBN: 978-989-758-716-0; ISSN: 2184-3228

approach to generate text, leading to the Analogy Materials Corpus (AMC) used in this paper. Finally, we compare the human and machine AMC texts using sentiment.

This paper is structured as follows. Firstly we discuss the background for comparing human and LLM generated text. We describe HC3 corpus (and each of its constituent sub-collections), before describing how the human portion of AMC coprus was compiled. We then detail how an LLM was used to generate analogous texts before presenting an analysis of the AMC texts.

We briefly describe our system before presenting and analysing our resutls on the HC3 corpus before analysing the AMC results. Finaly some conclusions and future work are discussed.

2 BACKGROUND

Widespread adoption of LLM since ChatGPT has raised concerns about its output and the presence of any hidden biases therein. Studies of LLM have shown the larger and more powerful models possess some surprising abilities, such as the ability to interpret analogical comparisons (Webb *et al*, 2023), they have shown an ability in terms of Theory of Mind (Strachan *et al*, 2024). (Ichen & Holyoak, 2024) evaluated text that they were confident was not included in any LLM training data to evaluate GPT-4's ability to detect & explain any contained metaphors. We did not follow this effort to ensure the novelty of the query text as we wish to better reflect typical usage of these LLM, which includes a combination of novel and familiar text in each query.

We argue that analogies offer a better mechanism to explore the sentiment of machine generated text. While the question-and-answer scenario restricts the range of possible responses to a prompt, generating novel analogies in contrast opens a much wider range of response types and topics. The semantic restriction that questions imposed on the range of possible answers is in effect removed by requesting the LLM to generate a comparable story both one that requires, or is even founded upon, a reasonable semantic distanced between the original and generated stories.

Thus, we argue, that generating analogous stories to a presented text imposes fewer constraints on the responses and thereby uncovers a more faithful reflection of the contents and biases contained within the LLM machine itself. Later in this paper we shall detail the Analogy Materials Corpus (AMC) that contains parallel human and machine generated text, containing analogous pairs of English texts.

3 PARALLEL CORPORA OF HUMAN AND MACHINE TEXT

This section describes two corpora of parallel human and machine generated text. Firstly, the pre-existing the Human ChatGPT Comparison Corpus (HC3) (Guo *et al*, 2023), which was produced under a question-answer scenario, by recording comparable answers to a given list of questions.

3.1 HC3 Corpus

The Human ChatGPT Comparison Corpus (HC3) is a collection of we collected 24,322 questions, 58,546 human answers and 26,903 ChatGPT answers (Guo *et al*, 2023). The corpus contains paired responses from both human experts and ChatGPT, allowing comparison of broad trends in the ability to each to generate text. Questions were grouped according to theme, including; open-domain, financial, medical, legal, and psychological areas. Their lexical analysis showed that ChatGPT uses more NOUN, VERB, DET, ADJ, AUX, CCONJ and PART words, while using less ADV and PUNCT words.

Sentiment analysis of text in (Guo *et al*, 2023) used a version of Roberta that was fine-tuned on a Twitter corpus. Additionally, their sentiment analysis focused on the collection as a whole and didn't examine the individual sub-collections. Limitations of the previous work include difficulty in reproducing the results (because of fine-tuning) and difficulty in benchmarking results against more established sentiment analysis models. Our sentiment analysis uses Vader (Hutto and Gilbert, 2014) one of the most widely used sentiment analysis models. This is compared with the newer TextBlob (Loria, 2018) model.

Table 1: Word count on the HC3 texts.

	Medicine		Finance		Open_qa		Wiki_csai	
	H	G	H	G	H	G	H	G
M	82	196	176	233	31	356	193	183
SD	46	76	160	100	19	161	124	49

The HC3_medical group contains text with strongly positive and strongly negative sentiments while the finance collection is dominated by a neutral sentiment.

The human texts contained an average of 120.5 words while the GPT texts were approximately twice that length at 241.3 words.

4 ANALOGY MATERIALS CORPUS (AMC)

Abgaz *et al* (2017) examined the characteristics of analogies between the text of publications in computer graphics. (O'Donoghue *et al*, 2015) showed how analogies can help stimulate creative thinking. Mitchell (2023) argues that large language models do not properly match human ability to form abstractions and use analogies. But recent studies (Webb *et al*, 2023) have shown that the bigger LLM models like ChatGPT possess the ability to correctly interpret analogical comparisons, including those between text stories.

In this paper we used an LLM to generate stories that are intended to be analogous to presented (human authored) stories.

As stated earlier, we see the generation of analogies as a powerful mechanism for evaluating the preferences and biases in LLM. Unlike the Question answer scenario that constrains the topic and arguably biases the expression of an answer, the hallmark of analogy is the presence a noticeable semantic difference between the presented information and its newly created analogous version.

The Structure Mapping Theory (Gentner, 1983) of analogy identifies the hallmarks of analogy as a semantic difference coupled with identifiable parallel systems of information between the two analogous scenarios.

We created the Analogy Materials Corpus [2] (AMC), composed of 169 short text stories selected from almost 40 distinct publications reporting empirical cognitive studies, including those reported by (Webb *et al*, 2023). These were first written by analogy researchers who were exploring the factors influence the human ability to interpret analogical comparisons and these materials (in the form of pairs of texts) were subsequently used on human experimental participants. Some of these experiments presented a target problem with alternate sources, to ascertain conditions that induce the expected solution in subjects. Other experiments couple a source solution with alternate target problems to see which are solved. Different participant groups are given different materials with solutions rates being studied, to ascertain different factors impacting on the analogy process. These working memory factors and the order of presentation of information (Keane, 1997) to the role of related sources on inducing general rules and their impact on subsequent reasoning (Gick and Holyoak, 1983).

Stories were selected from within these materials and Llama2 was used to generate novel source stories that were analogous to each presented text. The 7bn parameter version was used with the *temperature* set to Zero (for reproducible results) but other LLM parameters were generally left with default values. This produced parallel corpus of human and machine authored texts and this paper treats the pre-existing human written sources and the machine generated stories as a kind of parallel corpus.

For reproducibility, the *temperature* parameter was set to 0. Initial testing indicated that best results were produced by setting: *role* to *user* and *model* was set to *instruct*. Responses from Llama2 were frequently accompanied by standard pre-pended text such as *"Sure. Here is a story that is analogous to the given story:"*. Because these statements were not related specifically to the query and because they appeared in many answers, they were removed from each machine generated output.

4.1 Word Count and Vocabulary Size

The corpus contains 338 distinct text stories, in two paired collections of 169 texts each. The human texts had an average of 254.2 (SD=96.9) words, ranging in size from 67 to 882 words. The machine texts had an average size of 160.0 (SD=100.1), ranging in size from 17 to 523 words.

The average number of unique words in the human texts was 89.1 (SD=44.9) ranging from 16 to 233 unique words. The machine generated texts averaged 126.7 (SD=30.1) words, ranging in size from 49 to 228 distinct words.

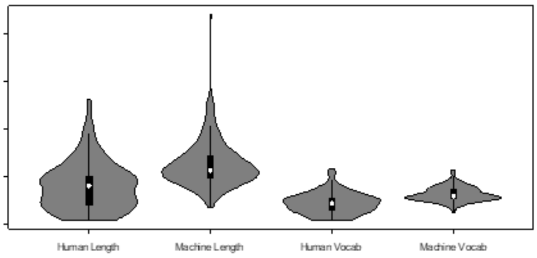


Figure 1: Machine generated texts were longer than the human texts (left) and used a larger vocabulary than the human texts.

We conclude that the LLM generated stories are highly comparable in size to the presented text. Furthermore, an *ad hoc* analysis of the generated stories suggested the presence of a semantic

[2] https://www.kaggle.com/diarmuidodonoghue/datasets

difference between the original and machine generated texts.

4.2 Part of Speech Analysis

We also performed a lexical analysis on the human and machine produced texts, as was performed in (Guo *et al*, 2003). We focus our analysis on the main lexical categories of; noun, verb, pronoun, adpositions as these are some of the lexical categories of greatest relevance to interpreting analogical comparisons. NLTK was used to perform the lexical analysis.

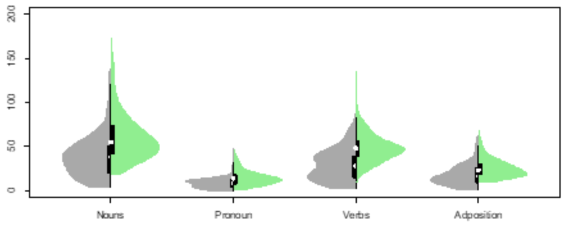


Figure 2: Lexical comparison of the human (grey) and machine (green) generated texts from the Analogy Materials Corpus.

Figure 2 depicts the number of words in each of the four analyzed lexical categories. The first split violin plot quantifies the number of nouns contained in each text. The left side of each violin (grey) quantifies the number of words in each text written by a human, while the right side of each violin (green) depicts the number of nouns in each of the machine generated texts.

These results show a high degree of similarity between the number of words in each of these categories. In this paper it was not necessary to validate whether the original and machine generated texts were in fact truly analogous to one another – or if they were merely similar to one another in some unspecified but abstract way.

4.3 Pairwise Differences

We performed a more detailed pairwise comparison of the difference in size between the human and machine originated text, with the results summarised in the table below. These again reflect the fact that machine generated texts are slightly larger than the corresponding human texts.

Table 2 details the size differences between the paired texts for each of the examined lexical categories, as well as for total number of words.

Overall we find that the machine generated texts are longer than the human texts.

Table 2: Differences between the human and ACM texts.

	Words	Adp	Noun	Pron	Verb
Mean	-94.1	-7.9	-23.9	-6.1	-19.8
SD	119.8	14.5	34.7	10.0	23.0

5 SYSTEM DESCRIPTION

A system was written in Python version 3.9.13 to determine the sentiment scores for each individual text in the HC3 corpus and in the AMC corpus. This system used the libraries; NLTK (Natural Language Tool Kit) 3.8.1 and its *sentiment.vader* library and TextBlob (*PatternAnalyser*) version 0.18. All experiments were performed on a standard laptop computer and all execution times were in the order of seconds and are not reported further.

VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool. TextBlob is a general-purpose text progressing system that includes a sentiment analysis system.

Vader returns scores ranging of -1 for the most negative sentiment and +1 for the most positive. Similariy, TextBlob returns polarity scores also in the range from -1 to 1. The following results were produced by these two systems. The two systems occasionally reveal the same insight into human and machine generated text, put frequently the two systems give somewhat different insights.

6 RESULTS AND ANALYSIS

We now present the results and analysis of the sentiment analysis of the two corpora. We begin with the HC3 results and each of its constituent collections, followed by the AMC analogy results.

6.1 HC3 - Overall

Vader (Hutto and Gilbert, 2014) and TextBlob (Loria, 2018) showed very different results on the HC3-medicine corpus. Vader identified stronger positivity on three machine generated texts, but humans showed more positivity on the other medicine collection.

Table 3 details sentiment scores for the 4 collections in HC3, with H indicating human text and G for machine generated text.

Table 3: Average Vader Sentiment on the HC3 texts.

	Medicine		Finance		Open_qa		Wiki_csai	
	H	G	H	G	H	G	H	G
Mean	.34	.08	.46	.74	.11	.38	.49	.67
SDev	.60	.79	.57	.45	.40	.55	.54	.47

Table 4 shows sentiment values generated by TextBlob for each of the HC3 collections. TextBlob scores were more neutral than the corresponding Vader scores.

Table 4: Average TextBlob Sentiment on the HC3 Corpus.

	Medicine		Finance		Open_qa		Wiki_csai	
	H	G	H	G	H	G	H	G
Mean	.14	.12	.10	.12	.05	.09	.05	.07
SDev	.46	.53	.44	.45	.23	.41	.40	.45

6.1.1 Levels of Agreement

We compared the sentiment of human and machine text for the HC3 corpus, dividing the resulting differences into three categories, as follows:

Strong Disagreement:	difference > 0.5
Disagreement:	<= 0.1 difference <= 0.5
Strong Agreement:	difference <= 0.1

This categorisation divided the overall HC3 corpus into three approximately equally sized categories, accounting for between 31% and 35% of the overall corpus in each of the three categories.

Table 5: Sentiment comparison between human and machine text using Vader.

	Medicine	Finance	Open_qa	Wiki_csai
Strong Disagree	55.05	36.49	43.15	31.21
Disagree	29.89	32.87	43.89	35.50
Strong Agree	15.06	30.64	18.98	33.38

Table 5 shows the greatest degree of dissimilarity between human and machine text in the Medicine collection, while Finance and Wiki_csai showed the greater levels of agreement in sentiment.

6.2 HC3 - Medicine

Looking more closely at HC3-medicine, Vader identified more positivity in human texts (M=0.34, SD=0.6) and more neutral sentiment in the GPT text M=0.08, but also showed machine had greater variation (SD=0.79). In contrast, TextBlob showed almost the opposite trend, with neutral scores dominating and few positive and negative scores for both human and machine text.

Figure 3 below shows the distribution of sentiment scores for these texts. Figure 3 contains two graphs; the left bar-graph depicts the Vader results while the right shows the TextBlob results. Within each graph the human results are depicted in blue while results produced on machine generated text are shown in red.

The Vader results show that human text had a larger number of neutral scores while the machine generated text had more highly negative and far more highly positive scores. The TextBlob results show a different pattern, with most scores centered on neutral sentiments. TextBlob generally showed a greater Sentiment was evident in the human written text.

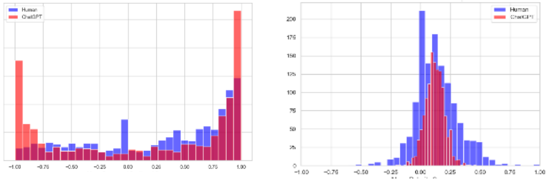


Figure 3: HC3_medicine scores of human (blue) and machine (red) generated text, using Vader (left) and TextBlob (right) polarity scores.

A Mann-Whitney analysis of the human and machine scores on HC3 - Medicine gave a two-tailed z-score of 1.07469. and the p-value is < 0.14. Thus, the difference in sentiment scores was not significant at p < 0.1.

6.3 HC3 – Finance

Figure 4 (and the subsequent diagrams in this section) also depict Vader results on the left and TextBlob on the right. Vader results show that ChatGPT text showed a far higher incidence of highly positive scores. The majority of both human and machine text showed that there were few texts with low levels of positive or with negative sentiment.

TextBlob scores indicate high levels of neutral or slightly positive sentiment on both human and machine text. However, the machine text seems to display very slightly more positive sentiment. Overall however, there appeared to be broad agreement under sentiment between human and machine generated text for this sub-collection.

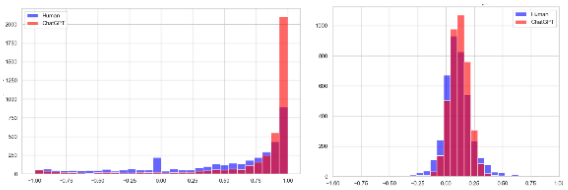


Figure 4: Vader and TextBlob found similar pattern of sentiment scores on HC3_Finance between the human and machine generated text.

A Mann-Whitney analysis of the human and machine scores on HC3_Finance gave a two-tailed z-score of -6.9308 and the p-value is < 0.0001. Thus, the difference was significant at p < 0.1. So, there is a statistically significant difference in the sentiment scores between these two collections.

6.4 HC3 – Open_qa

Vader results (Figure 5) show the human text was dominated by a neutral sentiment well the machine text was dominated by very positive sentiment. TextBlob analysis loosely echoed the dominance of neutral sentiment in the human text, while the sentiment of machine text was centered on a very slightly positive sentiment.

In this collection we see a moderate degree of agreement between Vader and TextBlob, both showing human texts to be predominantly neutral. However Vader detected a greater degree of positivity than Texblob.

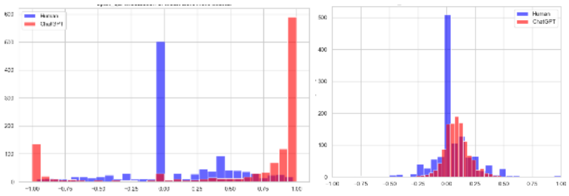


Figure 5: Comparing HC3_Open_qa scores on human and machine generated text, using Vader (left) and TextBlob (right).

A Mann-Whitney analysis of the human and machine scores on HC3_Wiki_open_qa gave a two-tailed z-score of -6.9308 and the p-value is < 0.0001. Thus, the difference was significant at p < 0.1. So, there is a statistically significant difference in the sentiment scores between these two collections.

6.5 HC3 – Wiki_csai

Figure 6 shows the sentiment analysis results on the wiki_csai collection from HC3. Vader scores the dominance of positive sentiment for both human and machine generated text. However the machine generated text exhibits a greater number of highly positive scores.

TextBlob analysis showed a similar trend between human and machine texts, centered on the dominance of very slightly positive scores. However human texts displayed a greater incidence of this sentiment then was found in the machine texts.

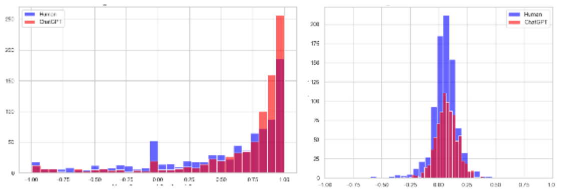


Figure 6: HC3_csai human and machine text scores, using Vader (left) and TextBlob (right).

A Mann-Whitney analysis of the human and machine scores on HC3_Wiki_csai gave a two-tailed z-score of -6.9308 and the p-value is < 0.0001. Thus, the difference was significant at p < 0.1. So, there is a statistically significant difference in the sentiment scores between these two collections.

6.6 Analogy Materials Corpus (AMC)

Finally, Vader analysis (Figure 7) of the AMC corpus revealed the machine generated text showed a large number of highly positive sentiments. While the human text showed a large positive sentiment, it also had a broader distribution of sentiment scores.

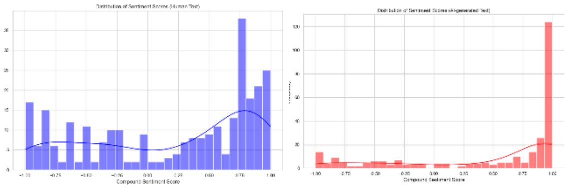


Figure 7: The machine generated AMC text showed a strong bias towards highly positive sentiment scores.

TextBlob analysis (Figure 8) indicated the human and Llama2 text both had a tendency towards a neutral sentiment. However, Llama2 text was slightly more positive than the original human texts.

While the sentiment in these results were far from identical it did show a surprising degree of agreement, given the very general nature of the task of generating analogous text. However this level of agreement should be seen in the light of the original texts being dominated by sentiment scores very close to 0 and with these scores also displaying something akin to a normal distribution.

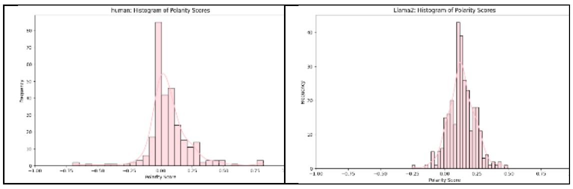


Figure 8: TextBlob polarity on human and machine AMC text.

A Mann-Whitney analysis of the human and machine scores on the AMC corpus gave a two-tailed z-score of -4.42117 and the p-value is < 0.0001. Thus, the difference was significant at p < 0.1. So, there is a statistically significant difference in the sentiment scores between these human and machine texts. This was an interesting result as generating source analogs was seen as giving a great deal of freedom to the LLM in terms of its chosen subject matter and the manner in which that was expressed.

7 CONCLUSIONS

We present a comparison between text written by humans with comparable machine generated text. The objective in this paper was to assess Large Language Machines, like ChatGPT and Llama2, for biases and significant differences that distinguish their output from human text. This paper focuses on sentiment of the text, using two established sentiment analysis systems, Vader and TextBlob, to perform the analysis.

Two corpora were used; firstly the existing Human ChatGPT Comparison Corpus (HC3) corpus, containing human and machine responses to questions. Secondly we present the Analogy Materials Corpus (AMC) containing human writes texts used in psychology experiments, with the Llama2 LLM being tasked with generating analogous texts to the presented stories.

Many instances of statistically significant differences between the sentiment of human and machine text were identified. In general, machine generated text seemed to exhibit a more positive sentiment than the comparable human text. These differences were often relatively small in magnitude, but the HC3_medicine collection showed the greatest difference in the pattern of sentiment scores.

Based on these findings we additionally conclude that any putative use of LLM generated content in the place of retrieved (human) information needs to carefully consider (the often subtle) differences between human and LLM generated content.

ACKNOWLEDGEMENTS

This publication has emanated from research supported in part by a grant from Science Foundation Ireland under Grant number 21/FFP-P/10118. For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

REFERENCES

Abgaz, Yalemisew; O'Donoghue, Diarmuid P.; Hurley, Donny; Chaudhry, Ehtzaz; Zhang, Jian Jun. Characteristics of Pro-c Analogies and Blends between Research Publications, *International Conference on Computational Creativity (ICCC),* pp 1 – 8, Atlanta, GA, USA, June 2017.

Bonta, Venkateswarlu, Nandhini Kumaresh, and Naulegari Janardhan. “A comprehensive study on lexicon based approaches for sentiment analysis.” *Asian Journal of Computer Science and Technology* 8, no. S2: 1-6. (2019).

Connell, Louise, and Dermot Lynott. “What Can Language Models Tell Us About Human Cognition?.” *Current Directions in Psychological Science* 33, no. 3: 181-189. (2024).

Cotton, Debby RE, Peter A. Cotton, and J. Reuben Shipway. “Chatting and cheating: Ensuring academic integrity in the era of ChatGPT.” *Innovations in Education and Teaching International* 61, no. 2 (2024): 228-239.

Feng, Yunzhen, Elvis Dohmatob, Pu Yang, Francois Charton, and Julia Kempe. “Beyond Model Collapse: Scaling Up with Synthesized Data Requires Reinforcement.” *arXiv preprint arXiv:2406.07515* (2024).

Gentner, Dedre. “Structure-mapping: A theoretical framework for analogy.” *Cognitive Science* 7, no. 2 (1983): 155-170.

Gibney, Elizabeth. “Chatbot AI makes racist judgements on the basis of dialect.” *Nature* 627, no. 8004: 476-477. (2024).

Gick, Mary L., and Keith J. Holyoak. “Schema induction and analogical transfer.”, *Cognitive Psychology,* 15, no. 1: 1-38 (1983).

Guo, Biyang, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. “How close is ChatGPT to human experts? comparison corpus, evaluation, and detection.” *arXiv preprint arXiv:2301.07597* (2023).

Hutto, Clayton, and Eric Gilbert. “Vader: A parsimonious rule-based model for sentiment analysis of social media text.” In *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 8, no. 1, pp. 216-225. (2014).

Ichien, Nicholas, Dušan Stamenković, and Keith Holyoak. “Interpretation of Novel Literary Metaphors by Humans and GPT-4.” In *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 46. 2024.

Katib, Iyad, Fatmah Y. Assiri, Hesham A. Abdushkour, Diaa Hamed, and Mahmoud Ragab. “Differentiating chat generative pretrained transformer from humans: detecting ChatGPT-generated text and human text using machine learning.” *Mathematics* 11, no. 15 (2023): 3400.

Keane, Mark T. “What makes an analogy difficult? The effects of order and causal structure on analogical mapping.” *Journal of Experimental Psychology: Learning, Memory, and Cognition* 23, no. 4 (1997): 946.

Khalil, Mohammad, and Erkan Er. “Will ChatGPT G et You Caught? Rethinking of Plagiarism Detection.” *In International Conference on Human-Computer Interaction*, pp. 475-487. Cham: Springer Nature Switzerland, 2023.

Liao, Wenxiong, Zhengliang Liu, Haixing Dai, Shaochen Xu, Zihao Wu, Yiyang Zhang, Xiaoke Huang et al. “Differentiating ChatGPT-generated and human-written medical texts: quantitative study.” *JMIR Medical Education* 9, no. 1 (2023): e48904.

Loria, Steven. “TextBlob Documentation.” *Release 0.15 2, no. 8*: 269. (2018)

Mitchell, Melanie. “Abstraction and analogy in AI.” *Annals of the New York Academy of Sciences* 1524, no. 1 (2023): 17-21. DOI: 10.1111/nyas.14995.

O'Donoghue, Diarmuid, Yalemisew Abgaz, Donny Hurley, and Francesco Ronzano. “Stimulating and simulating creativity with Dr Inventor.” *International Conference on Computational Creativity (ICCC),* Park City, Utah, USA, pp220-227 (2015).

Strachan, James WA, Dalila Albergo, Giulia Borghini, Oriana Pansardi, Eugenio Scaliti, Saurabh Gupta, Krati Saxena *et al.* “Testing theory of mind in large language sorrymodels and humans.” *Nature Human Behaviour*: 1-11, (2024).

Webb, Taylor, Keith J. Holyoak, and Hongjing Lu. “Emergent analogical reasoning in large language models.” *Nature Human Behaviour* 7, no. 9: 1526-1541. (2023).

Yiu, Eunice, Eliza Kosoy, and Alison Gopnik. “Transmission versus truth, imitation versus innovation: What children can do that large language and language-and-vision models cannot (yet).” *Perspectives on Psychological Science* (2023): 17456916231201401.

Enhancing LLMs with Knowledge Graphs for Academic Literature Retrieval

Catarina Pires[1][a], Pedro Gonçalo Correia[1][b], Pedro Silva[1][c] and Liliana Ferreira[1,2][d]
[1] *Faculty of Engineering, University of Porto, R. Dr. Roberto Frias, Porto, Portugal*
[2] *Fraunhofer Portugal AICOS, Rua Alfredo Allen 455/461, Porto, 4200-135, Portugal*
{up201907925, up201905348, up201907523, lsferreira}@edu.fe.up.pt

Keywords: Knowledge Graphs, Large Language Models, Knowledge Graph Augmented LLMs, Academic Literature Retrieval, Natural Language Generation, Hallucination, Prompt Engineering.

Abstract: While Large Language Models have demonstrated significant advancements in Natural Language Generation, they frequently produce erroneous or nonsensical texts. This phenomenon, known as hallucination, raises concerns about the reliability of Large Language Models, particularly when users seek accurate information, such as in academic literature retrieval. This paper addresses the challenge of hallucination in Large Language Models by integrating them with Knowledge Graphs using prompt engineering. We introduce GPTscholar, an initial study designed to enhance Large Language Models responses in the field of computer science academic literature retrieval. The authors manually evaluated the quality of responses and frequency of hallucinations on 40 prompts across 4 different use cases. We conclude that the approach is promising, as the system outperforms the results we obtained with gpt-3.5-turbo without Knowledge Graphs.

1 INTRODUCTION

Despite the remarkable success of Large Language Models (LLMs) for Natural Language Generation in recent years, it has been shown that these models will frequently generate nonsensical or inaccurate texts, a phenomenon known as hallucination (Ji et al., 2023). These models become unreliable, as they are prone to answer a user prompt in a confident tone with incorrect information. Particularly on prompts related to academic literature, LLMs commonly refer to non-existing titles, digital object identifiers (DOI), and authors or confuse information from different publications. Given the importance of accurate information in this domain, such a response from the LLM may bring no value to the user, or, worst case, be even misleading (Emsley, 2023; Goddard, 2023).

An approach to mitigate hallucinations in LLMs is to combine them with Knowledge Graphs (KG) (Pan et al., 2023). By interconnecting typed entities and their attributes in a structured way (Pan et al., 2017), KGs may be used to inform the model or restrict its output, preventing it from generating inaccurate information. In this paper, we explore the potential to improve LLMs' responses in the domain of computer science academic literature retrieval by making it query a KG to retrieve the relevant facts about publications. We introduce *GPTscholar*, a proof-of-concept system for natural language queries and answers in the same domain.

Section 2, presents and discusses similar solutions to reduce LLM hallucinations by leveraging KGs, which were applied to different domains. In Section 3, we explain the architecture and implementation details of the solution. In Section 4, we describe the experiment we conducted in order to evaluate the solution. In Section 5, the obtained results are presented. These results are discussed in Section 6. Finally, in Section 7, the main conclusions of the study are presented.

2 RELATED WORK

Despite the capabilities of LLMs, significant concerns have emerged regarding their propensity to generate non-factual or misleading content. This issue, known as the factuality problem, can lead to misunderstand-

[a] https://orcid.org/0009-0005-5000-6333
[b] https://orcid.org/0009-0002-1728-0670
[c] https://orcid.org/0009-0005-2465-2788
[d] https://orcid.org/0000-0002-2050-6178

Pires, C., Correia, P., Silva, P. and Ferreira, L.
Enhancing LLMs with Knowledge Graphs for Academic Literature Retrieval.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 343-347
ISBN: 978-989-758-716-0; ISSN: 2184-3228

ings and potentially harmful consequences, especially in domains that demand high levels of accuracy such as health, law, and finance (Wang et al., 2023).

Researchers have explored strategies to mitigate LLM hallucinations across various domains by incorporating knowledge graphs (KGs). Leveraging KGs as a source of external knowledge holds promise for enhancing LLM performance. KGs offer structured information about entities and their relationships, which aids LLMs in reasoning more effectively and significantly reduces hallucinations.

According to Agrawal, G., Kumarage, T., Alghami, Z., and Liu, H. (Agrawal et al., 2023), there are three primary approaches for leveraging Knowledge Graphs (KGs) to enhance Large Language Models (LLMs): Knowledge-Aware Inference, Knowledge-Aware Learning, and Knowledge-Aware Validation. These methodologies are distinguished by their respective stages within the retrieval pipeline architecture, particularly concerning the point at which the KG is integrated. Knowledge-Aware Inference involves incorporating KGs at the input stage to enrich the context provided to the LLM. By supplying additional relevant information, it aids the model in better comprehending the prompt, thereby reducing the likelihood of irrelevant or nonsensical outputs. Knowledge-Aware Learning focuses on embedding KGs into the training process of LLMs. Introducing factual knowledge during training enhances the model's capacity to learn and generate accurate responses, leading to more reliable outputs. Knowledge-Aware Validation establishes mechanisms to verify the LLM's outputs using KGs. By cross-referencing the generated content with factual information contained within the knowledge graph, this approach significantly improves the model's reasoning and accuracy.

This study follows the first approach, Knowledge-Aware Inference, leveraging KGs at the input stage. Martino, A., Iannelli, M. and Truong, C. (Martino et al., 2023) proposed a similar approach that aimed at reducing hallucinations and improving responses to online customer reviews. To do so, information from a KG is mapped to a templated prompt that serves as input to the LLM, providing context about the place that is being reviewed. A manual evaluation process was conducted by domain experts, who rated each response and tallied the number of correct and incorrect assertions. They concluded that the KG-enhanced LLM responses had more correct assertions, fewer incorrect assertions, and better response ratings.

Brate, R. et al. (Brate et al., 2022) leverage Wikidata to improve language models on the task of movie genre classification. SPARQL queries extract movie information from Wikidata, and the results are fed into a templated prompt to the language model. The evaluation was done on a subset of the ML25M dataset (Harper and Konstan, 2016). They concluded that the context from the KG improved the results unless too much information was given relative to the size of the language model.

The study presented is distinct from the aforementioned works in that it is the LLM that produces both the SPARQL query to the KG and the final response to the user, as well as the fact that it is applied to the domain of academic literature retrieval.

3 GPTscholar

The main idea behind the system presented here is that when a user writes a prompt, two prompts are sent to an LLM under the hood before giving the answer to the user. The first prompt queries a KG to obtain accurate information from a reliable source, while the second prompt produces an answer to the user using this information to avoid hallucinations. For the LLM, we used OpenAI's *gpt-3.5-turbo*[1]. For the KG, we used the DBLP Computer Science Bibliography (Ley, 2002), which is accessible through a SPARQL endpoint[2] and contains bibliographic information on major computer science publications, counting with over seven million publications and over three million authors. We devised a flow with five steps based on this idea, as illustrated in Figure 1.

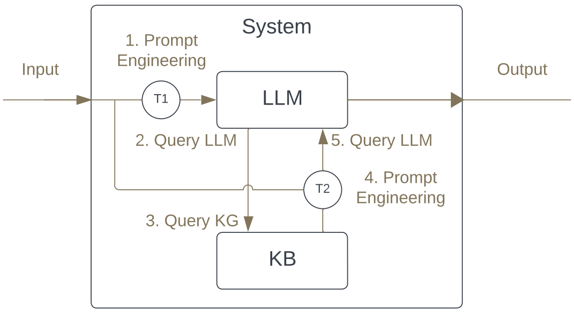


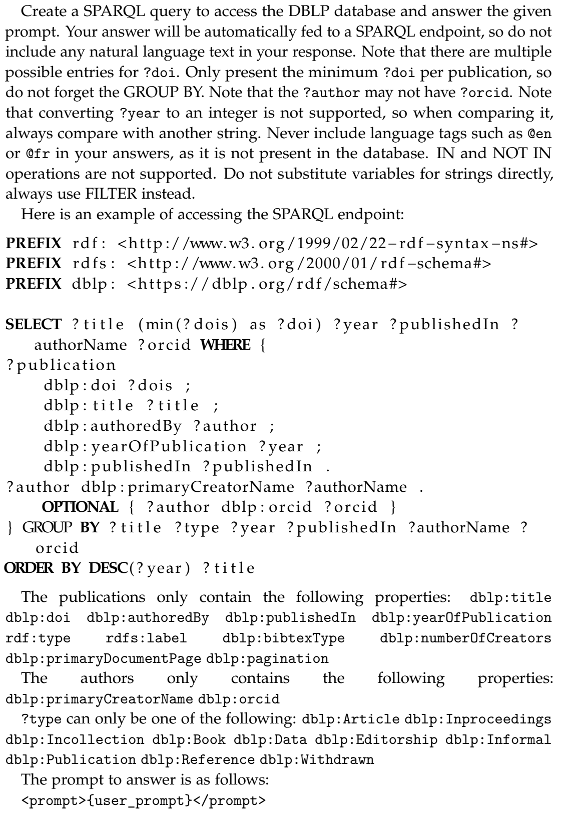
Figure 1: Flow from the initial user input to the final output, with the five steps described in Section 3. T1 and T2 represent the templated prompts to the LLM.

In the first step, the user prompt is converted into a prompt for the LLM to generate SPARQL code to get relevant information from the KG, using the template shown in Figure 2. Some instructions are given so that the LLM does not include natural language in the response, the duplicated DOIs are eliminated,

[1] https://platform.openai.com/docs/models/gpt-3-5

[2] https://sparql.dblp.org

year operations are done in a way supported by DBLP, and some constructs that aren't supported are explicitly avoided. An example of a generic SPARQL query is also given, as well as the list of properties and publication types supported.



Create a SPARQL query to access the DBLP database and answer the given prompt. Your answer will be automatically fed to a SPARQL endpoint, so do not include any natural language text in your response. Note that there are multiple possible entries for `?doi`. Only present the minimum `?doi` per publication, so do not forget the GROUP BY. Note that the `?author` may not have `?orcid`. Note that converting `?year` to an integer is not supported, so when comparing it, always compare with another string. Never include language tags such as `@en` or `@fr` in your answers, as it is not present in the database. IN and NOT IN operations are not supported. Do not substitute variables for strings directly, always use FILTER instead.

Here is an example of accessing the SPARQL endpoint:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dblp: <https://dblp.org/rdf/schema#>

SELECT ?title (min(?dois) as ?doi) ?year ?publishedIn ?
   authorName ?orcid WHERE {
?publication
    dblp:doi ?dois ;
    dblp:title ?title ;
    dblp:authoredBy ?author ;
    dblp:yearOfPublication ?year ;
    dblp:publishedIn ?publishedIn .
?author dblp:primaryCreatorName ?authorName .
    OPTIONAL { ?author dblp:orcid ?orcid }
} GROUP BY ?title ?type ?year ?publishedIn ?authorName ?
   orcid
ORDER BY DESC(?year) ?title
```

The publications only contain the following properties: `dblp:title` `dblp:doi` `dblp:authoredBy` `dblp:publishedIn` `dblp:yearOfPublication` `rdf:type` `rdfs:label` `dblp:bibtexType` `dblp:numberOfCreators` `dblp:primaryDocumentPage` `dblp:pagination`

The authors only contains the following properties: `dblp:primaryCreatorName` `dblp:orcid`

`?type` can only be one of the following: `dblp:Article` `dblp:Inproceedings` `dblp:Incollection` `dblp:Book` `dblp:Data` `dblp:Editorship` `dblp:Informal` `dblp:Publication` `dblp:Reference` `dblp:Withdrawn`

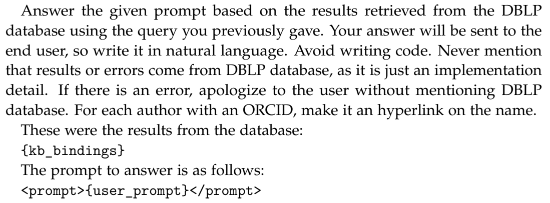
The prompt to answer is as follows:
`<prompt>{user_prompt}</prompt>`

Figure 2: Template for the first prompt to the LLM, which tells it to generate a SPARQL query based on the user prompt. *user_prompt* is replaced with the original user prompt.

In the second step, this prompt is fed to the LLM, and some processing is done to the response, such as stripping the markdown marks for code blocks, fixing the SPARQL prefixes if needed, and changing the query results limit to 100 to take into account that each publication has one result per author.

In the third step, the KG is queried, and its results are condensed in JSON format, where the authors of the same paper are merged into the same entry. The results are then truncated to the limit originally imposed by the LLM if any, or 10 entries to avoid exceeding the maximum tokens supported in a prompt to the LLM.

In the fourth step, these results are combined with the original user prompt using the template shown in Figure 3. The answer is asked to be in natural language, avoiding technical details such as code or references to the DBLP database. Instructions for the answer to give when an error occurs and to include ORCID as a link on the author's name whenever possible are also given.



Answer the given prompt based on the results retrieved from the DBLP database using the query you previously gave. Your answer will be sent to the end user, so write it in natural language. Avoid writing code. Never mention that results or errors come from DBLP database, as it is just an implementation detail. If there is an error, apologize to the user without mentioning DBLP database. For each author with an ORCID, make it an hyperlink on the name.

These were the results from the database:
`{kb_bindings}`

The prompt to answer is as follows:
`<prompt>{user_prompt}</prompt>`

Figure 3: Template for the second prompt to the LLM, which tells it to generate a response based on the KG results and the user prompt. *user_prompt* is replaced with the original user prompt, while *kb_bindings* is replaced with the results of the KG after processing.

In the final step, the resulting prompt is fed into the LLM, and its result is output to the user.

The system, named *GPTscholar*[3], includes a simple web server to showcase how the solution would be used and viewed by an end user, with the frontend implemented in React[4] and the backend in Flask[5]. This backend interfaces with the LLM and KG to reproduce the flow described in the previous paragraph.

4 EXPERIMENT

We prepared 40 prompts across 4 different use cases (10 prompts each) for academic literature retrieval to evaluate the system, using OpenAI's *gpt-3.5-turbo* without prompt engineering as a baseline. The use cases comprise the following:

1. Get publications based on the **author** (e.g. "Give me 3 papers authored by Wayne Xin Zhao.");
2. Get publications based on their **domain** (e.g. "Give me articles about generative music.");
3. Get publications based on their **attributes** (e.g. "Retrieve articles from ROBIO published before the year 2018");
4. Get publications based on **information from another publication** (e.g. "Enumerate papers written by the same authors of 'From the Semantic Web to social machines: A research challenge for AI on the World Wide Web.'").

In the experiment, our system uses OpenAI's *gpt-3.5-turbo* as the LLM and DBLP's SPARQL endpoint as the KG. We employed the schema released on October 17, 2023, which includes 61 classes, 45 object

[3] https://github.com/Goncalerta/GPTScholar

[4] https://react.dev

[5] https://flask.palletsprojects.com/en/3.0.x

Table 1: Results of the experiment for each use case. *B* denotes the baseline (gpt-3.5-turbo), while *S* denotes the GPTscholar system.

		Use Case							
		1		2		3		4	
		B	S	B	S	B	S	B	S
Prompts	Correct	2	6	1	8	2	4	0	4
	Incorrect	4	0	9	0	2	0	4	4
	No Results	4	4	0	2	6	7	6	2
	Total	10	10	10	10	10	10	10	10
Mentioned Publications	Correct	6	46	40	47	14	16	4	35
	Hallucinated Title	12	0	20	0	12	0	7	6
	Partially Incorrect	2	0	24	0	0	0	2	0
	Total	20	46	84	47	26	16	13	41

properties, and 28 datatype properties. Additionally, we used DBLP's RDF dump from December 1, 2023, which contains a total of 378,406,765 triples, including 3,384,740 person entities, 6,972,941 publication entities, and 9,355,764 external URIs.

After running the prompts through the system and the baseline, we evaluate the results manually by analyzing each response to assess its quality and the frequency of hallucinations. For each use case, we count the number of responses that:

- correctly answered the respective prompt;
- answered the prompt with incorrect information;
- found no results.

To assess the frequency of hallucinations, the total number of publications mentioned and the number of publication titles mentioned that do not exist were evaluated, as well as the number of publications whose titles exist but have incorrect information.

5 RESULTS

The results can be found in Table 1, comparing our system, *S*, to the baseline *B*. For each use case, the table shows the number of prompts where each system gave a correct result, an incorrect result, and where it did not give any result. The table also shows the number of publications mentioned by each system in the given use case. These mentions are categorized into correct publications, publications where the title has been hallucinated and does not exist, and publications that exist, but some of the details provided by the system were incorrect.

GPTscholar significantly outperformed the baseline in every use case. In total, our system got 55% of the prompts correct, while the baseline only got 12.5%. In the first three use cases (retrieving publications based on the author, their domain, and their attributes), the system did not hallucinate publications nor mention incorrect information in any prompt, while in the last use case (retrieving publications based on information from another publication), which requires more complex reasoning with indirect steps, it produced fewer incorrect results than the baseline. Even when GPTScholar couldn't answer correctly, it would more likely present no results to the user than present wrong results.

6 DISCUSSION

Given that the system queries an LLM twice and a KG once, it has more points of failure than the baseline. However, our system achieved significantly better results than the baseline since it is augmented with knowledge about publications. This suggests that the task of retrieving publications without hallucinating and without consulting a KG is more difficult for state-of-the-art LLMs than the tasks of generating a SPARQL query and generating an answer based on information present in the prompt.

We manually analyzed the intermediate step from our system in the prompts with incorrect final results. All incorrect results from the system generated incorrect SPARQL code or used SPARQL operations not supported by the SPARQL endpoint, which suggests this is the intermediate step with the biggest potential for improvement.

While GPTscholar outperformed the baseline, it had the downside of being significantly slower due to querying the LLM twice and the KG once. This can lead to a worse user experience, especially since the final output can only start being written in the last query to the LLM.

7 CONCLUSION

Leveraging KGs to enhance LLMs is a promising approach to increasing the accuracy of responses and reducing hallucinations and incorrect facts. In this document, a system is introduced to retrieve academic literature information through natural language queries and responses. After the evaluation of the solution, it can be concluded that the proposed approach hallucinates less frequently than an LLM without KGs.

For future work, different prompt templates could be tried and compared, namely to improve the generation of SPARQL code. We also envision the expansion of the DBLP knowledge base to include the abstract or even the body of the publication, which would allow the LLM to answer queries that require reasoning about the content of publications. Another possibility would be the integration of knowledge bases of academic publications in fields other than computer science. Additionally, we could analyze and assess the limitations associated with having intermediate steps, as these introduce multiple points of failure. By identifying and evaluating the error potential at each stage, we can pinpoint the most critical step and focus our efforts on improving the overall system.

REFERENCES

Agrawal, G., Kumarage, T., Alghami, Z., and Liu, H. (2023). Can knowledge graphs reduce hallucinations in llms? : A survey. *CoRR*, abs/2311.07914.

Brate, R., Dang, M.-H., Hoppe, F., He, Y., Meroño-Peñuela, A., and Sadashivaiah, V. (2022). Improving Language Model Predictions via Prompts Enriched with Knowledge Graphs ⋆. In *DL4KG@ ISWC2022*, Hangzhou, China.

Emsley, R. (2023). Chatgpt: these are not hallucinations – they're fabrications and falsifications. *Schizophrenia*, 9(1):52.

Goddard, J. (2023). Hallucinations in chatgpt: A cautionary tale for biomedical researchers. *The American Journal of Medicine*, 136(11):1059–1060.

Harper, F. M. and Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Ley, M. (2002). The dblp computer science bibliography: Evolution, research issues, perspectives. In Laender, A. H. F. and Oliveira, A. L., editors, *String Processing and Information Retrieval*, pages 1–10, Berlin, Heidelberg. Springer Berlin Heidelberg.

Martino, A., Iannelli, M., and Truong, C. (2023). Knowledge injection to counter large language model (llm) hallucination. In Pesquita, C., Skaf-Molli, H., Efthymiou, V., Kirrane, S., Ngonga, A., Collarana, D., Cerqueira, R., Alam, M., Trojahn, C., and Hertling, S., editors, *The Semantic Web: ESWC 2023 Satellite Events*, pages 182–185, Cham. Springer Nature Switzerland.

Pan, J. Z., Razniewski, S., Kalo, J.-C., Singhania, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., Biswas, R., de Melo, G., Bonifati, A., Vakaj, E., Dragoni, M., and Graux, D. (2023). Large language models and knowledge graphs: Opportunities and challenges.

Pan, J. Z., Vetere, G., Gomez-Perez, J. M., and Wu, H. (2017). *Exploiting Linked Data and Knowledge Graphs in Large Organisations*. Springer Publishing Company, Incorporated, 1st edition.

Wang, C., Liu, X., Yue, Y., Tang, X., Zhang, T., Jiayang, C., Yao, Y., Gao, W., Hu, X., Qi, Z., Wang, Y., Yang, L., Wang, J., Xie, X., Zhang, Z., and Zhang, Y. (2023). Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *CoRR*, abs/2310.07521.

An Explainable Classifier Using Diffusion Dynamics for Misinformation Detection on Twitter

Arghya Kundu and Uyen Trang Nguyen
Electrical and Computer Engineering, York University, Toronto, Canada
arghyak@yorku.ca, utn@cse.yorku.ca

Keywords: Misinformation, XAI, Social Network, Twitter.

Abstract: Misinformation, often spread via social media, can cause panic and social unrest, making its detection crucial. Automated detection models have emerged, using methods like text mining, usage of social media user properties, and propagation pattern analysis. However, most of these models do not effectively use the diffusion pattern of the information and are essentially black boxes, and thus are often uninterpretable. This paper proposes an ensemble based classifier with high accuracy for misinformation detection using the diffusion pattern of a post in Twitter. Additionally, the particular design of the classifier enables intrinsic explainability. Furthermore, in addition to using different temporal and spatial properties of diffusion cascades this paper introduces features motivated from the science behind the spread of an infectious disease in epidemiology, specially from recent studies conducted for the analysis of the COVID-19 pandemic. Finally, this paper presents the results of the comparison of the classifier with baseline models and quantitative evaluation of the explainability.

1 INTRODUCTION

Misinformation refers to inaccurate or misleading news that is propagated through various digital or analog communication channels. Misinformation is corrosive as it has a propensity to cause panic in the population and social unrest. Studies point out that people refrain from spreading misinformation if they know it to be false (Zubiaga et al., 2016). However, identifying false news is non-trivial and this motivates the effort of misinformation detection. Journalists and fact-checking websites such as PolitiFact.com can be used to track and detect misinformation. However, their underlying methodology is manual, thus being prone to poor coverage and low speed. Therefore, it is necessary to develop automated approaches to facilitate real-time misinformation tracking and debunking.

Most of the previous work related to automated misinformation detection focuses on news content, user metadata, source credibility and propagation cascades. These methods mostly do not consider or tend to oversimplify the structural information associated with misinformation propagation. However, the propagation patterns have been shown to provide useful insights for identifying misinformation.

A landmark study conducted on Twitter found that the diffusion cascades of misinformation is different from that of true information (Vosoughi et al., 2018). Specifically, misinformation propagates significantly farther, faster, deeper, and broader. Moreover, in a separate recent study (Juul and Ugander, 2021) on the same dataset used in (Vosoughi et al., 2018), the authors found that these differences in diffusion patterns on Twitter can be attributed to the “infectiousness” of the posts (tweets). They concluded that misinformation is more “infectious” than true information. While the mentioned studies provide empirical evidence that misinformation can be differentiated based on the propagation cascades and “infectiousness”, it remains unclear how it can be properly used to create verifiable automated detection mechanisms.

Additionally, modern AI systems solve complex problems but often produce unexplainable results. For misinformation detection, user trust in the model impacts their view of an article’s credibility. Explainable AI (XAI) models produce interpretable results (Mishima and Yamana, 2022). Previous XAI research on misinformation detection has mainly focused on content and social context, often overlooking propagation cascades.

This motivated us to investigate this approach further, focusing solely on diffusion patterns to identify misinformation and provide explanations based on the model’s intrinsic properties. In this study, we propose an ensemble misinformation detection model using spatio-temporal and epidemiological features of

Kundu, A. and Nguyen, U.
An Explainable Classifier Using Diffusion Dynamics for Misinformation Detection on Twitter.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 348-357
ISBN: 978-989-758-716-0; ISSN: 2184-3228

diffusion cascades with intrinsic explanation generations for users. Then, we compare the accuracy of our proposed model against five state-of-the-art misinformation detection models. Finally, we validate the explainability of our model using quantitative metrics.

The contributions of this paper are as follows:

- Firstly, this study only uses propagation patterns of social media posts to develop a misinformation detection model as opposed to most prior work which uses additional characteristics like content-based, source based and style-based methods.
- Secondly, we propose an ensemble system for misinformation detection by applying three classifiers, namely, K-nearest neighbour, decision tree and multilayer perceptron (MLP). Furthermore, the ensemble system is designed in a specific way to always provide intrinsic explainability.
- Thirdly, this paper uses temporal and spatial properties of diffusion cascades, along with features inspired by epidemiology, particularly insights from recent COVID-19 studies.

The paper is structured as follows. Section 2 details the related work. Section 3 discusses the datasets used in the study. Section 4 explains the tweet propagation structures. Section 5 discusses the methodology to build the misinformation detection system. Section 6 focuses on the model's explainability. Section 7 discusses the experimental results. Section 8 details the explainability evaluation. Finally, Section 9 concludes the paper.

2 RELATED WORK

2.1 Automatic Misinformation Detection

Automatic misinformation detection on social media platforms is grounded on the use of traditional classifiers that detect fake news deriving from the pioneering study of information credibility on Twitter (Carlos Castillo and Poblete, 2011). In following works (Xiaomo Liu and Shah, 2015) (Ma et al., 2015), different sets of unique features were used to classify whether a news is credible. Most of these prior works attempted to classify the veracity of spreading news using information beyond the text content, such as post popularity, user credibility features, and more. However, these studies did not take into account the propagation structure of a post. In this paper, we focuses on using the diffusion cascade of the posts (tweets).

Nevertheless, some studies have investigated capturing the temporal traits of a post. One study introduced a time-series-fitting model (Kwon et al., 2013), focusing on the temporal properties of a single feature – tweet volume. Another study (Ma et al., 2015) expanded upon this model by using dynamic time series to capture the variation of a set of social context features. In addition, another study (Friggeri et al., 2014) characterized the structure of misinformation cascades on Facebook by analyzing comments.

However, these studies does not effectively take into account the relevance of the spread of misinformation to that of an infectious disease. In this study we took motivation from the field of epidemiology and account for the spatio-temporal features originating from the study of the spread of infectious diseases, specifically from the recent studies conducted for the analysis of the COVID-19 pandemic.

2.2 Explainability of Models

Approaches to explainable machine learning are generally classified into two categories: intrinsic explainability and post-hoc explainability. Intrinsic interpretability is achieved by constructing self-explanatory models which incorporate interpretability directly to their structures. In contrast, the post-hoc XAI requires creating a second model to provide explanations for an existing model which is considered as a black-box. Studies have shown that intrinsic XAIs provide better explanations than post-hoc XAIs (Du et al., 2018), however they have a trade-off with accuracy. Moreover, existing XAI models for misinformation detection often overlook propagation statistics. This motivated us to design an XAI model that generates explanations solely from diffusion characteristics of the tweet. The proposed model offers both intrinsic explainability and high accuracy.

Nevertheless, evaluating XAI models remains crucial, yet due to the nascent nature of this field, consensus on explanation evaluation is lacking. A recent survey (Mishima and Yamana, 2022) highlighted that many XAI models lack standardized evaluation methods; they often rely on informal assessments or even skip evaluation altogether. In this study, we use three quantitative metrics to evaluate the explainability of our model.

3 DATASET

For evaluation of our model we use the popular publicly available datasets (Ma et al., 2017), Twitter15 and Twitter16, which have been widely adopted as standard data in the field of misinformation detection. Some important characteristics of the dataset are mentioned in Table 1.

Table 1: Basic Statistics of the datasets.

Statistic	Twitter15	Twitter16
# Users	306,402	168,659
# Tweets	331,612	204,820
Max. # retweets	2,990	999
Min. # retweets	97	100
Avg. # retweets	493	479

4 PROPAGATION STRUCTURE REPRESENTATIONS

Propagation networks of information on social media are represented in various ways. For this study we employ the following two representations,

- Hop based structure
- Time based structure

4.1 Hop Based Structure

In this type of structure, the diffusion of a post is represented as a directed acyclic graph, with the root of the tree being the source tweet and the corresponding children being the retweets.

The advantage of using this representation lies in its ability to readily leverage the spatial properties of post diffusion. Additionally, this method of representation effectively captures the user-follower relationship of tweets propagation in Twitter.

4.1.1 Analysis of the Representation

Figure 1a depicts a random news dissemination sample in hop based cascade representation. The source tweet is located at the centre of the biggest cluster and all other nodes represents the successive retweets.

The following observations were made,

- Maximum number of retweets are made directly from the source tweet.
- Most of the graphs have at least one dense cluster which does not include the source tweet i.e the tree has at least one very popular retweet.

4.2 Time Based Structure

In this cascade representation, we calculate the time delay between a retweet and its source tweet. Using this delay, the retweet is positioned on the relevant stack using a sampling time. The sampling time (d) is chosen to be 60 minutes for this study. The advantage of using this representation is the ease of using the temporal properties of the diffusion of a post. This representation effectively captures the life-cycle, popularity of a post and the amount of interactions accounted by the tweet over time.

4.2.1 Analysis of the Representation

Figure 1b depicts a random news dissemination sample in time based propagation representation.
Following are some observations:

- During the first couple of hours the tweet had the farthest spread. That is, the news penetrated with more traction in the social media.
- Most of the plots follow an approximation of power law distribution.

Table 2: Feature Categorization.

Number	Feature	Type
1	Number of Nodes	Spatial Feature
2	Total Diffusion Time	Temporal Feature
3	Total Peaks	Temporal Feature
4	Mean of timestamps delays	Temporal Feature
5	Basic Reproduction Number	Epidemiological Feature
6	Basic Transmission Rate	Epidemiological Feature
7	Super Spreaders	Epidemiological Feature
8	Growth Acceleration	Epidemiological Feature
9	Average Growth Speed	Epidemiological Feature
10	SD of Timestamps Delays	Temporal Feature
11	RMSSD of Timestamps Delays	Temporal Feature
12	Height	Spatial Feature

5 METHODOLOGY

This section details the methodology of our explainable ensemble classifier.

5.1 Feature Selection

The following features are used as referred in Table 2 along with their corresponding feature type.

5.1.1 Number of Nodes

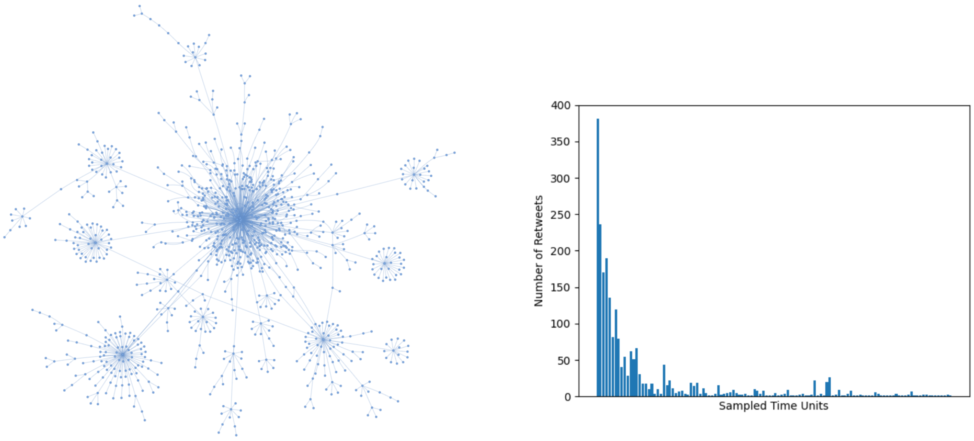
This represents the number of unique users involved in the diffusion of information. Thus, for a news dissemination pattern $N_i = \{R_i, reT_1, reT_j, .., reT_M\}$

$$\text{number of nodes} = \mathbf{card}(N_i) \tag{1}$$

where R_i is the source tweet, reT_j is a retweet and $\mathbf{card}\{S\}$ is the cardinality of the set S.

5.1.2 Total Diffusion Time

This represents the total time taken for the information to propagate in the network, i.e. the life time of



(a) Hop Based Cascade sample

(b) Time Based Cascade sample

Figure 1: Sample Propagation Structure Representations.

the news in Twitter.

$$\text{Total Diffusion Time} = t(reT_M) - t(R_i) \tag{2}$$

where t(x) is the timestamp of the tweet object x and reT_M is the last retweet

5.1.3 Total Peaks

This feature constitutes the number of nodes having timestamp value greater than the graph mean timestamp. Thus,

$$\text{TP} = \textbf{card}\{v \in V \mid G(v,E)[\text{time}] > mean(G(V,E)[\text{time}])\} \tag{3}$$

where V are the nodes in the diffusion cascade G(V,E).

5.1.4 Basic Reproduction Number

In epidemiology, the basic reproduction number is the expected number of cases directly generated by one case in a population where all individuals are susceptible to the disease (NG et al., 2006) More precisely, it is the number of secondary infections produced by an infected individual. This number is important in determining how quickly a disease will spread through a population. For this study we define the basic reproduction number (R0) as the number of retweets directly from the source tweet (R_i),

$$\text{R0} = \textbf{card}\{e \in E \mid e \in G(V,e) \wedge G(R_i,e)\} \tag{4}$$

where R_i is the source tweet and $\textbf{card}\{S\}$ is the cardinality.

5.1.5 Basic Transmission Rate

The Susceptible-Exposed-Infectious (SIR) model is used to render a simple model for the spread of a infectious disease. The basic transmission rate (denoted β) is defined as the number of effective contacts made by an infected person per unit time in a given population. In this study, we interpret basic transmission rate as the number of retweets made during the first day (Td) of the source tweet.

$$\beta = \textbf{card}\{v \in V \mid G(v,E)[\text{time}] \leqslant G(R_i,E)[\text{time}] + Td\} \tag{5}$$

where R_i is the source tweet and G(V,E) is diffusion cascade.

5.1.6 Super Spreaders

In an investigation conducted (Brainard et al., 2023) to analyze the transmission of coronavirus infections, researchers observed a significant impact on the spread of the virus were attributable to individuals identified as 'super spreaders'. Super spreaders are individuals with greater than average propensity to infect. Within this study, we delineate super spreaders (SS) as the number nodes exhibiting an edge count exceeding the average edge count.

$$\text{SS} = \textbf{card}\{v \in V \mid G(v,E) > mean(E)\} \tag{6}$$

where G(V,E) is the diffusion cascade and E are the edges.

5.1.7 Growth Acceleration

In epidemiology, Growth Acceleration is defined as the $(cases \setminus day^2)$. Recently, in a study it has been shown that Growth Speed and Growth acceleration are very effective for the analysis of the COVID-19 pandemic (Utsunomiya et al., 2020). In this study, we consider the edges i.e the retweets as the cases and define Growth Acceleration (GA) as follows,

$$GA = \sum_{i=1}^{V} \frac{1}{(G(v,E)[\text{time}] - G(R_i,E)[\text{time}])^2} \tag{7}$$

where G(V,E) is diffusion cascade and V are the nodes.

5.1.8 Average Growth Speed

Additionally, as mentioned previously Growth speed was also shown to be very effective in the analysis of the COVID-19 pandemic (Utsunomiya et al., 2020). In this study, we define Average Growth Speed (avgGS) as follows,

$$avgGS = \frac{height G(V,E)}{\text{(avg. timestamps delays)}} \tag{8}$$

where (avg. timestamps delays) is the average of all the timestamp delays and *height* of $G(V,E)$ is the length of the longest path from the root to the farthest node in the diffusion tree.

5.1.9 Standard Deviation of Timestamps Delays

Using this feature we try to take into account the measure of the spread of values from the mean.

$$\sigma = \sqrt{\frac{1}{V-1}\sum_{i=1}^{V}(t_i - \bar{t})^2} \tag{9}$$

where t are the individual timestamp delays of each retweet from the source tweet.

5.1.10 RMSSD of Timestamps Delays

We also consider the root mean square of successive differences between retweet timestamps (RMSSD). In medical science, RMSSD is considered the primary time domain measure used to estimate the vagally mediated changes (Minarini, 2020). RMSSD reflects the peak-to-peak variance in a time series data. As mentioned in subsection 4.2.1, during the initial stages of propagation, claims exhibit the widest spread, with minimal successive differences between retweets. Consequently, we integrated RMSSD as a feature in our model to capture early-hour changes in news dissemination flow.

$$\text{RMSSD} = \sqrt{\text{mean}\{\text{diff}\{t1, t2, .., tN\}^2\}} \tag{10}$$

where t are the individual timestamp delays of each retweet from the source tweet.

5.1.11 Height

Represents the length of the path from source tweet to its farthest retweet node.

$$\text{Height} = \sum_{R_i}^{reT_n} 1 \tag{11}$$

where R_i is the source tweet and reT_n is the farthest retweet.

5.2 Data Preparation

Firstly, The datasets contained four annotations namely true rumours, non-rumours, false rumours and unverified rumours. As our study focuses on binary classification, we re-annotated to two class labels namely, true and fake news and disregarded the unverified rumours. Secondly, we normalized the features by scaling and translating. We used the Min Max Normalization method. Finally, For a fair comparison,we randomly split the datasets into 80% for training and 20% for testing.

5.3 Classification Model

For this study we used a voting classifier. A voting classifier is a ensemble machine learning classifier that trains various base models and predicts on the basis of aggregating the findings of each base estimator. Voting classifiers has been shown to reduce the aggregate errors of a variety of the base models and increase final accuracy. The aggregating criteria used in this study is hard voting which is the combined decision of the class label that has been predicted most frequently by the classification models. The base models used are as follows, refer Figure 2 :

- KNN Classifier
- Decision Tree Classifier
- Multi-layer Perceptron classifier

Thus, the predicted class label $\hat{y}$ of our proposed classifier is as follows,

$$\hat{y} = \text{mode}\{C_1(x), C_2(x), C_3(x)\} \tag{12}$$

where $C_i(x)$ is the predicted class label of classifier i.

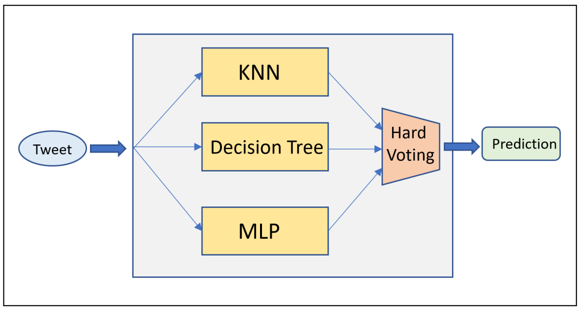


Figure 2: Voting Classifier Flowchart.

6 EXPLANABILITY OF MODEL

Research shows that intrinsic explainable AI (XAI) provide better explanations than post-hoc XAIs (Du et al., 2018), though sometimes with reduced accuracy. Our method balances intrinsic model explanations via KNN and decision tree (DT) classifiers while maintaining high accuracy.

The following two methods were used to accomplish this,

- Firstly, the unique design of the ensemble classifier asserts that at least one intrinsic explainable classifier is part of the final aggregated vote. That is, in the best case scenario both the intrinsic explainable classifiers (KNN or DT) have the same predicted label. Whereas, in the average/worst case scenario along with MLP classifier either KNN or DT classifier has the same predicted label which can be used to provide intrinsic explainability.
- Secondly, we added MLP to balance the accuracy-explainability trade-off in intrinsic XAIs. Although, MLP is not an intrinsic explainable algorithm on its own, the combination the three classifiers provides intrinsic explainability along with high accuracy.

6.1 KNN Classifier

In system interpretability, KNN relies on similarity and distance, making it inherently interpretable as the nearest neighbors provide explanations.

For providing human readable explanations for a given prediction, we employed the following steps:

1. Collect nearest K neighbours of considered point (*P*).
2. Filter out same-class neighbors of *P*, which are inherently higher in number.
3. Project (P_{new}) using arithmetic mean of filtered points.
4. Get the four highest correlated features between P_{new} and *P* using Manhattan distance.
5. Display the number of nearby same class label neighbours and the highest correlated features.

6.2 Decision Tree Classifier

A decision tree provides a hierarchy of very specific questions and predicts outcomes based on decision rules (if-then-else rules). The answer to one question guides the prediction process down various branches of the tree. At the bottom of the tree is the prediction. Hence, for interpretations, we review decisions by traversing top-to-bottom tree paths and noting question responses for explanations. To this direction we used the following steps,

1. Fetch the decision rules from the classifier.
2. Use the rules to showcase the answers the specific rule addresses.
3. Every rule corresponds to one feature, delivering a local explanation for that feature's value.
4. For a given point (*P*), traversing the decision tree from top to bottom reveals explanations for the predicted class label. Inherently the number of the explanations is the depth of the decision tree.

7 EVALUATION OF CLASSIFICATION MODEL

In this section we discuss the results of the individual and ensemble classifiers. We used Twitter16 dataset for selecting the parameters and Twitter15 for testing.

7.1 Individual Classifiers

7.1.1 KNN Classifier

The number of neighbors used in this model is ten. Table 3 shows the results from the k-nearest neighbors classifier with varying hyper-parameters. From the table, we can see that the model using Manhattan as the distance metric performs the best, achieving the highest accuracy and precision, along with a good overall recall. This is rational as studies have shown that Manhattan distance (L1 norm) ususally performs better than common distance measures in the case of high dimensional data.

Table 3: Results of the KNN classifier on Twitter16.

Distance metric	Accuracy	Precision	Recall
Cosine	0.8123	0.8436	0.8787
Manhattan	**0.8129**	**0.8591**	0.8865
Correlation	0.8045	0.7899	**0.8934**
Euclidean	0.8104	0.8087	0.8799
BrayCurtis	0.7903	0.7832	0.8811

7.1.2 Decision Tree Classifier

The maximum depth of the tree is chosen to be three, in order to reduce computational complexity. Table 4 depicts the results with varying hyper-parameters. It can be observed that the results are better with the entropy splitting criterion. However, the accuracy is almost the same for both the splitting methods. This is reasonable as the internal working of both the splitting methods are very similar. Nevertheless, for the ensemble model we choose the entropy criterion.

Table 4: Results of the Decision Tree classifier on Twitter16.

Splitting criterion	Accuracy	Precision	Recall
Gini	0.8117	0.8548	0.8676
Entropy	**0.8123**	**0.8679**	**0.8815**

7.1.3 Multi-Layer Perceptron Classifier

The MLP classifier has been configured with two hidden layers containing (5,2) units respectively. Limited-memory BFGS (lbfgs) algorithm has been used for weight optimization as it converges faster and performs better for small datasets. Table 5 depicts the results. It can be observed from the results that the accuracy and recall is highest with the Sigmoid activation function. As the number of hidden layers is very low in this model the vanishing gradient problem does not play a significant role and hence the accuracy using sigmoid function is higher compared to other activation functions.

Table 5: Results of the MLP classifier on Twitter16.

Activation function	Accuracy	Precision	Recall
Tanh	0.7945	0.7712	**0.9117**
Sigmoid	**0.8231**	**0.8574**	0.8905
ReLU	0.8117	0.8419	0.8620

7.2 Ensemble Classifier

The results for the individual classifiers are shown in Table 6 with the optimum parameter configurations. Firstly, It can be observed that MLP classifier has the highest accuracy of 82.31%, however the precision is low with 0.8574.

Table 6: Results of the individual classifiers on Twitter16.

Type	Accuracy	Precision	Recall
KNN classifier	0.8129	0.8591	0.8865
MLP classifier	**0.8231**	0.8574	**0.8905**
Decision tree classifier	0.8123	**0.8679**	0.8815

Secondly, the KNN classifier also has a high accuracy of 81.29% with the low recall and precision. Finally, the precision is highest in the case of the decision tree classifier with 0.8679, with an accuracy almost similar to that of the KNN classifier. Thus, it can be reasoned that a combination of these three classifiers might produce better results. This motivated us to implement an ensemble voting classifier for this study.

Table 7 depicts the results for the Voting Classifier.

Table 7: Results of the Voting Classifier on Twitter16.

Type	Accuracy	Precision	Recall
Voting Classifier	**0.8522**	**0.8843**	**0.8917**

It can be inferred from Table 7 that the accuracy has increased to 85.22 % with the use of the voting classifier. Ensemble methods like the voting classifier are ideal for reducing the variance in models, thereby increasing the accuracy of predictions. The variance is eliminated when multiple classifiers are combined to form a single prediction. Additionally, it can be observed that the precision and recall of the voting classifier are also high with 0.8843 and 0.8917 respectively. From our experiments, it can be reasoned that the voting ensemble outperforms all the individual models.

7.3 Baseline Model Comparison

We compared our proposed model with the following five state-of-the-art misinformation detection models,

1. CSI (Ruchansky et al., 2017): A misinformation detection model that captures temporal patterns using an LSTM to analyze user activity and calculates user scores.
2. tCNN (Yang et al., 2023): a modified convolution neural network that learns the local variations of user profile sequence, combining with the source tweet features.
3. CRNN (Liu and Wu, 2018): a state-of-the-art joint CNN and RNN model that learns local and global variations of retweet user profiles, together with the resource tweet.
4. dEFEND (Shu et al., 2019): a state-of-the-art co-attention-based misinformation detection model that learns the correlation between the source article's sentences and user profiles.
5. GCAN (Lu and Li, 2020): a state-of-the-art graph-aware co-Attention network based misinformation classifier that uses user profiles metadata, news content and propagation pattern.

Table 8 compares our approach to the industry standards. It can be inferred that our proposed model outperforms most of the state-of-the-art approaches on both datasets in terms of accuracy while attaining highest precision and recall. In particular, our model achieves an accuracy of 84.47% and 85.22% on the datasets respectively. Although GCAN achieved the highest accuracy, the precision and recall are low due to the class imbalance in the datasets, where GCAN favors the majority class, leading to higher accuracy but poorer minority class detection. Whereas, our model received at par accuracy with GCAN with higher precision and recall. Furthermore, GCAN in addition to propagation pattern uses the user profile metadata and tweet content, which might not always be available in real-world scenarios. Whereas, our model solely uses the diffusion pattern to create a classifier.

Table 8: Experimental results on Twitter15 (T15) and Twitter16 (T16) datasets.

Method	Recall		Precision		Accuracy	
	T15	T16	T15	T16	T15	T16
tCNN	0.5206	0.6262	0.5199	0.6248	0.5881	0.7374
CRNN	0.5305	0.6433	0.5296	0.6419	0.5919	0.7576
CSI	0.6867	0.6309	0.6991	0.6321	0.6987	0.6612
dEFEND	0.6611	0.6384	0.6584	0.6365	0.7383	0.7016
GCAN	0.8295	0.7632	0.8257	0.7594	**0.8767**	**0.9084**
Our model	**0.8512**	**0.8917**	**0.8568**	**0.8843**	0.8447	0.8522

7.4 Ablation Study

To study the contribution of each feature type towards the ensemble classifier, we carry out ablation experiments. The results are shown in Table 9. The ablation experiments include the following three variants:

- w/o Spatial: Removing the spatial features of the ensemble classifier.
- w/o Temporal: Removing the temporal components of the ensemble classifier.
- w/o Epidemiological: Removing the epidemiological features of the ensemble classifier.

Table 9: Results of the Ablation experiments using Twitter16.

Type	Accuracy	Precision	Recall
w/o Spatial	0.8213	0.8229	0.8078
w/o Temporal	0.8256	0.8594	0.8810
w/o Epidemiological	0.7714	0.7803	0.8276
Voting classifier	**0.8522**	**0.8843**	**0.8917**

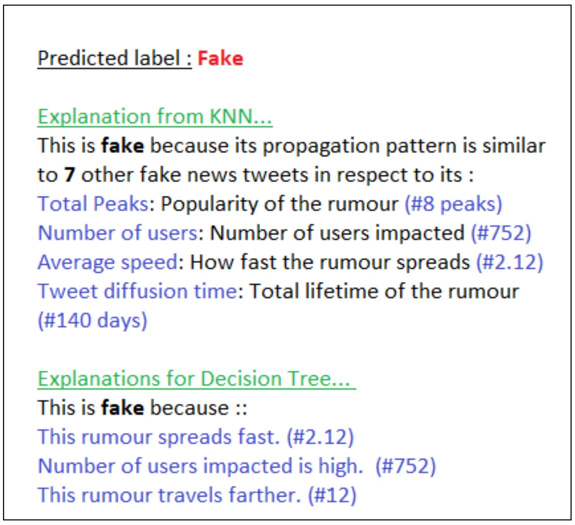
From Table 9, we can observe that all ablation variants drop some accuracy compared with the primary model. Specifically, when removing the spatial features, the accuracy drops by 3.1%, the precision and recall also dropped. The replacement of the temporal features caused the accuracy to decrease by 2.7% with lower precision and recall. However, the accuracy drop was most significant when the epidemiological features were removed, accounting to 8.1% along with lowest precision and recall. This corroborates that epidemiological features inspired from the study on COVID-19, play an essential role for misinformation detection using propagation cascades. In conclusion, overall the primary model, with the three component types involved, provides a better choice compared to the ablation variants.

8 EVALUATION OF EXPLAINABILITY OF THE MODEL

Evaluation of an XAI model essential, as it provides a way to understand its practical implication.

8.1 Sample Explanation

Figure 3 displays the explanations generated by our model on a random data point (*P*), where KNN classifier and DT classifier had the same predicted class label. We can observe that, three explanations were generated for the DT classifier, which is logical as the depth of the tree was three. Furthermore for point *P*, the KNN classifier interpretations were made from the seven nearby fake tweets out of the ten neighbours. An interesting observation can also be made that explanations for both the classifiers almost correspond for the same statistical properties of the propagation cascade.



Predicted label : **Fake**

Explanation from KNN...

This is **fake** because its propagation pattern is similar to **7** other fake news tweets in respect to its :

Total Peaks: Popularity of the rumour (#8 peaks)

Number of users: Number of users impacted (#752)

Average speed: How fast the rumour spreads (#2.12)

Tweet diffusion time: Total lifetime of the rumour (#140 days)

Explanations for Decision Tree...

This is **fake** because ::

This rumour spreads fast. (#2.12)

Number of users impacted is high. (#752)

This rumour travels farther. (#12)

Figure 3: Explanation generated for a random sample (*P*).

8.1.1 Metrics Used

We evaluated the model's interpretability using the three metrics mentioned below. These are extensions of three metrics used in (ElShawi et al., 2021) for evaluating interpretability frameworks like LIME, SHAP, LORE and more.

- *Stability*: Similar instances should have similar explanations.
- *Separability*: Different instances should yield different explanations.
- *Identity*: Identical instances must produce identical explanations.

For measuring the metrics we randomly select 100 data points and create the testing dataset using their class labels and generated explanations. The stability metric is measured by applying K-means clustering with two clusters to group explanations in the testing dataset. For simplicity, we use the three explanations generated by the decision tree (DT), converting each explanation string into a unique numerical value to form an integer array. The assigned cluster labels are then compared with the predicted class labels to evaluate whether instances of the same class have similar explanations. To measure the separability metric, two subsets S1 and S2 of the testing dataset are selected corresponding to different class labels. Then, for each instance in S1, its explanation is compared with all other explanations of instances in S2. If the explanation have no duplicates, it satisfies the separability metric. Finally, the identity of the explanations offered by the various deterministic techniques may be easily measured theoretically. The explanations generated by the decision tree is rule based thus conforming to complete identity conservation. Additionally, due to the nature of KNN alrogithm identical instances will have identical explanations.

8.1.2 Results

The experimental findings can be seen in Table 10. The figures in this table show the percentage of instances that meet the specified metrics. From the table we can infer that identity metric is 100%, as identical instances will have a similar explanations. The stability is very high, thus conforming that instances with same class labels have comparable interpretations. Finally, the separability is also very high, thus acknowledging that dissimilar instances have dissimilar explanations.

9 CONCLUSION

This paper demonstrates the effectiveness of an ensemble-based classifier using a tweet's diffusion pattern for accurate misinformation detection. We improve the classification by using features inspired by epidemiology and recent COVID-19 research, while providing understandable predictions. The intrinsic explanations help users to understand the predicted class label without compromising accuracy.

Future work will focus on the following areas:

- Incorporating statistical and qualitative measures to evaluate the results and generated explanations.
- Expanding the model's applicability to other social networks such as Instagram and Facebook.
- Investigate and document how hyperparameters, such as the value of k in k-NN, sampling rate, affect model performance.
- Conduct deeper analysis on the consistency and comparability of explanations generated by different models (e.g., k-NN vs. DT).

Table 10: Metrics for the evaluation of explanations.

Metric	Score
Stability	89%
Separability	97%
Identity	100%

REFERENCES

Brainard, J., Jones, N. R., Harrison, F. C., Hammer, C. C., and Lake, I. R. (2023). Super-spreaders of novel coronaviruses that cause sars, mers and covid-19: a systematic review. *Annals of Epidemiology*, 82:66–76.e6.

Carlos Castillo, M. M. and Poblete, B. (2011). Information credibility on twitter. page 675–684.

Du, M., Liu, N., and Hu, X. (2018). Techniques for interpretable machine learning.

ElShawi, R., Sherif, Y., Al-Mallah, M., and Sakr, S. (2021). Interpretability in healthcare: A comparative study of local machine learning interpretability techniques. *Computational Intelligence*, 37(4):1633–1650.

Friggeri, A., Adamic, L., Eckles, D., and Cheng, J. (2014). Rumor cascades. *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014*, pages 101–110.

Juul, J. L. and Ugander, J. (2021). Comparing information diffusion mechanisms by matching on cascade size. *Proceedings of the National Academy of Sciences*, 118(46).

Kwon, S., Cha, M., Jung, K., Chen, W., and Wang, Y. (2013). Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108.

Liu, Y. and Wu, Y.-F. (2018). Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Lu, Y.-J. and Li, C.-T. (2020). GCAN: Graph-aware co-attention networks for explainable fake news detection on social media. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Ma, J., Gao, W., Wei, Z., Lu, Y., and Wong, K.-F. (2015). Detect rumors using time series of social context information on microblogging websites.

Ma, J., Gao, W., and Wong, K.-F. (2017). Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Minarini, G. (2020). Root mean square of the successive differences as marker of the parasympathetic system and difference in the outcome after ans stimulation. In Aslanidis, T., editor, *Autonomic Nervous System Monitoring*, chapter 2. IntechOpen, Rijeka.

Mishima, K. and Yamana, H. (2022). A survey on explainable fake news detection. *IEICE Trans. Inf. Syst.*, E105.D(7):1249–1257.

NG, B., K, G., B, B., and Caley P, Philp D, M. J. (2006). Using mathematical models to assess responses to an outbreak of an emerged viral respiratory disease. *National Centre for Epidemiology and Population Health*.

Ruchansky, N., Seo, S., and Liu, Y. (2017). Csi: A hybrid deep model for fake news detection. pages 797–806.

Shu, K., Cui, L., Wang, S., Lee, D., and Liu, H. (2019). Defend: Explainable fake news detection. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 395–405, New York, NY, USA. Association for Computing Machinery.

Utsunomiya, Y. T., Utsunomiya, A. T. H., Torrecilha, R. B. P., de Cássia Paula, S., Milanesi, M., and Garcia, J. F. (2020). Growth rate and acceleration analysis of the covid-19 pandemic reveals the effect of public health measures in real time. *medRxiv*.

Vosoughi, S., Roy, D., and Aral, S. (2018). The spread of true and false news online. *Science*, 359:1146–1151.

Xiaomo Liu, Armineh Nourbakhsh, Q. L. R. F. and Shah, S. (2015). Real-time rumor debunking on twitter. page 1867–1870.

Yang, Y., Zheng, L., Zhang, J., Cui, Q., Li, Z., and Yu, P. S. (2023). Ti-cnn: Convolutional neural networks for fake news detection.

Zubiaga, A., Liakata, M., Procter, R., Wong Sak Hoi, G., and Tolmie, P. (2016). Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PLOS ONE*, 11(3):1–29.

A Network Learning Method for Functional Disability Prediction from Health Data

Riccardo Dondi[1] [a] and Mehdi Hosseinzadeh[1,2] [b]

[1] *Università degli Studi di Bergamo, Bergamo, Italy*

[2] *University of Calabria, Rende(CS), Italy*

riccardo.dondi@unibg.it, m.hosseinzadeh@unibg.it

Keywords: Network Analysis, Disability Classification, Learning Algorithms, Healthcare Analytics, Graph Data Mining.

Abstract: This contribution proposes a novel network analysis model with the goal of predicting a classification of individuals as either 'disabled' or 'not-disabled', using a dataset from the Health and Retirement Study (HRS). Our approach is based on selecting features that span health indicators and socioeconomic factors due to their pivotal roles in identifying disability. Considering the selected features, our approach computes similarities between individuals and uses this similarity to predict disability. We present a preliminary experimental evaluation of our method on the HRS dataset, where it shows an enhanced average accuracy of 62.48%.

1 INTRODUCTION

A relevant problem for supporting elderly individuals is the prediction of their health status. In this context, it is extremely valuable to predict the risk of functionally disability, in order to provide the needed support (Stuck et al., 1999).

Current studies demonstrate the advancements in the use of knowledge graphs and network analysis in the fields of biology and healthcare (Hosseinzadeh, 2020; Hosseinzadeh et al., 2022) and (Pham et al., 2018; Tao et al., 2020; Wang et al., 2020; Pham et al., 2022; Cui et al., 2023). In this context, the development of prediction models based on graphs in healthcare is essential for improving disease diagnosis and reducing human error. In particular, (Wang et al., 2020) developed a predictive model that classifies individuals according to their disability risk, using a network to represent disease progression. (Tao et al., 2020) introduced a novel classification model that uses a heterogeneous knowledge graph for conceptualizing medical domain knowledge. The developed model was used to forecast possible health risks for patients using data from the National Health and Nutrition Examination Survey (NHANES). (Cui et al., 2023) provided a comprehensive review of knowledge graph applications in healthcare, highlighting the instruments, applications, and possibilities for improved understanding and prediction of complex medical scenarios.

Our contribution aims to build a prediction method inspired by approaches for classifying individuals based on their risk of becoming disabled. Our approach proposes a novel network analysis model based on the features presented in a dataset from the Health and Retirement Study (HRS)[1] (Health and Study, 2008). We select some features that span health indicators and socioeconomic factors due to their pivotal roles in identifying disability. Each individual is then represented as a vector on the selected features and similarity between two individuals is evaluated by computing a function of the difference in the values of the features. A user is then assigned to the category ('disabled', meaning high-risk of becoming disable, or 'non-disabled', meaning low-risk of becoming disable) based on the average similarity with each single group (disable individuals and non disable individuals).

We present some preliminary experimental evaluation of our method on the HRS dataset. We select 10 samples of 100 individuals extracted randomly from the HRS dataset and on each of this sample we evaluate the performance of our method. The method shows a moderate accuracy in the classification (average accuracy of 62.48%).

The remainder of the paper is organized as follows. In section 2, we start by introducing some defi-

[a] https://orcid.org/0000-0002-6124-2965

[b] https://orcid.org/0000-0003-3275-6286

[1] https://hrs.isr.umich.edu.

Dondi, R. and Hosseinzadeh, M.
A Network Learning Method for Functional Disability Prediction from Health Data.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 358-362
ISBN: 978-989-758-716-0; ISSN: 2184-3228

nitions and by providing the formal definition by formally introduces the research problem. In Section 3, we present the computational approach used to address the research problem, including the construction and application of the bipartite graph model. In Section 4, we present the results from the experimental analysis, discussing the implications and insights gained from applying our methodology to the HRS dataset. Finally, In Section 5, we conclude the main outcomes with some future directions.

2 DEFINITIONS AND RESEARCH PROBLEM

In this section, we define the main concepts needed for our methodology, mainly graph theory and network analysis, and we present the formal research problem our study addresses.

All the graphs we consider in this paper are undirected. A graph G is defined as a pair $G = (V, E)$, where V is a set of nodes and E is a set of edges. Each edge $e \in E$ is an unordered pair $\{v, w\}$, indicating a connection between nodes v and w in V (Bondy and Murty, 2008). We mainly consider bipartite graph, defined in the following.

Definition 1. *A graph is* bipartite *if there exist two disjoint sets $X \subseteq V$ and $Y \subseteq V$ such that $X \uplus Y = V$*[2]*, and every edge in E links a node of X and a node of Y.*

We now provide the definition of the neighborhood of a node, which is a relevant concept needed for describing our method.

Definition 2. *Given a graph $G = (V, E)$ and a node $v \in V$, the neighborhood of v in G is defined as $N_G(v) = \{u \in V \mid \{u, v\} \in E\}$.*

We now introduce a specific bipartite weighted graph we consider to represent the relations between features and individuals, called *Feature-Individual Classification Network* (FICN).

Definition 3. *The Feature-Individual Classification Network is a bipartite weighted graph denoted as $G = (V, E, W_E, W_F)$ where:*

- *V is the set of nodes, partitioned into two disjoint subsets V_I and V_F. The subset V_I represents individuals, and V_F represents features.*
- *$E \subseteq V_I \times V_F$ is the set of edges, each connecting an individual node in V_I and a feature node in V_F.*
- *$W_E : E \to \mathbb{R}^+$ is a weight function for the edges, where each weight represents the value of an individual for a specific feature (these weights are derived from the input data, their computation is described later).*
- *$W_F : V_F \to \mathbb{R}^+$ is a weight function for the feature nodes, assigning a weight to each feature; this weight represents the relevance of the specific feature for classification. Unlike W_E, W_F is not obtained from the input data but it is computed using an optimization technique (that we will describe in Section 3).*

Given an edge $uv \in E$, $w_E(uv)$ denotes the edge weight of uv. Given a node $u \in V_F$, $w_F(u)$ denotes the feature weight of node u.

Note that $G = (V, E, W_E, W_F)$ is not a complete bipartite graph as some edges may not defined, reflecting possible missing data of individuals.

The classification of an individual is based on the similarity value between the node related to the individual, and the nodes of the Feature-Individual Classification Network, as defined in the following.

Definition 4. *Let $G = (V, E, W_E, W_F)$ be a Feature-Individual Classification Network, where $V = V_I \uplus V_F$. Consider a candidate node c (note that $c \notin V$) such that c is connected with a subset F_c of feature nodes (hence $F_c \subseteq V_F$). The similarity measure $\sigma(c, G)$ of c with respect to G is defined as:*

$$\sigma(c, G) = \frac{\sum_{f \in F_c} w_I(f) z_{cf}}{\sum_{f \in F_c} w_I(f)},$$

where $w_I(f)$ is the weight function of individual I for feature f, and for each $f \in N_G(c)$, z_{cf} is the z-score [3] *of $w_E(cf)$ in the following set:*

$$\{w_E(uf) : u \in N_G(f)\}.$$

Note that the similarity measure $\sigma(c, G)$ is based on the weight of the features that are computed by the optimization technique described in Section 3.

2.1 Research Problem

Next we describe our problem. Given two disjoint sets of individuals (‘disable’ and ‘non-disable’), we define a Feature-Individual Classification Network for each of these sets.

$$G_1 = (V_{I_1} \uplus V_F, E_1, W_{E_1}, W_{F_1})$$

represents the graph consisting of the set V_{I_1} of individuals identified as ‘disabled’.

$$G_2 = (V_{I_2} \uplus V_F, E_2, W_{E_2}, W_{F_2})$$

[2] We recall that $\uplus$ denotes the disjoint union of sets.

[3] The z-score for feature f is computed as $z_{cf} = \frac{v_{cf} - \mu_f}{\sigma_f}$, where v_{cf} is the value of feature f for the candidate node c, μ_f is the mean value of feature f across all the neighbor nodes $N_G(f)$ in G that have f, and σ_f is the standard deviation of f among the same nodes.

represents the set V_{I_2} of 'non-disabled' individuals.

We introduce now the main research problem we consider in this paper, which aims to compute the feature weights in order to optimize the classification.

Problem 1. *Weight Feature Optimization Problem.*
Input: *Two Feature-Individual Classification Networks $G_1 = (V_{I_1} \uplus V_F, E_1, W_{E_1}, W_F)$ and $G_2 = (V_{I_2} \uplus V_F, E_2, W_{E_2}, W_F)$.*
Output: *Compute $W_F : V_F \to \mathbb{R}^+$ for the feature nodes so that the classification in 'disabled' or 'non-disabled' individuals is optimized.*

Assume that the feature weights are known. For each individual i_{class} to be classified, we compute the similarity $\sigma(i_{\text{class}}, G_i)$, with $i \in \{1,2\}$. After calculating these similarity scores, the overall classification of i_{class} as either 'disabled' or 'non-disabled' is obtained by aggregating these scores:

$$\text{Classification}(i_{class}) = \arg \max_{G \in \{G_1, G_2\}} \sigma(i_{\text{class}}, G).$$

This aggregate score assigns i_{class} to the group with the highest computed similarity score. So, in order to apply the classification, we need to compute the feature weights.

The Weight Feature Optimization Problem is solved by considering a training set of individuals for which we already know the classification and then compute the value of the weights W_F in order to maximize the correct classification. Formally, we consider the following problem.

Problem 2. *Training Weight Feature Optimization Problem.*
Input: *Two Feature-Individual Classification Networks $G_1 = (V_{I_1} \uplus V_F, E_1, W_{E_1}, W_F)$ and $G_2 = (V_{I_2} \uplus V_F, E_2, W_{E_2}, W_F)$; two sets X_1, X_2 of candidate nodes that are classified as disable and non-disable, respectively.*
Output: *Compute $W_F : V_F \to \mathbb{R}^+$ so that the number of individuals of $X_1 \uplus X_2$ correctly classified is maximized.*

3 METHODOLOGY

In the preliminary phase of our study, we define our classification criteria based on the established guidelines from (Li et al., 2017; Rossetti and Cazabet, 2018). Specifically, an individual is considered 'disabled' when encounters two or more difficulties in any of the six identified Activities of Daily Living (ADL). In order to address the classification problem, we structure our data into two disjoint sets, i.e. a training set and a test set.

The training set consists of distinct subsets for different phases of the model development process. The first subset of the training set consists of (1) 250 individuals randomly selected from the 'disabled' individuals and used to build the Feature-Individual Classification Network G_1 representing 'disabled' individuals, (2) 250 individuals randomly selected from the 'disabled' individuals and used to build G_2 representing 'non-disabled' individuals. The second subset consists of 100 'disabled' individuals, and 100 'non-disabled' individuals used for the weight optimization phase of our model. Note that the first and second subsets are disjoint.

The test set is a subset consisting of individuals whose disability status is also known; it is not utilized to compute feature weights, but for assessing the accuracy of our model. This set will be described in Section 4.

In order to solve the Training Weight Feature Optimization Problem, we implemented an optimization technique. This process starts with uniform initial weights for each feature. Then we adopt a greedy algorithm to incrementally adjust these weights, one at a time. This process is mathematically formulated as follows:

Initial Setup: The optimization starts with uniform initial weights for each feature: $W_F(f) = 1$ for all $f \in V_F$, where recall that V_F is the set of all feature nodes.

Greedy Algorithm for Weight Adjustment: We adopt a greedy algorithm to incrementally adjust these weights, where each iteration focuses on changing the value of a single feature weight. The adjustment process is mathematically formulated as follows:

$$W_F(f) \leftarrow W_F(f) + \Delta w_f,$$

where Δw_f is the change in weight for feature f. After applying this change, we evaluate its effectiveness on the model's classification accuracy.

Accuracy Assessment: The impact of each weight adjustment is assessed by recalculating the classification accuracy. Each individual i_{class} is classified based on the highest similarity score for the networks G_1 and G_2.

Each individual i_{class} in the dataset has a 'Ground Truth' label, denoted by $\tau(i_{\text{class}})$, which indicates whether the individual is 'disabled' or 'non-disabled'. After all individuals have been classified, we evaluate the accuracy of the model by determining the fraction of individuals that have been correctly classified according to the 'Ground Truth'. The accuracy of the

Table 1: Features in the HRS Dataset That Have Been Used in This Study.

Variables
Years of education
Ever had cancer
Body Mass Index (BMI)
Ever drinks alcohol
Ever had high blood pressure
Total of all assets
Ever had lung disease
Ever had cancer
Ever had arthritis
Any difficulty-Using the toilet
Any difficulty-Walk across room
Any difficulty-Dressing
Any difficulty-Bathing or showering
Any difficulty-Eating
Any difficulty-Get in/out of bed

Table 2: Experimental Results Summary.

Experiment	TP	TN	FP	FN	FPR	FNR	Accuracy (%)
1	25	31	18	21	0.37	0.46	58.95
2	27	34	16	22	0.32	0.50	61.62
3	25	30	19	23	0.39	0.48	56.70
4	30	35	14	20	0.29	0.40	65.66
5	29	34	16	20	0.32	0.41	63.64
6	27	31	19	21	0.38	0.44	59.18
7	28	38	12	21	0.24	0.43	66.67
8	30	38	12	19	0.24	0.39	68.69
9	25	31	19	25	0.38	0.50	56.00
10	28	37	11	20	0.23	0.42	67.71
Average	27	34	16	21	0.32	0.44	62.48

classification model is calculated as follows:

$$\text{Accuracy} = \frac{1}{N}\sum_{j=1}^{N}\left(Classification(i_{\text{class}_j}) = \tau(i_{\text{class}_j})\right),$$

where N is the number of classified nodes, i_{class_j} is the node being classified, G_1 and G_2 are the two networks. Classification(i_{class_j}), and $\tau(i_{\text{class}_j})$ are the predicted classification outcome and the 'Ground Truth' label for the j-th individual, respectively.

At each iteration, we increased the weight of a single feature, evaluating the impact of this change on the model's accuracy. Then the following steps are applied:

- If the accuracy of the model increases following the weight change, we update the selected weight with the change made. Then we randomly select a weight feature to further explore potential improvements in model performance.
- If there is no improvement in accuracy the weight is reverted to its previous value, and the adjustment process randomly select a feature different from the one that has been considered in this iteration.

This process is repeated for all features, until the method converges, that is each weight feature change does not improve accuracy.

4 EXPERIMENTAL RESULTS

In this section, we present the outcomes of a series of preliminary experiments to evaluate the performance of our predictive model. For each experiment, a distinct random sample of 100 individuals was selected from a larger dataset, which included 50 disabled individuals and 50 non-disabled individuals. It is important to note that some selected individuals (at most 5%) may have incomplete information available in the dataset, hence they are not used for the validation phase.

We use RAND U.S. Health and Retirement Study (HRS) data [4]. The used dataset comprises health status and risk factor details from 42,406 survey participants born between the years 1890 and 1995. The features in the HRS dataset that were used in this research are described in Table 1.

Table 2 presents the performance of our methods for the data random samples from the test dataset. The performance metrics considered include True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN) [5] , the False Positive Rate (FPR), the False Negative Rate (FNR), and the overall accuracy in percentage. The FPR and FNR provide insights into the model's tendency to categorize negative and positive cases erroneously, which are respectively calculated as: $FPR = FP/(FP+TN)$, and $FNR = FN/(TP+FN)$. Finally, accuracy quantifies the percentage of actual findings in the dataset that match the ground truth.

In Table 2, experiment 10, which has the highest accuracy at 68.69%, shows a balance between identifying true positives and true negatives while minimizing both false positives and false negatives. In contrast, Experiment 5 shows the lowest accuracy, indicating a higher misclassification rate. On average, these experiments have the accuracy 62.48%, and across the 10 experiments, the model achieved a TP rate of 27, a TN rate of 34, with FP and FN averaging at 16 and 21, respectively. The average FPR was observed at 0.32, with the FNR at 0.44.

In Table 2, a notable pattern across all experiments is the higher number of TN compared to TP, and FN compare to FP. This trend shows that the model has a tendency to classify individuals as 'not-disabled'. In particular, the methods has better performances in correctly identifying individuals who are not disabled than it is at identifying those who are disabled.

5 CONCLUSION

This preliminary study explores feature weight optimization for disability classification and shows how learning and network approaches can be integrated into healthcare frameworks in a potentially fruitful way. We plain to compare the results of our method with other prediction methods. Another possible future direction is to improve the ability to classify 'disabled' individuals. Extending our dataset to include a wider variety of demographic and geographic characteristics is expected to enhance the generalizability and relevance of our findings.

[4] https://hrs.isr.umich.edu.

[5] The TP refers to when an individual's ground truth is 'not disabled', but they are incorrectly classified as 'disabled', and the FN refers to when an individual's ground truth is 'disabled', but they are incorrectly classified as 'not disabled'.

REFERENCES

Bondy, J. A. and Murty, U. S. R. (2008). *Graph theory*. Springer Publishing Company, Incorporated.

Cui, H., Lu, J., Wang, S., Xu, R., Ma, W., Yu, S., Yu, Y., Kan, X., Ling, C., Ho, J., et al. (2023). A survey on knowledge graphs for healthcare: Resources, applications, and promises. *arXiv preprint arXiv:2306.04802*.

Health and Study, R. (2008). Public use dataset. produced and distributed by the university of michigan with funding from the national institute on aging (grant number nia u01ag009740).

Hosseinzadeh, M. M. (2020). Dense subgraphs in biological networks. In *International conference on current trends in theory and practice of informatics*, pages 711–719. Springer.

Hosseinzadeh, M. M., Cannataro, M., Guzzi, P. H., and Dondi, R. (2022). Temporal networks in biology and medicine: a survey on models, algorithms, and tools. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 12(1):10.

Li, Z., Shao, A. W., and Sherris, M. (2017). The impact of systematic trend and uncertainty on mortality and disability in a multistate latent factor model for transition rates. *North American Actuarial Journal*, 21(4):594–610.

Pham, T., Tao, X., Zhanag, J., Yong, J., Zhang, W., and Cai, Y. (2018). Mining heterogeneous information graph for health status classification. In *2018 5th International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC)*, pages 73–78. IEEE.

Pham, T., Tao, X., Zhang, J., Yong, J., Li, Y., and Xie, H. (2022). Graph-based multi-label disease prediction model learning from medical data and domain knowledge. *Knowledge-based systems*, 235:107662.

Rossetti, G. and Cazabet, R. (2018). Community discovery in dynamic networks: a survey. *ACM computing surveys (CSUR)*, 51(2):1–37.

Stuck, A. E., Walthert, J. M., Nikolaus, T., Büla, C. J., Hohmann, C., and Beck, J. C. (1999). Risk factors for functional status decline in community-living elderly people: a systematic literature review. *Social science & medicine*, 48(4):445–469.

Tao, X., Pham, T., Zhang, J., Yong, J., Goh, W. P., Zhang, W., and Cai, Y. (2020). Mining health knowledge graph for health risk prediction. *World Wide Web*, 23:2341–2362.

Wang, T., Qiu, R. G., Yu, M., and Zhang, R. (2020). Directed disease networks to facilitate multiple-disease risk assessment modeling. *Decision Support Systems*, 129:113171.

Enhancing Dyeing Processes with Machine Learning: Strategies for Reducing Textile Non-Conformities

Mariana Carvalho[1] [a], Ana Borges[1] [b], Alexandra Gavina[2] [c], Lídia Duarte[1], Joana Leite[3,4] [d],
Maria João Polidoro[5,6] [e], Sandra Aleixo[6,7] [f] and Sónia Dias[8,9] [g]

[1] *CIICESI, ESTG, Polytechnic of Porto, Rua do Curral, Casa do Curral, Margaride, Felgueiras, 4610-156, Portugal*
[2] *Lema-ISEP, Polytechnic of Porto, Rua Dr. António Bernardino de Almeida, 431, Porto, 4249-015, Portugal*
[3] *Polytechnic University of Coimbra, Rua da Misericórdia, Lagar dos Cortiços,*
S. Martinho do Bispo, 3045-093 Coimbra, Portugal
[4] *CEOS.PP Coimbra, Polytechnic University of Coimbra, Bencanta, 3045-601 Coimbra, Portugal*
[3] *ESTG, Polytechnic of Porto, Rua do Curral, Casa do Curral, Margaride, Felgueiras, 4610-156, Portugal*
[6] *CEAUL – Centro de Estatística e Aplicações da Universidade de Lisboa, Portugal*
[7] *Department of Mathematics, ISEL – Instituto Superior de Engenharia de Lisboa, Portugal*
[8] *ESTG, Instituto Politécnico de Viana do Castelo, Portugal*
[9] *LIAAD-INESC TEC, Portugal*
mrc@estg.ipp.pt, aib@estg.ipp.pt, alg@isep.ipp.pt, 8140330@estg.ipp.pt, jleite@iscac.pt, mjp@estg.ipp.pt,
sandra.aleixo@isel.pt, sdias@estg.ipvc.pt

Keywords: Textile Dyeing, Non-Conformity, Data Mining, Knowledge Discovery, Prediction, Random Forest, Gradient Boosted Trees.

Abstract: The textile industry, a vital sector in global production, relies heavily on dyeing processes to meet stringent quality and consistency standards. This study addresses the challenge of identifying and mitigating non-conformities in dyeing patterns, such as stains, fading and coloration issues, through advanced data analysis and machine learning techniques. The authors applied Random Forest and Gradient Boosted Trees algorithms to a dataset provided by a Portuguese textile company, identifying key factors influencing dyeing non-conformities. Our models highlight critical features impacting non-conformities, offering predictive capabilities that allow for preemptive adjustments to the dyeing process. The results demonstrate significant potential for reducing non-conformities, improving efficiency and enhancing overall product quality.

1 INTRODUCTION

Nowadays, there has been a notable evolution in the textile sector due to technological progress and a growing focus on quality and sustainability. Among the key areas constantly scrutinized is the dyeing process, which plays a vital role in achieving the desired appearance and meeting strict product requirements. Yet, this procedure is fulled with challenges, including non-conformities such as stains, fading and color mismatches. These challenges not only influence the aesthetic of textile items but also affect customer approval and the ecological impact of manufacturing methods.

Considering this, a Portuguese company in the textile dye sector has proposed a significant challenge. The company's goal is to uncover patters that may lead to non-conformities in the dyeing process. The challenge requires examining numerous variables that may impact the results of dyeing, such as the fabric type, the chemical makeup of dyes and the details of the dyeing equipment. Understanding the complex interplay between these factors is crucial for identifying the root causes of non-conformities, which can vary widely and be influenced by subtle changes in the production process.

[a] https://orcid.org/0000-0003-2190-4319
[b] https://orcid.org/0000-0003-4244-5393
[c] https://orcid.org/0000-0002-4694-933X
[d] https://orcid.org/0000-0001-6828-9486
[e] https://orcid.org/0000-0002-2220-4077
[f] https://orcid.org/0000-0003-1740-8371
[g] https://orcid.org/0000-0002-2100-2844

Carvalho, M., Borges, A., Gavina, A., Duarte, L., Leite, J., Polidoro, M., Aleixo, S. and Dias, S.
Enhancing Dyeing Processes with Machine Learning: Strategies for Reducing Textile Non-Conformities.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 363-370
ISBN: 978-989-758-716-0; ISSN: 2184-3228

Therefore, the authors suggest performing an detailed data analysis and applying machine learning algorithms to prediction of key factors that may lead to non-conformities, such as Random Forest (RF) and Gradient Boosted Trees (GBT) algorithms. Machine learning algorithms allow for the examination of large quantities of data in order to uncover patterns and relationships that may not be readily apparent using conventional analysis techniques. Also, these machine learning models are used to recognize main factors that affect dyeing non-conformities and, since both models have the ability to predict outcomes, it is possible to suggest proactive modifications in the dyeing process, showing considerable potential in decreasing flaws, enhancing productivity and improving the overall quality of the product.

This paper is organized as follows: the background section explores the integration of advanced data analysis techniques and machine learning algorithms in textile dyeing processes, emphasizing the identification of key factors influencing dyeing non-conformities and offering strategies to enhance product quality. Next, the authors present a descriptive analysis of the dataset on non-conformities, highlighting its key features. This is followed by a detailed exploratory data analysis section, organized into several subsections: Analysis of Non-Conformities, Causes of Non-Conformities, Fabrics with Non-Conformities, Colourants in Non-Conformities and Colouring Machines that Lead to Non-Conformities. Subsequently, the paper provides a detailed explanation of the entire process of predicting significant factors that may be resposible for non-conformities using machine learning. Finally, the paper concludes with a discussion and comparison of the results, followed by the Conclusions and Future Work section.

2 BACKGROUND

As stated before, the textile industry has recently advanced due to technological progress and a focus on sustainability and quality control, particularly in dyeing processes. The main challenges include minimizing environmental impacts and addressing non-conformities. Studies like (Zhang et al., 2018) have proposed improved designs for textile production processes based on life cycle assessment, targeting the reduction of environmental impacts by identifying best available technologies and focusing on critical stages like printing and dyeing to improve product quality and reduce resource depletion and ecological influence. (Parisi et al., 2015) emphasize the need for more sustainable production processes, demonstrating the feasibility of alternative dyeing methods that reduce energy, water and raw materials consumption, thereby aligning with consumer demand for eco-friendly products.

In response to these challenges, the integration of advanced data analysis techniques and machine learning into the textile dyeing process represents a significant shift towards more data-driven decision-making. Research by (Park et al., 2020) has developed a cyber-physical energy system that utilizes manufacturing big data and machine learning techniques to improve energy efficiency in dyeing processes without the need for expensive equipment, thereby enhancing process and system efficiency. Furthermore, efforts to incorporate green solvents, as discussed by (Meksi and Moussa, 2017) and to explore the ecological application of ionic liquids in textile processes, offer innovative pathways for reducing the environmental footprint and improving the sustainability of the dyeing process. These developments not only aim to address immediate quality control challenges but also signify a broader movement towards incorporating advanced technologies in traditional textile dyeing industries, setting a new benchmark for sustainability and efficiency.

3 DATASET ON NON-CONFORMITIES

In this section, the authors present the dataset used for the analysis, detailing the preprocessing steps and the comprehensive descriptive statistics of the variables involved.

All preprocessing tasks were conducted using RapidMiner[1] and Python[2]. Missing data were imputed using the K-Nearest Neighbour (Fix, 1985) method to ensure the integrity and completeness of the dataset. This preprocessing step is crucial for accurate and reliable machine learning model training.

In our analysis, the original dataset comprises a total of 5,546 records across 23 distinct variables. But, in order to maintain the confidentiality and anonymity of the textile company, the authors only consider the following set of variables in the subsequent analysis: Fabric, Colourant, Date, Defect (which corresponds to Non-Conformity), Cause and Colouring Machine. The descriptive statistics of all variables are as summarized in Table 1. For categorical variables, the table provides name of variable and unique values. For numerical variables, it includes the name of the variable,

[1] https://altair.com/altair-rapidminer

[2] https://www.python.org/

Table 1: Descriptive Statistics of the Dataset.

Variable name	Unique	Mean	STD	Min	Max
Fabric	13	-	-	-	-
Colourant	-	3.35	2.95	0.00	17.00
Date	754	-	-	-	-
Defect	6	-	-	-	-
Cause	9	-	-	-	-
Colouring Machine	39	-	-	-	-

mean, standard deviation (STD), minimum (Min) and maximum (Max) values.

4 EXPLORATORY ANALYSIS OF THE DATASET ON NON-CONFORMITIES

In this section an exploratory analysis of the content of the database is presented. The authors explore the textile manufacturing non-conformities from January 2020 to July 2023 and show the patterns and trends that emerge from the data, seeking to understand the underlying causes and their temporal dynamics.

4.1 Analysis of Non-Conformities

First, it is important to analyse the evolution of non-conformities occurrences over the years. Figure 1 shows this evolution over the period from 2020 to 2023. The non-conformities considered in this study are 'Stained', 'Oil', 'Other', 'Failed', 'Undyed' and 'Creases'. Overall, the total number of non-conformities (represented by the dark blue line) decreased each year, reflecting an overall improvement in quality control measures. 'Failed' non-conformities (represented by the yellow line) consistently has the highest number of non-conformities. The 'Oil' (represented by the orange line) exhibits variability, with a slight peak in 2021, followed by a consistent decline in 2023. The 'Other' non-conformities occurrences (represented by the grey line), which includes miscellaneous non-conformities, peaked in 2020 and showed a gradual decrease by 2023. 'Stained' non-conformities (represented by the blue line) shows a decreasing trend over the years, starting in 2020 and declining in 2023. 'Undyed' (represented by the light blue line) shows fluctuations, with the highest number in 2022. Despite these fluctuations, the trend appears relatively stable with a slight increase. Lastly, 'Creases' (represented by the green line) shows a slight decrease over the years.

The distribution of non-conformities is detailed as follows: The 'Failed' non-conformity has the highest count, with 2142 occurrences, representing 39% of the total non-conformities. The 'Other' and 'Undyed' categories follow, each constituting 16% of the total non-conformities, with counts of 903 and 915 respectively. 'Stained' non-conformities account for 12% of the total, with 673 occurrences, while 'Creases' represent 9% with 486 occurrences. The 'Oil' non-conformity, although the least frequent, still comprises 8% of the total non-conformities, with 427 occurrences.

4.2 Causes of Non-Conformities

The next step is to analyse the causes that influence non-conformities. The distribution of causes of non-conformities are described as followed: the most significant issue is 'Poorly analysed,' with 1880 occurrences, corresponding a total of 34%. 'Other' reasons have also led to a considerable number of occurrences, totalling 1017 (18%). The 'Poorly executed/monitored process' accounts for 786 occurrences (14%). 'Process phases in different conditions' have contributed to 567 (10%) non-conformities. 'Insufficient disposal by normal process' is the next most frequent concern with 430 occurrences (8%). 'Rope jammed/rebent/running poorly', 'Dyed (folded) accessory together with mesh', 'Lack of machine/cart cleaning' and 'Process interrupted for review' have occurrences over 200 (each one with 4% of total occurrences).

4.3 Fabrics with Non-Conformities

Following this, the analysis of fabrics with non-conformities is also important. The description of fabrics with non-conformities' distribution is as follows. The predominant fabric with non-conformities is Jersey, with 1825 of total occurrences, comprising 33% of the total occurrences. Followed by Rib (with a total of 1337 occurrences) at 24% of the non-conformities. Felpa fabrics represent 14% of the non-conformities and with a total of 754 occurrences, while Golve fabrics contribute 6%. Both Piquet and Screen fabrics account for 7% each. Other fabric types, such as

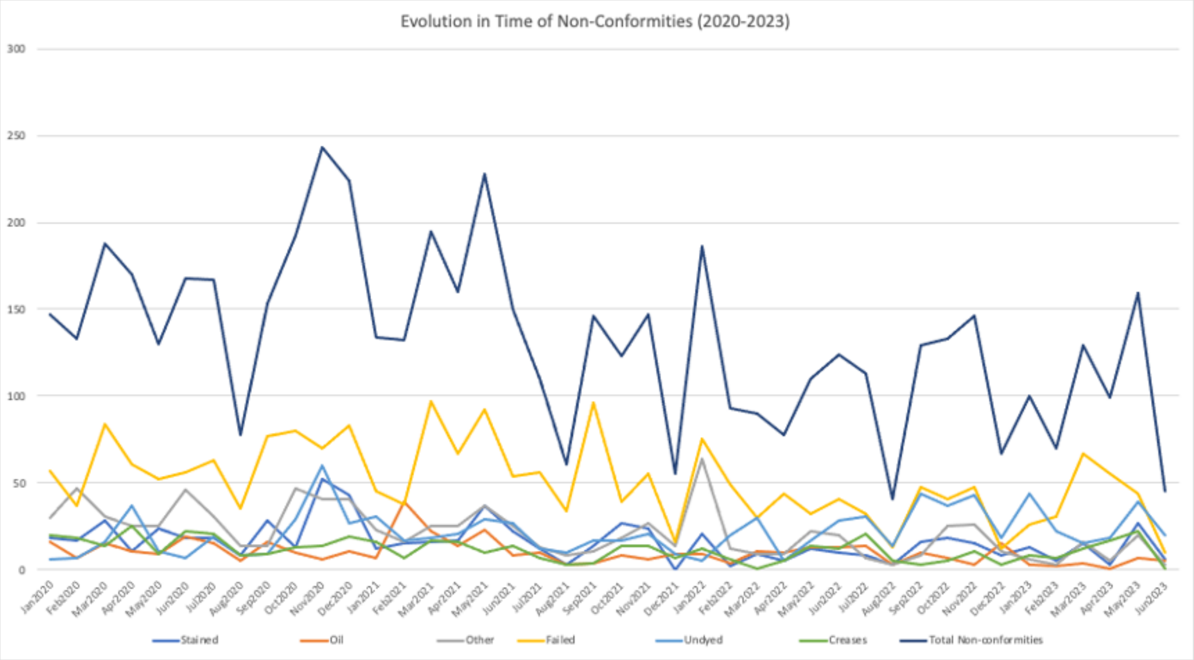


Figure 1: Evolution of Non-Conformities over the years.

Nastro and Interlock, each represent 3% of the non-conformities. Minor categories include Screen, Nets and Cord, each constituting 1% and Turca and Strips have a negligible 0% presence of non-conformities.

4.4 Colourants Presented in Non-Conformities

Next, the authors analyse the distribution of colourants presented in non-conformities. The 'Reactive' colourant has an overwhelmingly high count of non-conformities, totaling 4405, which constitutes 76.46% of the total non-conformities. The colourant 'Reactive/Disperse' also shows a substantial number of non-conformities, with a count of 535, accounting for 9.29% of the total. While significantly lower than 'Reactive', this combination of colourants still represents a considerable source of non-conformities. With 271 non-conformities, 'Colourless' dyes represent 4.70% of the total. 'White' dyes account for 96 non-conformities, with 1.67% of the total occurrences. The colourant 'Acid' has 87 non-conformities, with 1.51% of the total. The combination of 'Reactive/Acid' dyes results in 79 non-conformities, which is 1.37% of the total. Disperse' dyes show a relatively low count of 23 non-conformities, representing 0.40% of the total. The 'Direct' and 'Indefinite' colourants have the lowest counts, with 30 (0.52%) and 10 (0.17%) occurrences respectively. Similarly, 'Cationic/Reactive' colourants also have a low count of 10 non-conformities, which is 0.17% of the total occurrences.

4.5 Colouring Machines

Another important variable that may impact the non-conformities occurrences is the variable colouring machines. This dataset present a total of 39 colouring machines and overall, there's a fluctuation in the percentage of non-conformities for each machine across the four years. Some machines show a reduction in non-conformities over time, while others exhibit an increase or inconsistent patterns. The top 5 colouring machines leading to the most occurrences of non-conformities are: 'TNJT13' with a total of 419 (7.55%), 'TNJT05' with 346 occurrences (6.24%), 'TNJT19' with a total of 304 (5.48%) 'TNJT11' with a sum of 300 occurrences (5.41%) and finally, 'TNJT32' with 287 (5.17% of total occurrences).

This extensive analysis of data helps improve comprehension of the data, leading to better feature engineering and model development in future analysis.

5 PREDICTION OF DYEING NON-CONFORMITIES FACTORS

This section explores the use of machine learning algorithms, specifically RF and GBT, to predict key factors that may contribute to non-conformities and extract feature importance, identifying the most significant factors contributing to these issues. Understanding these key features enables targeted interventions and process optimizations, enhancing product quality and reducing defect rates.

RF combines multiple decision trees to enhance predictive accuracy and control overfitting, making it suitable for datasets with numerous features and non-linear relationships (Robnik-Sikonja, 2004). This algorithm has been effectively utilized in various industrial contexts, such as predictive maintenance, where it anticipates equipment failures by analyzing sensor data and operational logs, thus minimizing downtime and improving productivity (Kusiak and Verma, 2011). Additionally, RF provides insights into feature importance, crucial for understanding key factors influencing non-conformities in dyeing processes (Breiman, 2001).

Conversely, the GBT algorithm builds trees sequentially, with each new tree correcting errors made by the previous ones, thereby significantly enhancing prediction accuracy (Friedman, 2001). GBTs have demonstrated superior performance in industrial applications and in manufacturing. GBTs optimize production processes by identifying critical factors influencing product quality, enabling precise control and reduction of non-conformities (He and Wu, 2018).

5.1 Findings of the Random Forest Model

The Figure 2 shows a representative tree model obtained using the RF algorithm (Ho, 1995). The model configuration chosen was: the number of trees in the forest equals to 100 and the minimum number of samples required in leaf node equals to 50 and the data was divided into 80% for training and 20% for testing. The returned RF model represented in the Figure shows the factors that lead to non-conformities, which was used as classe label. According to the root node, the most important factor is 'Poorly analysed' processes mainly resulting in 'Failed' classifications. After a thorough analysis, the next significant factor is the 'Dyed (folded) accessory along with mesh' process, frequently leading to 'Undyed' non-conformities. Next, there are machine-specific factors, especially those involving the colouring machine 'TNJT25' and 'TNJT23', as well as fabric-related issues like problems with 'Golves' fabric, play a significant role in influencing non-conformities. In the Figure, it also possible to see that 'Colourless' colourants and 'Piquet' fabrics play a major role in 'Other' non-conformities. In addition, issues related to the dyeing process such as 'Process phases in different conditions' and 'Insufficient dispossal by normal process' are important elements.

The obtained RF estimator's performance measures shows the model's accuracy in predicting different types of non-conformities. The precision for predicting the non-conformity 'Crease' is 0.53, which means that 53% of the predicted creases were correct. The recall is 0.23, suggesting that only 23% of the actual creases were identified. The f1-score is 0.33, reflecting the balance between precision and recall. The 'Failed' non-conformity has perfect precision (1.00) and high recall (0.87), resulting in a high f1-score (0.93), which means that 100% of 'Failed' predictions are accurate. The precision and recall in the 'Oil' non-conformity are both very high (0.97 and 0.99, respectively) and a f1-score of 0.98. The 'Stained' non-conformity has a precision of 0.60 and a recall of 0.88. The precision is 0.92 and the recall is 0.81 on the 'Undyed' non-conformity, resulting in an f1-score of 0.86. In the 'Other' non-conformity the precision is 0.54 and the recall is 0.73, resulting in an f1-score of 0.62. The overall accuracy of the model is 0.79, so it shows that almost 80% of the predictions are accurate. This is a strong performance, indicating that the model is successful in identifying various types of non-conformities.

With the analysis of important features from the RF model one can know which are the features that impact mostly the non-conformities appearances. The cause 'Poorly analysed' remains the most influential feature of non-conformities, with an importance value of 0.385060. The second most influential feature is the cause 'Process phases in different conditions' which presents an importance of 0.141318. The cause 'Insufficient disposal by normal process', rated at 0.135305 in terms of importance, is the third most influential feature. The cause 'Poorly executed/monitored process', with a significance rating of 0.117062, is also a major factor in non-conformities.

Additional important factors are the 'Other' causes (0.052049), the cause 'Lack of machine/cart cleaning' (0.049191) and the cause 'Process interrupted for review' (0.042287). While not as influential enough as the other main causes, these factors still greatly affect non-conformities. Other process problems like the cause 'Dyed (folded) accessory together with mesh' (0.019247) and the cause 'Rope jammed/rebent/running poorly' (0.013083) also play an important role in leading to non-conformities. The colourant 'Colourless' (0.008710), the fabric 'Jersey' (0.007713), the colourant 'Reactive' (0.003967) and the fabric 'Rib' (0.003472) shows that not only the causes and processes influence the non-conformities. While not as significant as cause and process-related factors, these features still contribute to influence non-conformities. Also, Machine-specific features, like the colouring machine 'TNJT05' (0.003181), show that particular machines impact non-conformity rates as well. The fabric 'Piquet' (0.003146) is also considered in the top 15 of the more influential features, suggesting that along with the fabric Jersey and Rib can also lead to non-conformities.

5.2 Findings of the Gradient Boosted Trees Model

Next, the authors apply the GBT algorithm. The chosen model configuration was learning rate ('classifier_learning_rate') are 0.2, maximum depth ('classifier_max_depth') equals to 5, the number of trees ('classifier_n_estimators') equals to 100 and also, the data was split with 80% allocated for training and 20% for testing.

The returned performance metrics in this model are very similar to the ones obtained previously using the RF model. In the 'Creases' non-conformity, the model obtained a precision score of 0.48, which means that 48% of the predicted creases were correct; and a recall score of 0.38, suggesting that only 38% of the actual creases were identified. Within the 'Failed' prediction, the model showed strong results with a precision of 0.96 and a recall of 0.89. The 'Oil' group also showed good outcomes, achieving a precision of 0.97 and a recall of 0.96. On predicting 'Other' non-conformities, the model achieved a precision of 0.59 and a recall of 0.71. Similarly, the 'Stained' non-conformity showed a precision of 0.66 and a recall of 0.79. In the 'Undyed' non-conformity classification, the model reached a precision of 0.90 and a recall of 0.84. Overall, the GBT model achieved an Accuracy of 0.80.

The top 15 factors identified by the GBT

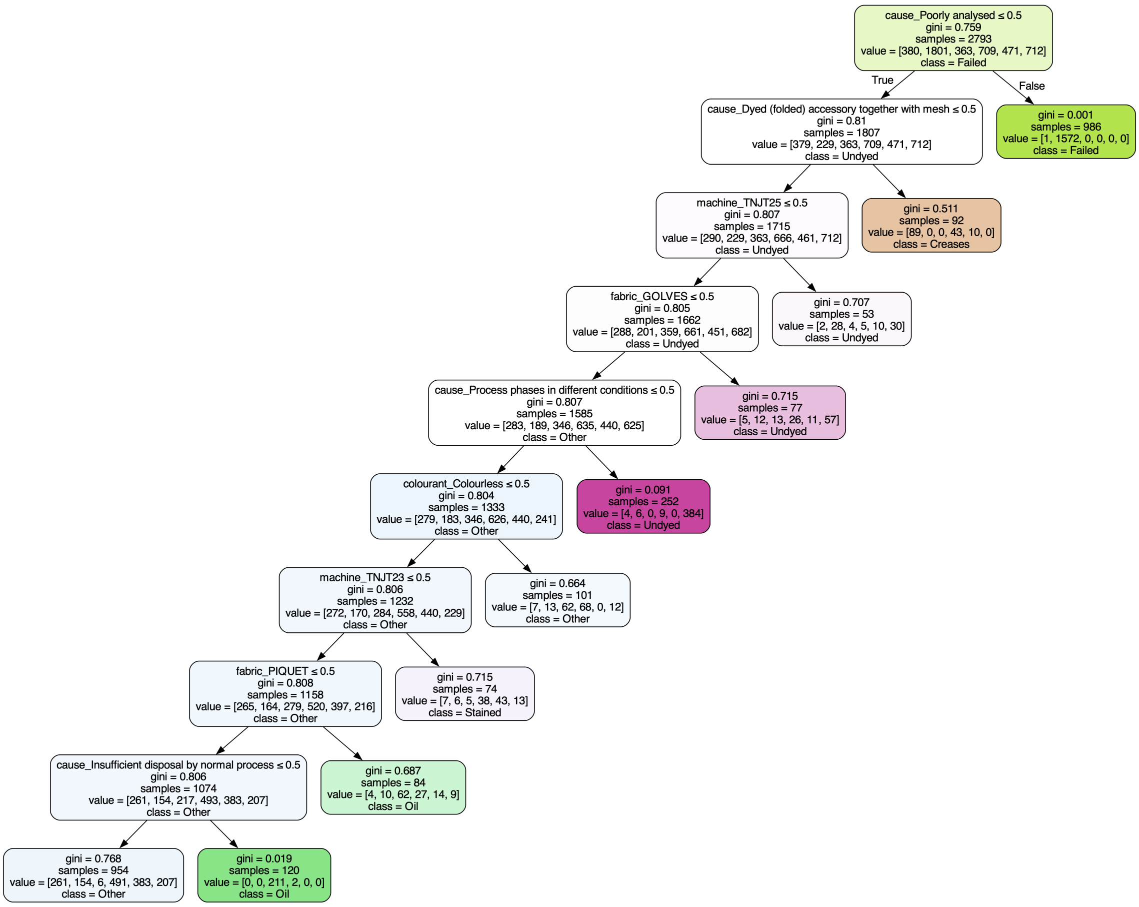

cause_Poorly analysed ≤ 0.5
gini = 0.759
samples = 2793
value = [380, 1801, 363, 709, 471, 712]
class = Failed
True
False
cause_Dyed (folded) accessory together with mesh ≤ 0.5
gini = 0.81
samples = 1807
value = [379, 229, 363, 709, 471, 712]
class = Undyed
gini = 0.001
samples = 986
value = [1, 1572, 0, 0, 0, 0]
class = Failed
machine_TNJT25 ≤ 0.5
gini = 0.807
samples = 1715
value = [290, 229, 363, 666, 461, 712]
class = Undyed
gini = 0.511
samples = 92
value = [89, 0, 0, 43, 10, 0]
class = Creases
fabric_GOLVES ≤ 0.5
gini = 0.805
samples = 1662
value = [288, 201, 359, 661, 451, 682]
class = Undyed
gini = 0.707
samples = 53
value = [2, 28, 4, 5, 10, 30]
class = Undyed
cause_Process phases in different conditions ≤ 0.5
gini = 0.807
samples = 1585
value = [283, 189, 346, 635, 440, 625]
class = Other
gini = 0.715
samples = 77
value = [5, 12, 13, 26, 11, 57]
class = Undyed
colourant_Colourless ≤ 0.5
gini = 0.804
samples = 1333
value = [279, 183, 346, 626, 440, 241]
class = Other
gini = 0.091
samples = 252
value = [4, 6, 0, 9, 0, 384]
class = Undyed
machine_TNJT23 ≤ 0.5
gini = 0.806
samples = 1232
value = [272, 170, 284, 558, 440, 229]
class = Other
gini = 0.664
samples = 101
value = [7, 13, 62, 68, 0, 12]
class = Other
fabric_PIQUET ≤ 0.5
gini = 0.808
samples = 1158
value = [265, 164, 279, 520, 397, 216]
class = Other
gini = 0.715
samples = 74
value = [7, 6, 5, 38, 43, 13]
class = Stained
cause_Insufficient disposal by normal process ≤ 0.5
gini = 0.806
samples = 1074
value = [261, 154, 217, 493, 383, 207]
class = Other
gini = 0.687
samples = 84
value = [4, 10, 62, 27, 14, 9]
class = Oil
gini = 0.768
samples = 954
value = [261, 154, 6, 491, 383, 207]
class = Other
gini = 0.019
samples = 120
value = [0, 0, 211, 2, 0, 0]
class = Oil

Figure 2: A representative Tree obtained from the Random Forest model.

model that most influence the occurrence of non-conformities are as follows: The ‘Cause_Poorly analysed’ holds the top score of 0.318236, highlighting its major influence on the model’s forecasts. Following this are the phrases ‘Cause_Process phases under various circumstances’ with a significance rating of 0.129308 and ‘Cause_Inadequate disposal through regular procedures’ with a rating of 0.121097. Some other significant characteristics are ‘Reason for lack of cleaning of machine/cart’ (0.083689), ‘Reason for process interruption for review’ (0.052792) and ‘Reason for poorly executed/monitored process’ (0.036939). The factor ‘Cause_Other’ also has a significance level of 0.028393. Further factors like ‘Cause_Dyed accessory folded with mesh’ (0.025745) and ‘Cause_Rope jammed/rebent/running poorly’ (0.018613) also play a role in the model’s predictions. Some characteristics related to particular devices and dyes are also present in the top 15. The listed items are ‘Colouring_Machine_TNJT06’ (0.005783), ‘Colouring_Machine_TNJT32’ (0.005715), ‘Colourant_Colourless’ (0.005656), ‘Fabric_Screen’ (0.005642), ‘Colourant_Acid’ (0.005449) and ‘Colouring_Machine_TNJT25’ (0.005265).

6 DISCUSSION

The application of machine learning algorithms, specifically RF and GBT, to the textile dyeing process has yielded significant insights into the factors influencing non-conformities. Our analysis identified several key variables that impact the occurrence of non-conformities, such as poorly analysed processes, variations in process phases and insufficient disposal methods. These findings are summarized in Table 2.

The RF model’s high importance score for ‘Poorly analysed process’ underscores the necessity for thorough inspections and quality checks at each stage of the dyeing process. This feature’s dominance suggests that many non-conformities could be mitigated by improving the rigor of process analysis. Similarly, the GBT model aligns closely with this finding, reinforcing the critical role of detailed process scrutiny. ‘Process phases in different conditions’ emerged as another significant factor. Variations in these conditions can lead to inconsistencies in dye application, resulting in non-conformities. Both models consistently rated this feature highly, suggesting that addressing these variations could significantly reduce non-conformities.’Insufficient disposal by normal process’ also featured prominently in both models, indicating that the methods used to remove excess materials or byproducts during dyeing can influence the final product’s quality. Optimizing disposal processes to ensure complete removal of unwanted substances could enhance overall dyeing consistency. The ‘Poorly executed/monitored process’ factor, while rated lower in the GBT model, still showed considerable importance in the RF model. This points to the need for continuous monitoring and quality assurance practices during dyeing to prevent errors and ensure uniform quality.

Comparing our findings with existing literature, such as Zhang et al. (2018) and Parisi et al. (2015), reveals a consistent emphasis on the importance of process control and quality management in reducing non-conformities by improving sustainability in textile manufacturing. Our study extends these ideas by providing a data-driven approach to identifying and addressing specific factors leading to non-conformities.

7 CONCLUSIONS

This study demonstrates the potential of machine learning techniques in optimizing the textile dyeing process by identifying and mitigating factors leading to non-conformities. Machine learning models such as RF and GBT provide a detailed analysis of critical features impacting dyeing quality, which is relevant for industry practitioners to enhance process control and quality assurance practices.

The high importance scores for process analysis and conditions suggest that many non-conformities can be mitigated through more rigorous quality checks and maintaining consistent dyeing environments. The alignment of our results with existing literature further validates the significance of robust process control and quality management in the textile industry. Integrating these machine learning results into the dyeing process can lead to substantial improvements in efficiency, waste reduction and overall product quality. This approach not only addresses immediate quality control challenges but also sets a new standard for incorporating advanced technologies in traditional manufacturing processes.

Future work for this study includes expanding the dataset to cover a wider variety of textiles and dyeing methods to improve the accuracy of the predictive models. Incorporating more machine learning algorithms, such as deep learning methods, may yield more precise and reliable predictions. Moreover, utilizing real-time data analysis and anomaly detection systems may facilitate prompt corrective measures during dyeing, thereby enhancing efficacy and minimizing non-conformities occurrences.

Table 2: Top features influencing non-conformities in the dyeing process according to Random Forest and Gradient Boosted Trees models.

Feature	Description
Poorly analysed process	Indicates process steps not thoroughly checked, leading to non-conformities.
Process phases in different conditions	Variations in process phases affecting dyeing quality.
Insufficient disposal by normal process	Inadequate removal of materials causing non-conformities.
Poorly executed/monitored process	Indicates issues in the execution and monitoring of dyeing processes.

ACKNOWLEDGEMENTS

This work has been supported by national funds through FCT – Fundação para a Ciência e Tecnologia through project UIDB/04728/2020. The authors thank the textile company for providing the real data used in this study.

The research at CMAT was partially financed by Portuguese Funds through FCT (Fundação para a Ciência e a Tecnologia) within the Projects UIDB/00013/ 2020 and UIDP/00013/2020.

Partially supported by the Centre for Mathematics of the University of Coimbra (funded by the Portuguese Government through FCT/MCTES, DOI 10.54499/UIDB/00324/2020).

Research partially funded by FCT - Fundação para a Ciência e a Tecnologia, Portugal, under the project UIDB/00006/2020, DOI: 10.54499/UIDB/00006/2020 (CEAUL) and ISEL.

REFERENCES

Breiman, L. (2001). Random forests. *Machine Learning*, 45:5–32.

Fix, E. (1985). *Discriminatory analysis: nonparametric discrimination, consistency properties*, volume 1. USAF school of Aviation Medicine.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

He, Y., H. Z. and Wu, Q. (2018). Production line control system for the wulanchabu manufacturing plant. In *Proceedings of the 2018 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, pages 1–4.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

Kusiak, A., Z. Z. and Verma, A. (2011). Predictive maintenance: Thrust bearing fault classification. *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*.

Meksi, N. and Moussa, A. (2017). A review of progress in the ecological application of ionic liquids in textile processes. *Journal of Cleaner Production*, 161:105–126.

Parisi, M., Fatarella, E., Spinelli, D., Pogni, R., and Basosi, R. (2015). Environmental impact assessment of an eco-efficient production for coloured textiles. *Journal of Cleaner Production*, 108:514–524.

Park, K., Kang, Y. T., Yang, S., Zhao, W.-B., Kang, Y.-S., Im, S., Kim, D. H., Choi, S. Y., and Noh, S. (2020). Cyber physical energy system for saving energy of the dyeing process with industrial internet of things and manufacturing big data. *International Journal of Precision Engineering and Manufacturing-Green Technology*, 7:219–238.

Robnik-Sikonja, M. (2004). Improving random forests. In *Proceedings of the 15th European Conference on Machine Learning*, pages 359–370. Springer.

Zhang, Y., Kang, H.-S., Hou, H., Shao, S., Sun, X., Qin, C., and Zhang, S. (2018). Improved design for textile production process based on life cycle assessment. *Clean Technologies and Environmental Policy*, 20:1355–1365.

Comparative Analysis of Real-Time Time Series Representation Across RNNs, Deep Learning Frameworks, and Early Stopping

Ming-Chang Lee[a], Jia-Chun Lin[b] and Sokratis Katsikas[c]

Department of Information Security and Communication Technology, Norwegian University of Science and Technology (NTNU), Gjøvik, Norway

mingchang1109@gmail.com,{jia-chun.lin, sokratis.katsikas}@ntnu.no

Keywords: Comparative Analysis, Performance Evaluation, Real-Time Time Series Representation, Recurrent Neural Networks, Deep Learning Frameworks, Early Stopping.

Abstract: Real-Time time series representation is becoming increasingly crucial in data mining applications, enabling timely clustering and classification of time series without requiring parameter configuration and tuning in advance. Currently, the implementation of real-time time series representation relies on a fixed setting, consisting of a single type of recurrent neural network (RNN) within a specific deep learning framework, along with the adoption of early stopping. It remains unclear how leveraging different types of RNNs available in various deep learning frameworks, combined with the use of early stopping, influences the quality of representation and the efficiency of representation time. Arbitrarily selecting an RNN variant from a deep learning framework and activating the early stopping function for implementing a real-time time series representation approach may negatively impact the performance of the representation. Therefore, in this paper, we aim to investigate the impact of these factors on real-time time series representation. We implemented a state-of-the-art real-time time series representation approach using multiple well-established RNN variants supported by three widely used deep learning frameworks, with and without the adoption of early stopping. We analyzed the performance of each implementation using real-world open-source time series data. The findings from our evaluation provide valuable guidance on selecting the most appropriate RNN variant, deciding whether to adopt early stopping, and choosing a deep learning framework for real-time time series representation.

1 INTRODUCTION

In recent years, the increasing integration of the Internet of Things (IoT) within the cyber-physical world has led to a surge in the demand for time series analysis tasks such as clustering, classification, anomaly detection, forecasting, and indexing (Lee et al., 2020b; Lee et al., 2020a; Lee et al., 2021b; Ratanamahatana et al., 2005; Bagnall et al., 2017; Ismail Fawaz et al., 2019).This surge is primarily driven by the constant measurement and collection of large volumes of time series data from interconnected devices and sensors. However, analyzing raw time series data poses challenges due to its high computational cost and memory requirements (Ding et al., 2008). To address this, high-level representation approaches have emerged as a solution. These approaches aim to extract features from time series data or reduce its dimensionality while retaining its essential characteristics, thereby enabling effective and efficient time series analysis (Aghabozorgi et al., 2015).

Several time series representation approaches have been introduced, including Symbolic Aggregate Approximation (Lin et al., 2007), Piecewise Aggregate Approximation (Keogh et al., 2001), and the clipped representation approach (Bagnall et al., 2006). However, these methods typically operate solely on fixed-length time series rather than continuously updating or streaming time series data. Before generating a representation, these approaches require preprocessing the time series using z-normalization, which is a commonly employed technique in time series normalization (Dau et al., 2019).

However, z-normalization might cause two distinct time series to become indistinguishable (Höppner, 2014), potentially misguiding the representation approaches and negatively impacting subsequent data mining tasks (Lee et al., 2024b). Furthermore, many representation approaches require

[a] https://orcid.org/0000-0003-2484-4366

[b] https://orcid.org/0000-0003-3374-8536

[c] https://orcid.org/0000-0003-2966-9683

Lee, M.-C., Lin, J.-C. and Katsikas, S.
Comparative Analysis of Real-Time Time Series Representation Across RNNs, Deep Learning Frameworks, and Early Stopping.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 371-382
ISBN: 978-989-758-716-0; ISSN: 2184-3228

users to preconfigure and fine-tune parameters such as time series length, sliding window size, or alphabet size (Lin et al., 2007). Inadequate parameter values may result in poor representations, compromising the effectiveness of subsequent data mining operations.

Based on our investigation, only NP-Free, a real-time time series representation method developed by Lee et al. (Lee et al., 2023), meets the criteria for real-time time series representation. Unlike other approaches, NP-Free operates on ongoing time series without the need for z-normalization and does not require parameter tuning. It uniquely converts raw time series into root-mean-square error (RMSE) series in real time, ensuring that the resulting RMSE series represents the original raw series. However, this approach has only been implemented using a single type of recurrent neural network (RNN), specifically Long Short-Term Memory (LSTM) within a specific deep learning (DL) framework, namely Deeplearning4j (Deeplearning4j, 2023), along with the adoption of the early stopping function (EarlyStopping, 2023).

In reality, several other DL frameworks have been introduced and widely used, such as TensorFlow (Abadi et al., 2016) and PyTorch (Paszke et al., 2019). These frameworks aim to simplify the complex data analysis process by providing comprehensive libraries and tools for building, training, and deploying machine learning models (Nguyen et al., 2019). While numerous surveys and analyses have compared different DL frameworks, they have primarily focused on specific tasks (e.g., anomaly detection and natural language processing) or different types of computing environments.

To provide a comprehensive evaluation of how these different factors impact real-time time series representation, we implemented NP-Free using five RNN variants, with and without the early stopping function, across three DL frameworks. We conducted a series of experiments using open-source time series data to evaluate all the implementations. The results demonstrate that the choice of RNN variants, DL frameworks, and the early stopping function significantly influence both representation quality and time efficiency. Therefore, it is crucial to carefully consider the selection of these factors when designing and implementing a real-time time series representation approach. The experimental results show that NP-Free implemented with DL4J, using LSTM and the early stopping function, provides more stable RMSE series than NP-Free implemented with PyTorch or TensorFlow (TFK), regardless of whether the early stopping function in PyTorch or TFK is activated.

The rest of this paper is structured as follows: Section 2 introduces the background related to RNNs, DL frameworks, and NP-Free. Section 3 provides an overview of the related work. In Section 4, we present and detail our evaluation setup and results. Finally, in Section 5, we conclude the paper and outline directions for future work.

2 BACKGROUND

In this section, we introduce various RNNs, several well-known DL Frameworks, early stopping, and the main design of NP-Free.

2.1 RNN Variants

An RNN is a type of artificial neural network designed to process sequential data or time series (Hopfield, 1982). Unlike traditional feedforward neural networks, RNNs have looping connections that allow them to maintain a hidden state or memory of previous inputs. This recurrent structure makes RNNs well-suited for tasks involving sequential or time series data. In an RNN, each time step in a time series is processed sequentially, with the network handling each element one at a time and updating its internal state based on the current input and previous state. This allows RNNs to capture dependencies and patterns across different time steps. However, RNNs face challenges in capturing long-term dependencies and may suffer from the vanishing gradient problem, which hinders their ability to learn from distant past inputs.

LSTM (Hochreiter and Schmidhuber, 1997) is an RNN variant designed to capture long-term dependencies and model temporal sequences. The structural framework of an LSTM resembles that of conventional RNN, with a key distinction being the presence of memory blocks as nonlinear units within each hidden layer. Each memory block operates autonomously, housing its own memory cells, and is equipped with three essential gates: the input gate, the output gate, and the forget gate. The input gate determines whether incoming data should be stored in the memory cell. The output gate decides whether the current content of the memory should be output. The forget gate determines whether the existing content within the memory cell should be retained or erased. The use of these gates allows LSTM to address the vanishing gradient problem (Hochreiter, 1998) by enabling gradients to flow unchanged.

Gated Recurrent Unit (GRU), introduced by Cho et al. (Cho et al., 2014), is another RNN variant designed to adaptively capture dependencies at various time scales. The core concept of GRU is to utilize gating mechanisms to selectively update the hidden state of the network at each time step. These mechanisms manage the flow of information into and out of the network. GRU consists of two key components: a reset gate and an update gate. The reset gate controls how much of the past information to forget, while the update gate determines how much of the new information to add.

Bidirectional Long Short-Term Memory (BiLSTM) (Graves and Schmidhuber, 2005) is an extension of the standard LSTM network that improves its ability to capture context from both past and future data in a sequence. It consists of two LSTM layers: one processes the input sequence in the forward direction (left to right), and the other processes it in the backward direction (right to left). By combining the outputs from both directions, BiLSTM can better understand the full context of the data, making it particularly useful for tasks like speech recognition, natural language processing, and time series prediction.

Bidirectional Gated Recurrent Unit (BiGRU) (Liu et al., 2021) is an extension of the standard GRU network designed to capture information from both past and future contexts in sequential data. Similar to BiLSTM, BiGRU consists of two GRU layers: one processes the input sequence in the forward direction, and the other processes it in the backward direction, aiming to better capture the full context of the data.

2.2 DL Frameworks

In recent years, several DL frameworks have been developed by academia, industry, and open-source communities. These frameworks share the goal of providing high-level abstractions and application programming interfaces (APIs) for building, training, and deploying deep learning models. Such abstractions simplify the complex process of designing neural networks, allowing practitioners to focus on solving their specific problems rather than dealing with low-level implementation details (Ketkar and Santana, 2017).

TensorFlow (Abadi et al., 2016) is an open-source DL framework developed by the Google Brain team and is one of the most popular DL frameworks. TensorFlow uses dataflow graphs to encapsulate both the computational logic of an algorithm and the corresponding state on which the algorithm operates. This means that users can define the entire computation graph before executing it. TensorFlow supports a wide range of neural network architectures and can utilize hardware acceleration using graphics processing units (GPUs) to speed up model training and inference for both small-scale and large-scale applications. However, TensorFlow's complexity arises from its low-level API, which can be challenging to use. To improve its user-friendliness and accessibility, TensorFlow is often paired with Keras (Keras, 2023), a popular Python library known for its high-level, modular, and user-friendly API.

PyTorch (Paszke et al., 2019) is an open-source deep learning framework that offers a flexible and user-friendly environment for developing and training machine learning models, particularly neural networks. It is widely used in various AI and deep learning applications, such as computer vision and natural language processing. PyTorch stands out with its high-performance C++ runtime, allowing developers to deploy models in production environments without relying on Python for inference (Ketkar and Santana, 2017). PyTorch is known for its dynamic computational graph, enabling flexible model architecture design and easier debugging. It also places a strong emphasis on tensor computation and benefits from robust GPU acceleration capabilities. Additionally, PyTorch supports the ONNX format, facilitating easy model interchangeability.

Deeplearning4j, introduced by Skymind in 2014 (Deeplearning4j, 2023; Wang et al., 2019), is an open-source distributed deep learning framework designed exclusively for the Java programming language and the Java Virtual Machine (JVM) environment. It aims to bring deep neural networks and machine learning capabilities to the JVM ecosystem. Deeplearning4j is known for its scalability and compatibility with popular programming languages, allowing Java and Scala developers to build and train deep learning models. Key features include support for various neural network architectures, distributed computing capabilities, compatibility with Hadoop and Spark for big data processing, and integration with other deep learning libraries like Keras. However, compared to PyTorch, Deeplearning4j has a steeper learning curve due to its lower-level APIs and the need for a solid understanding of Java and deep learning concepts. Additionally, development, updates, and new features for Deeplearning4j may not be as rapid as those for other DL frameworks.

2.3 Early Stopping

Early stopping (EarlyStopping, 2023) is a technique used during the training of machine learning models, particularly neural networks, to prevent overfitting. Overfitting occurs when a model learns the training data too well, including its noise and outliers, resulting in poor generalization to new, unseen data. The basic idea of early stopping is to monitor the model's performance on a validation dataset during training. Training is stopped when the performance on the validation set begins to degrade, indicating that the model has started to overfit the training data.

The detailed workflow of early stopping involves splitting the dataset into training, validation, and test sets. During training, the model's performance on the validation set is continuously monitored. If the performance does not improve for a specified number of epochs, known as the patience parameter, training is stopped. The model parameters from the epoch with the best validation performance are then used. This approach helps ensure the model generalizes well to new, unseen data by stopping the training process before overfitting occurs.

2.4 NP-Free

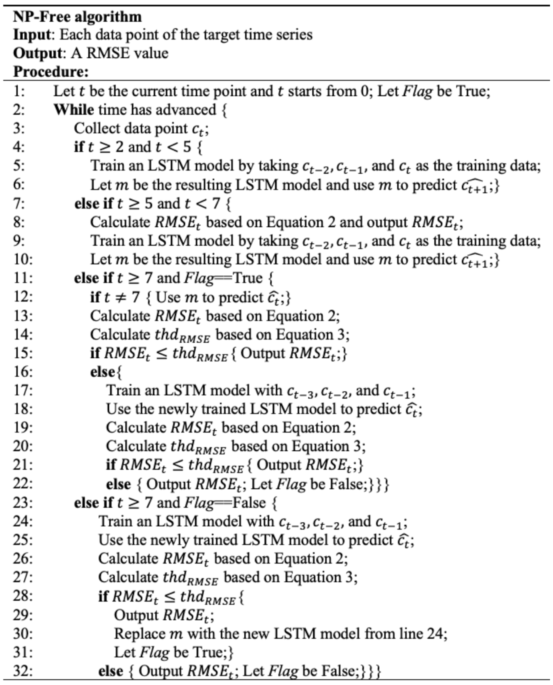
NP-Free, introduced by Lee et al. (Lee et al., 2023), is a real-time time series representation approach that eliminates the need for z-normalization and parameter tuning. It directly transforms raw time series into root-mean-square error (RMSE) series in real time, serving as an alternative to z-normalization in clustering applications.

NP-Free utilizes Long Short-Term Memory (LSTM) and the Look-Back and Predict-Forward strategy from RePAD (Lee et al., 2020b) to generate time series representations. Specifically, NP-Free predicts the next data point based on three historical data points and calculates the RMSE between the observed and predicted values, converting the target time series into an RMSE series. Figure 1 illustrates the pseudo code of NP-Free, where t denotes the current time point, starting from 0. Let c_t be the real data point collected at time point t, and $\widehat{c_t}$ be the data point predicted by NP-Free at t. NP-Free uses three data points to predict the next one. The first LSTM model is trained at t = 2 with c_0, c_1, and c_2 as input, and it predicts $\widehat{c_3}$. This process repeats as t advances, continuously training new LSTM models and making predictions based on the three most recent data points.

At t = 5, NP-Free computes the prediction error using the well-known Root-Mean-Square Error (RMSE) metric, as shown in Equation 1.

$$RMSE_t = \sqrt{\frac{\sum_{z=t-2}^{t} (c_z - \widehat{c_z})^2}{3}}, t \geq 5 \tag{1}$$

After deriving $RMSE_5$, NP-Free predicts $\widehat{c_6}$ (see lines 9 and 10 of Figure 1). At t = 6, NP-Free repeats the procedure to calculate $RMSE_6$ and predict $\widehat{c_7}$. When t = 7, NP-Free calculates $RMSE_7$ and thd_{RMSE} using Equation 2.



```
NP-Free algorithm
Input: Each data point of the target time series
Output: A RMSE value
Procedure:
1:  Let $t$ be the current time point and $t$ starts from 0; Let $Flag$ be True;
2:  While time has advanced {
3:     Collect data point $c_t$;
4:     if $t \geq 2$ and $t < 5$ {
5:        Train an LSTM model by taking $c_{t-2}, c_{t-1}$, and $c_t$ as the training data;
6:        Let $m$ be the resulting LSTM model and use $m$ to predict $\widehat{c_{t+1}}$;}
7:     else if $t \geq 5$ and $t < 7$ {
8:        Calculate $RMSE_t$ based on Equation 2 and output $RMSE_t$;
9:        Train an LSTM model by taking $c_{t-2}, c_{t-1}$, and $c_t$ as the training data;
10:       Let $m$ be the resulting LSTM model and use $m$ to predict $\widehat{c_{t+1}}$;}
11:    else if $t \geq 7$ and $Flag$==True {
12:       if $t \neq 7$ { Use $m$ to predict $\widehat{c_t}$;}
13:       Calculate $RMSE_t$ based on Equation 2;
14:       Calculate $thd_{RMSE}$ based on Equation 3;
15:       if $RMSE_t \leq thd_{RMSE}$ { Output $RMSE_t$;}
16:       else{
17:          Train an LSTM model with $c_{t-3}, c_{t-2}$, and $c_{t-1}$;
18:          Use the newly trained LSTM model to predict $\widehat{c_t}$;
19:          Calculate $RMSE_t$ based on Equation 2;
20:          Calculate $thd_{RMSE}$ based on Equation 3;
21:          if $RMSE_t \leq thd_{RMSE}$ { Output $RMSE_t$;}
22:          else { Output $RMSE_t$; Let $Flag$ be False;}}}
23:    else if $t \geq 7$ and $Flag$==False {
24:       Train an LSTM model with $c_{t-3}, c_{t-2}$, and $c_{t-1}$;
25:       Use the newly trained LSTM model to predict $\widehat{c_t}$;
26:       Calculate $RMSE_t$ based on Equation 2;
27:       Calculate $thd_{RMSE}$ based on Equation 3;
28:       if $RMSE_t \leq thd_{RMSE}$ {
29:          Output $RMSE_t$;
30:          Replace $m$ with the new LSTM model from line 24;
31:          Let $Flag$ be True;}
32:       else { Output $RMSE_t$; Let $Flag$ be False;}}}
```

Figure 1: The pseudo code of NP-Free (Lee et al., 2023).

$$thd_{RMSE} = \mu_{RMSE} + 3 \cdot \sigma \tag{2}$$

In Equation 2, μ_{RMSE} and σ represent the average RMSE and standard deviation at time point t, calculated using Equations 3 and 4, respectively.

$$\mu_{RMSE} = \begin{cases} \frac{1}{t-4} \cdot \sum_{z=5}^{t} RMSE_z, 7 \leq t < w+4 \\ \frac{1}{w} \cdot \sum_{z=t-w+1}^{t} RMSE_z, \; t \geq w+4 \end{cases} \tag{3}$$

$$\sigma = \begin{cases} \sqrt{\frac{\sum_{z=5}^{t} (RMSE_z - \mu_{RMSE})^2}{t-4}}, 7 \leq t < w+4 \\ \sqrt{\frac{\sum_{z=t-w+1}^{t} (RMSE_z - \mu_{RMSE})^2}{w}}, \; t \geq w+4 \end{cases} \tag{4}$$

Here, w limits the number of historical RMSE values considered to prevent exhausting system resources.

Whenever the time point t advances to 7 or beyond (i.e., line 11 or line 23 of Figure 1 evaluates to true), NP-Free recalculates $RMSE_t$ and thd_{RMSE}. If $RMSE_t$ is not greater than the threshold (as indicated in lines 15 and 28), NP-Free immediately outputs $RMSE_t$. Otherwise, NP-Free attempts to adapt to potential pattern changes by retraining a new LSTM model to re-predict $\widehat{c_t}$ and recalculate both $RMSE_t$ and thd_{RMSE} either at the current time point (lines 17 to 20) or the next (lines 24 to 27). If the recalculated $RMSE_t$ is no larger than thd_{RMSE}, NP-Free immediately outputs $RMSE_t$. Otherwise, it outputs $RMSE_t$ and performs LSTM model retraining at the next time point. This iterative process dynamically converts a time series into an RMSE series on the fly.

As previously mentioned, NP-Free distinguishes itself from conventional representation methods by eliminating preprocessing steps like z-normalization. This feature allows NP-Free to serve as an alternative normalization approach in clustering applications.

3 RELATED WORK

Several studies have compared DL frameworks. For example, Kovalev et al. (Kovalev et al., 2016) evaluated the training time, prediction time, and classification accuracy of a fully connected neural network using five different DL frameworks: Theano with Keras, Torch, Caffe, TensorFlow, and Deeplearning4j. Zhang et al. (Zhang et al., 2022) introduced a benchmark that included six DL frameworks, various mobile devices, and fifteen DL models for image classification, object detection, semantic segmentation, and text classification. Their analysis revealed that no single DL framework is superior across all tested scenarios and highlighted that the influence of DL frameworks may surpass both DL algorithm design and hardware capacity considerations. Despite the valuable insights provided by these studies, their findings do not address our specific question regarding the influence of different RNNs, DL frameworks, and the early stopping function on real-time time series representation.

Nguyen et al. (Nguyen et al., 2019) surveyed various DL frameworks, analyzing their strengths and weaknesses, but did not perform experimental comparisons. Wang et al. (Wang et al., 2019) assessed several DL frameworks on interface properties, deployment capabilities, performance, and design, providing recommendations for different scenarios. However, neither study addresses the specific question of this paper: the impact of RNN variants, DL frameworks, and the early stopping function on real-time time series representation.

A work more closely related to our paper is the study conducted by Lee and Lin (Lee and Lin, 2023). In their research, they evaluated the impact of three DL frameworks—TensorFlow with Keras, PyTorch, and Deeplearning4j—on two real-time lightweight time series anomaly detection approaches, RePAD (Lee et al., 2020b) and SALAD (Lee et al., 2021b). Their results indicated that DL frameworks significantly impact the detection accuracy of the two selected approaches. However, it is important to note that their evaluation did not consider the impact of different RNN variants, as the two approaches were exclusively implemented using one type of RNN, specifically LSTM, and focused on real-time time series anomaly detection. Consequently, there is a knowledge gap regarding the influence of RNN variants, DL frameworks, and the early stopping function on real-time time series representation.

4 EVALUATION

In this section, we detail how we conducted a comparative analysis of real-time time series representation. Recall that NP-Free was originally implemented using LSTM in Deeplearning4j. To understand the impact of various RNNs, DL frameworks, and early stopping on the performance of NP-Free, we implemented NP-Free using five different types of RNNs: RNN, LSTM, GRU, Bi-LSTM, and Bi-GRU, across three different DL frameworks: TensorFlow-Keras, PyTorch, and Deeplearning4j, both with and without early stopping.

In our evaluation, we used TensorFlow-Keras version 2.9.1, PyTorch version 1.13.1, and Deeplearning4j version 0.7-SNAPSHOT. It is important to note that Deeplearning4j officially supports only the LSTM architecture; it does not support RNN, Bi-LSTM, GRU, or Bi-GRU. Consequently, we implemented NP-Free using the LSTM architecture within the Deeplearning4j framework, referring to this specific implementation as DL4J-LSTM, which denotes the use of LSTM in Deeplearning4j for NP-Free.

A similar issue arises with PyTorch. PyTorch officially supports RNN, LSTM, and GRU but does not support the other two RNN variants. Due to this limitation, we could only implement NP-Free with the architectures supported by PyTorch: RNN, LSTM, and GRU. These implementations are referred to as PT-RNN, PT-LSTM, and PT-GRU, respectively.

Additionally, to assess the impact of early stopping on real-time time series representation across different RNNs and DL frameworks, we considered two scenarios. In Scenario 1, early stopping was not adopted by each implementation. In this case, each implementation was configured with the epoch parameter set to 50 to ensure fairness and consistency. Table 1 lists all implementations studied in Scenario 1. The term "N/A" indicates that the corresponding implementation is not available due to lack of support from the corresponding DL framework.

Table 1: The nine implementations studied in Scenario 1.

	PyTorch	TensorFlow-Keras	Deeplearning4j
RNN	PT-RNN	TFK-RNN	N/A
LSTM	PT-LSTM	TFK-LSTM	DL4J-LSTM
GRU	PT-GRU	TFK-GRU	N/A
BiLSTM	N/A	TFK-BiLSTM	N/A
BiGRU	N/A	TFK-BiGRU	N/A

Conversely, in Scenario 2, early stopping was adopted. It is important to note that early stopping (EarlyStopping, 2023) was not officially supported by PyTorch at the time of evaluation. Therefore, in Scenario 2, we excluded all implementations related to PyTorch, resulting in only six implementations as shown in Table 2 being evaluated and compared.

Table 2: The six implementations studied in Scenario 2.

	TensorFlow-Keras	Deeplearning4j
RNN	TFK-RNN	N/A
LSTM	TFK-LSTM	DL4J-LSTM
GRU	TFK-GRU	N/A
BiLSTM	TFK-BiLSTM	N/A
BiGRU	TFK-BiGRU	N/A

4.1 Configuration and Environment

To guarantee a fair evaluation, all implementations were configured with identical hyperparameters and parameters, as detailed in Table 3. These settings were originally suggested and employed in prior studies by (Lee et al., 2021a; Lee et al., 2023; Lee et al., 2024b). We adopted these settings for all our experiments. Each implementation consists of a single hidden layer with 10 hidden units and uses three historical data points (Look-Back parameter) to predict the next data point (Predict-Forward parameter). The models were trained for 50 epochs with a learning rate of 0.005, using the tanh activation function and a fixed random seed of 140 to ensure reproducibility. Additionally, the patience parameter of 5, the default setting in Deeplearning4j, was used when the early stopping function was activated.

Table 3: Configuration used for all implementations.

Hyperparameters/parameters	Value
The Look-Back parameter	3
The Predict-Forward parameter	1
The number of hidden layers	1
The number of hidden units	10
The number of epochs	50
Learning rate	0.005
Activation function	tanh
Random seed	140
Patience parameter	5

The evaluation of each implementation was conducted separately on a MacBook running macOS 14.5, equipped with a 2.6 GHz 6-Core Intel Core i7 processor and 16GB DDR4 SDRAM. It is important to note that the decision to use a standard laptop, without GPUs or high-performance computing resources, was intentional. This approach aims to assess how the combination of RNN variants, DL frameworks, and early stopping impacts the performance of real-time time series representation in a typical computing environment.

4.2 Real-World Time Series Data

To evaluate the nine implementations, we used a real-world open-source time series dataset collected by the Human Dynamics and Controls Laboratory at the University of Illinois at Urbana-Champaign (Helwig and Hsiao-Wecksler, 2022), available from the UC Irvine Machine Learning Repository (Helwig and Hsiao-Wecksler, 2022). This dataset is related to multivariate gait time series for biomechanical analysis of human locomotion. It consists of bilateral (left, right) joint angle (ankle, knee, hip) time series data collected from 10 subjects under three walking conditions: unbraced (normal walking on a treadmill), knee-braced (walking on a treadmill with a knee brace on the right knee), and ankle-braced (walking on a treadmill with an ankle brace on the right ankle).

For each condition, each subject's data comprises 10 consecutive gait cycles (replications), where each gait cycle starts and ends at heel-strike. Six joint angles are included, which cover all combinations of leg (left and right) and joint (ankle, knee, hip). Thus, this dataset forms a six-dimensional array of joint angle data: 10 subjects × 3 conditions × 10 replications × 2 legs × 3 joints × 101 time points. The total number of time series in this dataset is 1800, with each time series consisting of 101 data points.

4.3 Evaluation Methodology

To evaluate the representation ability of each implementation and its impact on a time series classification task, we analyzed the dataset to identify which subject had the most stable time series under a specific combination of walking condition, leg, and joint in their 10 replications. By 'stable', we mean that the 10 time series in the 10 replications are similar to each other. Once such a subject and combination were identified, each implementation in Scenarios 1 and 2 was applied to generate a representation (i.e., an RMSE series) for each of the subject's time series under that specific combination. The representation quality and time efficiency were then evaluated. Finally, their impact on time series classification were assessed.

To achieve the above evaluation, we first calculated the average Euclidean distance (ED) for all subjects under a specific combination of walking condition, leg, and joint after applying the min-max normalization (Codecademy, 2024) on each time series. As shown in Table 4, the combination of Unbraced_Left_Knee resulted in the smallest average ED with the smallest standard deviation (SD). In other words, all subjects exhibit stable time series under the Unbraced_Left_Knee combination. This is illustrated in Figure 2, where each subject has 10 similar time series collected from their left knee when unbraced.

Table 4: Average Euclidean distance of all subjects' time series under different combinations.

Combination	Average ED	SD
Unbraced_Left_Ankle	$6.65 \cdot 10^{-3}$	$2.45 \cdot 10^{-3}$
Unbraced_Left_Knee	$2.96 \cdot 10^{-3}$	$1.07 \cdot 10^{-3}$
Unbraced_Left_Hip	$3.04 \cdot 10^{-3}$	$1.14 \cdot 10^{-3}$
Unbraced_Right_Ankle	$6.14 \cdot 10^{-3}$	$2.13 \cdot 10^{-3}$
Unbraced_Right_Knee	$3.29 \cdot 10^{-3}$	$1.24 \cdot 10^{-3}$
Unbraced_Right_Hip	$3.24 \cdot 10^{-3}$	$1.26 \cdot 10^{-3}$
KneeBrace_Left_Ankle	$8.76 \cdot 10^{-3}$	$3.33 \cdot 10^{-3}$
KneeBrace_Left_Knee	$4.19 \cdot 10^{-3}$	$1.60 \cdot 10^{-3}$
KneeBrace_Left_Hip	$4.00 \cdot 10^{-3}$	$1.51 \cdot 10^{-3}$
KneeBrace_Right_Ankle	$11.07 \cdot 10^{-3}$	$3.24 \cdot 10^{-3}$
KneeBrace_Right_Knee	$8.83 \cdot 10^{-3}$	$2.72 \cdot 10^{-3}$
KneeBrace_Right_Hip	$4.24 \cdot 10^{-3}$	$1.45 \cdot 10^{-3}$
AnkleBrace_Left_Ankle	$7.81 \cdot 10^{-3}$	$2.75 \cdot 10^{-3}$
AnkleBrace_Left_Knee	$4.07 \cdot 10^{-3}$	$1.43 \cdot 10^{-3}$
AnkleBrace_Left_Hip	$4.22 \cdot 10^{-3}$	$1.52 \cdot 10^{-3}$
AnkleBrace_Right_Ankle	$11.06 \cdot 10^{-3}$	$3.52 \cdot 10^{-3}$
AnkleBrace_Right_Knee	$4.86 \cdot 10^{-3}$	$1.57 \cdot 10^{-3}$
AnkleBrace_Right_Hip	$4.04 \cdot 10^{-3}$	$1.51 \cdot 10^{-3}$

Following the previous experiment, we continued to identify which subject has the most stable time series under the Unbraced_Left_Knee combination. To do it, we separately calculated the average ED for each subject under the Unbraced_Left_Knee combination and present the results in Table 5. It is apparent that subject S9 has the lowest average ED with the smallest SD. This can be confirmed from the subfigure for subject S9 in Figure 2, where all 10 time series are almost overlapping.

Table 5: Average Euclidean distance of each subjects' time series under the Unbraced_Left_Knee combination.

Subject	Average ED	SD
S1	$2.91 \cdot 10^{-3}$	$0.80 \cdot 10^{-3}$
S2	$2.51 \cdot 10^{-3}$	$1.01 \cdot 10^{-3}$
S3	$2.72 \cdot 10^{-3}$	$1.19 \cdot 10^{-3}$
S4	$3.25 \cdot 10^{-3}$	$1.02 \cdot 10^{-3}$
S5	$2.19 \cdot 10^{-3}$	$0.86 \cdot 10^{-3}$
S6	$3.54 \cdot 10^{-3}$	$1.45 \cdot 10^{-3}$
S7	$2.81 \cdot 10^{-3}$	$0.91 \cdot 10^{-3}$
S8	$4.07 \cdot 10^{-3}$	$1.63 \cdot 10^{-3}$
S9	$2.15 \cdot 10^{-3}$	$0.62 \cdot 10^{-3}$
S10	$3.40 \cdot 10^{-3}$	$1.21 \cdot 10^{-3}$

Based on the above results, we used subject S9's 10 time series under the Unbraced_Left_Knee combination to evaluate each implementation in Scenarios 1 and 2. Because these 10 time series are the most similar to each other, they provide a suitable basis for achieving a fair and realistic comparison and evaluation among different implementations.

4.4 Scenario 1

In Scenario 1, all nine implementations of NP-Free did not adopt early stopping. We used each implementation to generate an RMSE series for each of subject S9's time series under the Unbraced_Left_Knee combination and then calculate the average ED for the 10 generated RMSE series. Additionally, we measured the time each implementation took to generate an RMSE series, referred to as transformation time in this paper.

Table 6 shows the results of each implementation. We can see that DL4J-LSTM outperforms all the other implementations because it resulted in the smallest average ED among them. In other words, the RMSE series generated by DL4J-LSTM are more similar to each other compared to the RMSE series generated by any other implementation. This phenomenon can be observed in Figure 3. Apparently, the 10 RMSE series generated by DL4J-LSTM had a high degree of overlap compared to the RMSE series generated by other implementations.

However, in terms of transformation time, DL4J-LSTM performs well, but not the best. Instead, all three implementations related to PyTorch are the most time-efficient, particularly PT-LSTM, which had an average transformation time of 1.52 seconds. Nevertheless, all PyTorch-related implementations resulted in a much higher ED than DL4J-LSTM, implying that PyTorch cannot guarantee to generate a stable RMSE series to represent the original time series.

Table 6: Performance of each implementation in Scenario 1.

	ED of RMSE series (10^{-3})		Transformation time (sec)	
	Average	SD	Average	SD
DL4J-LSTM	**3.19**	**0.90**	8.20	0.44
TFK-RNN	25.81	7.29	24.77	4.26
TFK-LSTM	16.96	5.14	79.20	2.99
TFK-GRU	21.58	6.79	77.69	2.02
TFK-BiLSTM	17.59	5.07	144.81	3.00
TFK-BiGRU	22.63	5.55	141.09	5.77
PT-RNN	22.03	9.48	2.09	0.16
PT-LSTM	15.25	4.93	**1.52**	**0.10**
PT-GRU	18.10	6.54	2.08	0.29

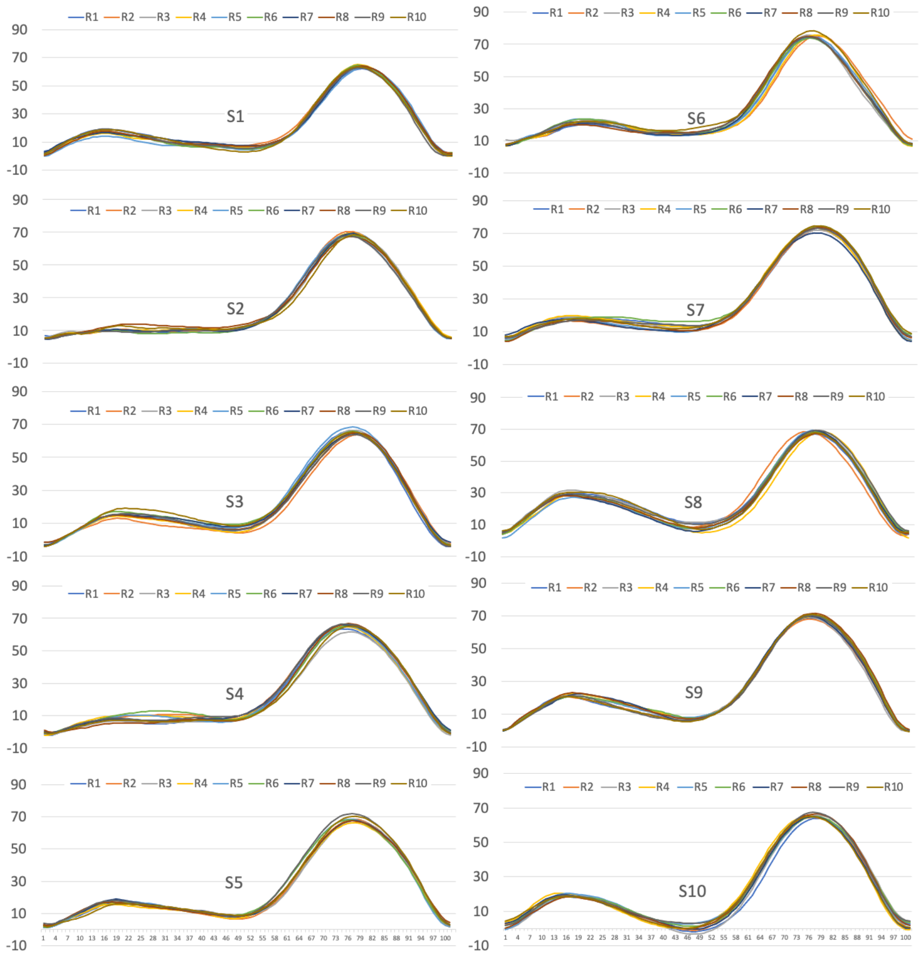


Figure 2: The original gait time series of each subject under the Unbraced_Left_Knee combination.

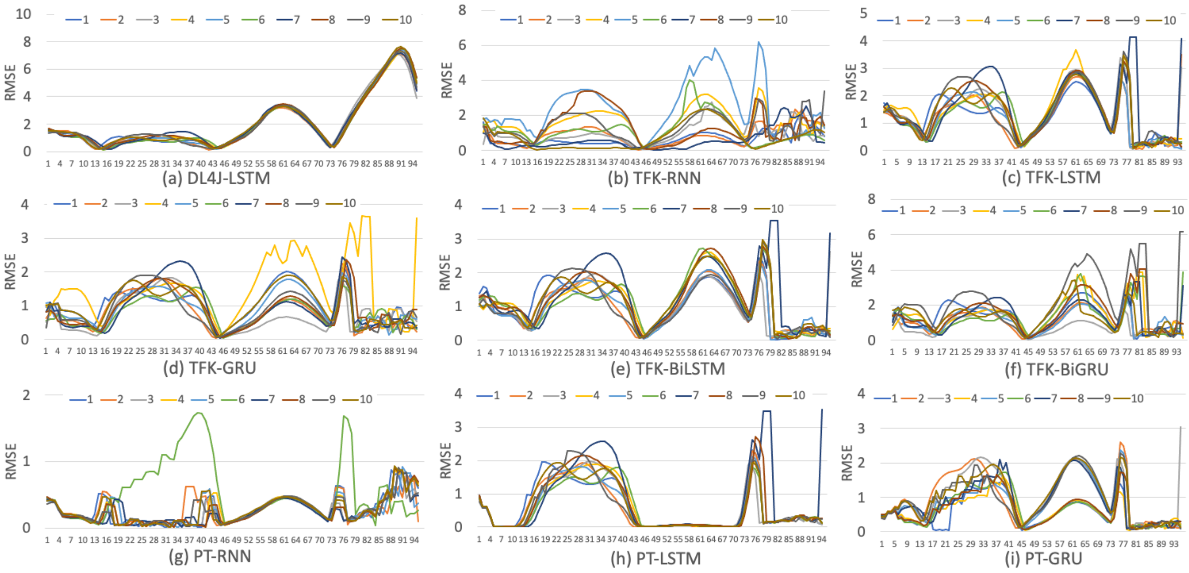


Figure 3: Visualization of RMSE series generated by each implementation in Scenario 1.

Among the three DL frameworks studied in this paper, TensorFlow-Keras resulted in the worst performance in terms of both representation ability and transformation time, regardless of the RNNs used. Similar poor results were also observed by Lee et al. in their study in (Lee and Lin, 2023; Lee et al., 2024a).

To further evaluate the impact of each implementation to time series classification, we used each implementation to transform each raw time series of each subject under the Unbraced_Left_Knee combination into an RMSE series. We then evaluated how accurately the well-known k-means algorithm from the tslearn package (Tavenard et al., 2020), a Python library specifically designed for time series analysis, could classify RMSE series into their corresponding subjects.

Table 7 lists the classification accuracy rate achieved by each implementation in Scenario 1. DL4J-LSTM resulted in the highest accuracy rate of 84%, indicating that 84 out of 100 time series were correctly classified by the k-means algorithm into their corresponding subjects, with only 16 incorrectly classified. This good performance is attributed to DL4J-LSTM's superior ability to generate stable and similar RMSE series representations for any specific subject.

Table 7: The classification accuracy rate achieved by each implementation in Scenario 1.

Implementation	Classification accuracy rate
DL4J-LSTM	84% (= 84/100)
TFK-RNN	0% (= 0/100)
TFK-LSTM	55% (= 55/100)
TFK-GRU	42% (= 42/100)
TFK-BiLSTM	55% (= 55/100)
TFK-BiGRU	41% (= 41/100)
PT-RNN	43% (= 43/100)
PT-LSTM	54% (= 54/100)
PT-GRU	54% (= 54/100)

On the contrary, TFK-RNN performed the worst among all the implementations because none of the RMSE series generated by TFK-RNN could be correctly classified by the k-means algorithm, leading to the classification accuracy rate of 0. This can be explained by the fact that it led to the highest average ED, as shown in Table 6. Although TensorFlow-Keras in combination with the other RNNs resulted in a higher classification accuracy rate, the results are still not satisfactory. Similarly, all PyTorch-related implementations resulted in unsatisfactory classification accuracy, ranging between 43% and 54%. This is because these implementations were unable to generate stable and similar RMSE series for any specific subject.

In summary, DL4J-LSTM proved to be a suitable implementation choice for NP-Free when early stopping was not adopted, whereas the other implementations were not suitable for NP-Free.

Note that while the RMSE series generated by DL4J-LSTM are more similar to each other, they do not indicate better prediction accuracy compared to the time series predictions of the TFK and PT implementations. As shown in Figure 3, most RMSE values fall between 0 and 3.5 for the PT implementations, between 0 and 6.1 for TFK implementations, and between 0 and 7.8 for DL4J-LSTM. Since lower RMSE values correspond to higher prediction accuracy, this scenario shows that although the prediction accuracy of TFK and PT implementations surpasses that of DL4J-LSTM, they do not produce RMSE series as consistent as those generated by DL4J-LSTM.

4.5 Scenario 2

In Scenario 2, we evaluated all implementations of NP-Free with early stopping enabled. Recall that early stopping was not officially supported by PyTorch at the time of evaluation, so the three implementations related to PyTorch were excluded. Similar to Scenario 1, we used each of the six implementations to generate an RMSE series for each of subject S9's time series under the Unbraced_Left_Knee combination and then calculate the average ED for the 10 generated RMSE series. Furthermore, we measured the transformation time each implementation took to generate an RMSE series.

Table 8 lists the performance of each implementation. It is clear to see that DL4J-LSTM performs the best among all the compared implementations, as it resulted in the smallest average ED. This indicates that the ten RMSE series transformed by DL4J-LSTM for subject S9 under the Unbraced_Left_Knee combination are closer to each other compared to the RMSE series transformed by any other implementation for the same subject under the same combination. This can be observed in Figure 4. In other words, DL4J-LSTM provides the best representation ability to preserve the characteristics of the original time series, even with the adoption of early stopping. Furthermore, DL4J-LSTM offers the best time efficiency with a transformation time of only 5.97 seconds, making it 3.73, 13.02, 14.70, 22.65, and 22.15 times faster than TFK-RNN, TFK-LSTM, TKF-GRU, TFK-BiLSTM, and TFK-BiGRU, respectively.

We continued to evaluate how each implementation impacts time series classification by employing the k-means algorithm to classify all the RMSE series

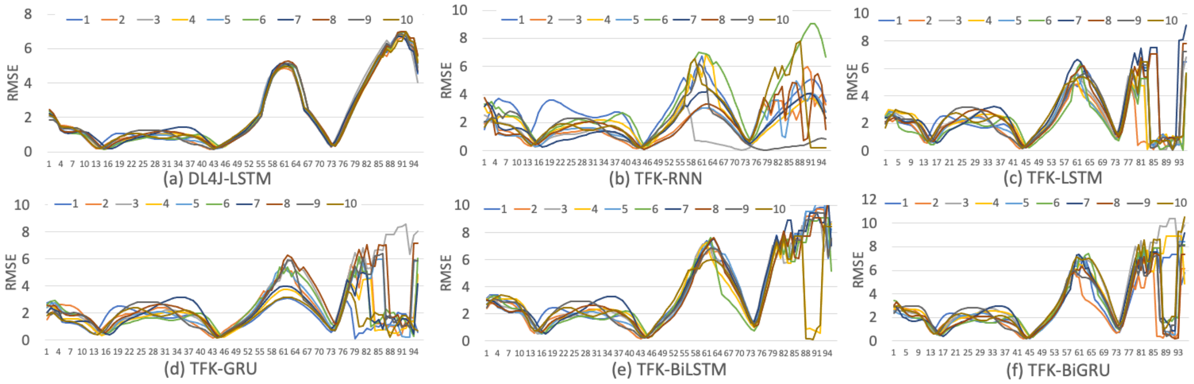


Figure 4: Visualization of RMSE series generated by each implementation in Scenario 2.

Table 8: Performance of each implementation in Scenario 2.

	ED of RMSE series (10^{-3})		Transformation time (sec)	
	Average	SD	Average	SD
DL4J-LSTM	**3.89**	**0.85**	**5.97**	**0.38**
TFK-RNN	20.06	5.52	22.29	2.36
TFK-LSTM	16.01	3.64	77.72	2.46
TFK-GRU	22.65	9.55	87.74	5.12
TFK-BiLSTM	12.63	6.59	135.25	2.94
TFK-BiGRU	16.52	6.26	132.22	4.31

transformed by each implementation, similar to what we did in Scenario 1. Table 9 lists the classification accuracy rate achieved by each implementation. Evidently, DL4J-LSTM with early stopping led to the best classification performance. However, when any TFK-related implementation was tested, they misled k-means, resulting in a classification accuracy rate lower than 60%. Based on the above results, it is confirmed that DL4J-LSTM with early stopping is recommended for implementing NP-Free.

Table 9: The classification accuracy rate achieved by each implementation in Scenario 2.

Implementation	Classification accuracy rate
DL4J-LSTM	94% (= 94/100)
TFK-RNN	0% (= 0/100)
TFK-LSTM	48% (= 48/100)
TFK-GRU	48% (= 48/100)
TFK-BiLSTM	55% (= 55/100)
TFK-BiGRU	59% (= 59/100)

If we cross-compare the results from Scenario 1 and Scenario 2 (please compare Table 6 with Table 8, and compare Table 7 with Table 9), we can see that adopting early stopping for DL4J-LSTM is the most recommended implementation strategy. This approach significantly reduces the average transformation time for each time series from 8.20 seconds to 5.97 seconds. Although it slightly increases the average ED from $3.19 \cdot 10^{-3}$ to $3.89 \cdot 10^{-3}$, it does not negatively affect k-means' classification. Instead, it led to a higher accuracy rate, increasing from 84% to 94%. To understand the reason behind this, we analyzed the results and found that DL4J-LSTM with early stopping was able to generate more distinct and stable RMSE series for each subject's original time series, resulting in a higher classification accuracy rate.

Therefore, DL4J-LSTM with early stopping emerges as the most recommended choice due to its superior ability to preserve the characteristics of the original time series, its time-efficient processing, and its ability to lead k-means algorithm to achieve high classification accuracy.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we investigated how three well-known DL frameworks (TensorFlow-Keras, PyTorch, and Deeplearning4j), five different types of RNNs (RNN, LSTM, GRU, Bi-LSTM, Bi-GRU), and the early stopping function impact real-time time series representation. We conducted a series of experiments using a state-of-the-art real-time time series representation method named NP-Free and real-world, open-source multivariate gait time series data. These experiments evaluated different implementation choices in terms of their representation ability, time efficiency, and impact on time series classification.

The results indicate that RNN variants, DL frameworks, and early stopping significantly impact not only representation quality and time efficiency but also subsequent time series classification. According to the results, TensorFlow-Keras is not recommended, regardless of which RNN is used, because it leads to unstable RMSE series generation and higher time consumption when transforming a time series into an RMSE series. On the other hand, PyTorch is the most efficient DL framework among the three, enabling NP-Free to provide instant processing and RMSE generation. However, similar to TensorFlow-Keras, it generates unstable RMSE series that cannot preserve the characteristics of the original time series.

Deeplearning4j is considered the most suitable DL framework among the three studied. Although it only supports LSTM rather than other RNNs, this combination preserves the characteristics of the original time series in a time-efficient manner, leading to satisfactory classification accuracy, especially when early stopping is enabled. Therefore, DL4J-LSTM with early stopping is the most recommended choice due to its superior ability to preserve the characteristics of the original time series, time-efficient processing, and enabling k-means algorithm to achieve high classification accuracy. Our study offers valuable guidelines for future research on real-time time series representation using deep learning.

In our future work, we plan to enhance the time efficiency of NP-Free by adopting strategies such as reducing the number of hidden units and the number of epochs. Additionally, we intend to release the source code for all the implementations studied in this paper, with the aim of advancing research in this area.

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous editors and reviewers for their reviews and valuable suggestions for this article. This work received funding from the Research Council of Norway through the SFI Norwegian Centre for Cybersecurity in Critical Sectors (NORCICS), project no. 310105.

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: a system for large-scale machine learning. In *Osdi*, volume 16, pages 265–283. Savannah, GA, USA.

Aghabozorgi, S., Shirkhorshidi, A. S., and Wah, T. Y. (2015). Time-series clustering–a decade review. *Information systems*, 53:16–38.

Bagnall, A., Lines, J., Bostrom, A., Large, J., and Keogh, E. (2017). The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31:606–660.

Bagnall, A., Ratanamahatana, C. A., Keogh, E., Lonardi, S., and Janacek, G. (2006). A bit level representation for time series data mining with shape based similarity. *Data mining and knowledge discovery*, 13(1):11–40.

Cho, K., Van Merriënboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Codecademy (2024). Normalization. https://www.codecademy.com/article/normalization. [Online; accessed 25-September-2024].

Dau, H. A., Bagnall, A., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., and Keogh, E. (2019). The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305.

Deeplearning4j (2023). Introduction to core Deeplearning4j concepts. https://deeplearning4j.konduit.ai/. [Online; accessed 24-September-2024].

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552.

EarlyStopping (2023). What is early stopping? https://deeplearning4j.konduit.ai/. [Online; accessed 24-September-2024].

Graves, A. and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

Helwig, N. and Hsiao-Wecksler, E. (2022). Multivariate Gait Data. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5861T.

Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558.

Höppner, F. (2014). Less is more: similarity of time series under linear transformations. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, pages 560–568. SIAM.

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963.

Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2001). Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3:263–286.

Keras (2023). Keras - a deep learning API written in python. https://keras.io/about/. [Online; accessed 25-September-2024].

Ketkar, N. and Santana, E. (2017). *Deep learning with Python*, volume 1. Springer.

Kovalev, V., Kalinovsky, A., and Kovalev, S. (2016). Deep learning with theano, torch, caffe, tensorflow, and deeplearning4j: Which one is the best in speed and accuracy?

Lee, M.-C. and Lin, J.-C. (2023). Impact of deep learning libraries on online adaptive lightweight time series anomaly detection. In *Proceedings of the 18th International Conference on Software Technologies - ICSOFT*, pages 106–116. INSTICC, SciTePress. https://arxiv.org/pdf/2305.00595.

Lee, M.-C., Lin, J.-C., and Gan, E. G. (2020a). ReRe: A lightweight real-time ready-to-go anomaly detection approach for time series. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 322–327. IEEE. arXiv preprint arXiv:2004.02319.

Lee, M.-C., Lin, J.-C., and Gran, E. G. (2020b). RePAD: real-time proactive anomaly detection for time series. In *Advanced Information Networking and Applications: Proceedings of the 34th International Conference on Advanced Information Networking and Applications (AINA-2020)*, pages 1291–1302. Springer. arXiv preprint arXiv:2001.08922.

Lee, M.-C., Lin, J.-C., and Gran, E. G. (2021a). How far should we look back to achieve effective real-time time-series anomaly detection? In *Advanced Information Networking and Applications: Proceedings of the 35th International Conference on Advanced Information Networking and Applications (AINA-2021), Volume 1*, pages 136–148. Springer. arXiv preprint arXiv:2102.06560.

Lee, M.-C., Lin, J.-C., and Gran, E. G. (2021b). SALAD: Self-adaptive lightweight anomaly detection for real-time recurrent time series. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 344–349. IEEE.

Lee, M.-C., Lin, J.-C., and Katsikas, S. (2024a). Impact of recurrent neural networks and deep learning frameworks on real-time lightweight time series anomaly detection. *The 26th International Conference on Information and Communications Security, 26-28 August, 2024, Mytilene, Lesvos, Greece (ICICS2024), arXiv preprint arXiv:2407.18439*.

Lee, M.-C., Lin, J.-C., and Stolz, V. (2023). NP-Free: A Real-Time Normalization-free and Parameter-tuning-free Representation Approach for Open-ended Time Series. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 334–339. IEEE. https://arxiv.org/pdf/2304.06168.

Lee, M.-C., Lin, J.-C., and Stolz, V. (2024b). Evaluation of K-Means Time Series Clustering Based on Z-Normalization and NP-Free. In *Proceedings of the 13th International Conference on Pattern Recognition Applications and Methods - ICPRAM*, pages 469–477. INSTICC, SciTePress. https://arxiv.org/pdf/2401.15773.

Lin, J., Keogh, E., Wei, L., and Lonardi, S. (2007). Experiencing sax: a novel symbolic representation of time series. *Data Mining and knowledge discovery*, 15:107–144.

Liu, X., Wang, Y., Wang, X., Xu, H., Li, C., and Xin, X. (2021). Bi-directional gated recurrent unit neural network based nonlinear equalizer for coherent optical communication system. *Optics Express*, 29(4):5923–5933.

Nguyen, G., Dlugolinsky, S., Bobák, M., Tran, V., López García, Á., Heredia, I., Malík, P., and Hluchỳ, L. (2019). Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey. *Artificial Intelligence Review*, 52:77–124.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Ratanamahatana, C., Keogh, E., Bagnall, A. J., and Lonardi, S. (2005). A novel bit level time series representation with implication of similarity search and clustering. In *Advances in Knowledge Discovery and Data Mining: 9th Pacific-Asia Conference, PAKDD 2005, Hanoi, Vietnam, May 18-20, 2005. Proceedings 9*, pages 771–777. Springer.

Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., and Woods, E. (2020). Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6.

Wang, Z., Liu, K., Li, J., Zhu, Y., and Zhang, Y. (2019). Various frameworks and libraries of machine learning and deep learning: a survey. *Archives of computational methods in engineering*, pages 1–24.

Zhang, Q., Li, X., Che, X., Ma, X., Zhou, A., Xu, M., Wang, S., Ma, Y., and Liu, X. (2022). A comprehensive benchmark of deep learning libraries on mobile devices. In *Proceedings of the ACM Web Conference 2022*, pages 3298–3307.

Multilayer Networks: For Modeling and Analysis of Big Data

Abhishek Santra, Hafsa Billah and Sharma Chakravarthy
Information Technology Lab, CSE Department, University of Texas at Arlington, Texas, U.S.A.
{*abhishek.santra, uxb7123*}@*mavs.uta.edu, sharmac@cse.uta.edu*

Keywords: Multilayer Networks, Modeling, Analysis, Big Data.

Abstract: In this **position paper**, we make a case for the appropriateness, utility, and effectiveness of graph models for big data analysis focusing on Multilayer Networks (or MLNs) – a specific type of graph. MLNs have been shown to be more appropriate for modeling complex data compared to their traditional counterparts. MLNs have also been shown to be useful for diverse data types, such as videos and information integration. Further, MLNs have been shown to be flexible for computing analysis objectives from diverse application domains using extant and new algorithms. There is research for automating the modeling of MLNs using widely used EER (Enhanced/Extended Entity Relationship) or Unified Modeling Language (UML) approaches.

We start by discussing different graph models and their benefits and limitations. We demonstrate how MLNs can be effectively used to model applications with complex data. We also summarize the work on the use of EER models to generate MLNs in a principled manner. We elaborate on analysis alternatives provided by MLNs and their ability to match analysis needs. We show the use of MLNs for - i) traditional data analysis, ii) video content analysis, iii) complex data analysis, and iv) propose the use of MLNs for information integration or fusion. We show examples drawn from the literature of their modeling and analysis usage. We conclude that graphs, specifically MLNs provide a rich alternative to model and analyze big data. Of course, this certainly does not preclude newer data models that are likely to come along.

1 INTRODUCTION

Big data analytics is predicated upon our ability to model and analyze disparate, complex data sets and associated application objectives. Relational and object-oriented data models have served well for modeling and analyzing transactional data sets that need to be managed over long periods. NoSQL data models filled the gap in modeling and analysis for data sets for which earlier data models were not best suited. New data models including graph models are gaining importance due to the diverse types of social networks and other data types being used for mining, knowledge discovery, querying, and analysis.

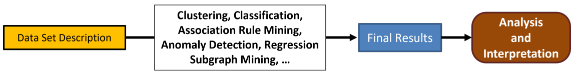

Data Set Description
Clustering, Classification, Association Rule Mining, Anomaly Detection, Regression Subgraph Mining, ...
Final Results
Analysis and Interpretation

Figure 1: Life Cycle Flow Chart of Mining.

In this paper, we focus on the applicability and versatility of graphs, especially Multilayer Networks (MLNs) for moving towards modeling and analysis of big data. In contrast to the mining approach shown in Figure 1, big data analysis needs to be addressed using a life cycle starting from modeling to drill-down and visualization. Currently, graph models are generated *manually* for a given data set without using any principled approach. For many data sets, both modeling and analysis computations are quite different from the ones addressed in earlier data models. In this paper, instead of generating a schema, application requirements and data are transformed into different types of graphs including MLNs. Moreover, an analysis may require graph computations, such as shortest path, substructure discovery, community, centrality (e.g., hubs), or their combination. Once the chosen data model is generated and the objectives are mapped into appropriate computations, any available package/algorithm can be used. Finally, the analysis results need to be drilled down and visualized in multiple ways for decision-making and for taking action. We present several results from the literature to convince the reader that this workflow is needed. **Figure 2** shows our view of the big data analysis life cycle from *gathered application requirements* to *analysis of objectives* to *result drill-down with visualization.* Only graph and MLN models are shown. This workflow is iterative.

Drill-down of analysis results is critical, especially for diverse data that has both structure and se-

Santra, A., Billah, H. and Chakravarthy, S.
Multilayer Networks: For Modeling and Analysis of Big Data.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 383-390
ISBN: 978-989-758-716-0; ISSN: 2184-3228

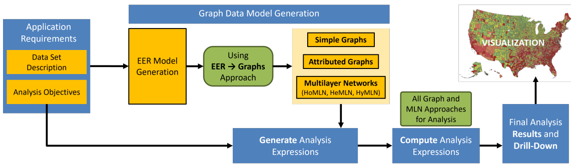

Graph Data Model Generation
Application Requirements
Data Set Description
Analysis Objectives
EER Model Generation
Using EER → Graphs Approach
Simple Graphs
Attributed Graphs
Multilayer Networks (HoMLN, HeMLN, HyMLN)
VISUALIZATION
All Graph and MLN Approaches for Analysis
Generate Analysis Expressions
Compute Analysis Expressions
Final Analysis Results and Drill-Down

Figure 2: Life Cycle of Big Data Modeling and Analysis using Graphs and MLNs.

mantics. For example, it is not sufficient to know the objects in a community, but additional object details are needed, similarly, for a centrality hub. For graph and MLN models, we also need to know the edges within and across layers, if any. From a computation/efficiency perspective, minimal information needs to be used for analysis whereas the drill-down phase needs to expand upon to the desired extent. Visualization is not new either and there exists a wide variety of tools for visualizing base data, results, and drilled-down information in multiple ways. Several data visualization platforms are available (GeP, 2014, Samant et al., 2021). Due to space constraints, we will not discuss drill-down and visualization in this paper. The contributions of this paper are:

- **Complete Life cycle** for big data analytics in comparison with mining
- **Graph and MLN** models, and analysis alternatives
- Use and applicability of **MLNs for complex data**
- **Graphs and MLNs** applicability for **video data** analysis
- **MLNs Applicability** for information integration/fusion

The rest of the paper is organized as follows. Relevant literature and different graph models are discussed in Sec. 2 and Sec. 3 respectively. The use of MLNs to model and analyze complex data is summarized in Sec. 4. The use of MLN in lieu of graphs is discussed in Sec. 5. Graphs and MLNs applicability to model and analyze video data is summarized in Sec. 6. Finally, the use and applicability of MLNs for information integration/fusion are discussed in Sec. 7. We conclude and outline future work in Sec. 8.

2 RELATED WORK

We discuss here how different phases of the lifecycle have been addressed in the literature.

EER Modeling: Since the 70s, *EER model* (Chen, 1976) has served as a methodology for database design, by representing data and functionality requirements of real-world applications in a precise manner by identifying entities, attributes, and relationships among them. However, with the emergence of data sets with multiple entity types and relationships along with complex analysis requirements, such as shortest paths, important neighborhoods, dominant nodes (or groups of nodes), etc., the relational data model was not adequate for modeling and analysis. Recently, there has been some work in modeling graphs from EER diagrams but is limited to simple and attributed graphs only (Roy-Hubara et al., 2017, Angles, 2018).

Graph and MLN Models: When a graph is used as a data model, the choice of nodes, edges, and their labels becomes important. There are multiple ways of creating them depending on the analysis objectives. Further, creating edges needs similarity/proximity criteria which need to be specified/identified. There needs to be a systematic and configurable approach for converting raw data sets (.csv files, extracted video contents, etc.) to graphs or MLN layers. Only recently, there has been some work (Komar et al., 2020, Santra et al., 2022) on extending the EER approach to generate MLN models.

Graph and MLN Analysis: There is substantial work in the area of simple, attributed graphs and MLNs. For simple graphs, many algorithms have been developed for shortest paths, spanning trees, community detection, centrality measures, and cliques. The breadth and depth-first approaches are also used for many algorithms. For attributed graphs, substructure discovery (Holder et al., 1994, Padmanabhan and Chakravarthy, 2009, Yan and Han, 2002) for interesting exact and inexact or similar substructures, and graph search and querying (Das et al., 2020) have been developed. For MLNs, algorithms have been developed for homogeneous (HoMLN) and heterogeneous (HeMLN) MLNs. Community detection algorithms have been extended to HoMLNs (review: (Kim and Lee, 2015, Magnani et al., 2021)). Further, methods have been developed to determine *centrality measures* to identify highly influential nodes (Solé-Ribalta et al., 2014, Zhan et al., 2015). Recently developed decoupling-based approaches combine partial analysis results from individual layers systematically *in a loss-less manner* to compute communities (Santra et al., 2017) or centrality hubs (Pavel et al., 2023) for layer combinations. Majority of HeMLN work (reviews in (Shi et al., 2017, Sun and Han, 2013)) focuses on developing meta-path based methods for object similarity, object classification, missing link prediction, ranking/co-ranking, and recommendations. Few existing works generate clusters of entities (Melamed, 2014). Most of them concentrate mainly on interlayer edges and not the networks themselves.

Graph Models for Video Analysis: Several custom approaches have been developed for modeling videos as scene graphs (Ji et al., 2020, Ou et al., 2022) by training deep learning algorithms and can perform fixed types of analysis (Billah et al., 2024). They need to be retrained or a new algorithm is required to perform a new type of analysis. Several frameworks are also available which models extracted video contents as attributed graphs (Yadav et al., 2020, Zhang et al., 2023) and perform analysis on them. However, they do not consider all the extracted video contents for modeling and only support simple analysis such as counting the number of objects. They cannot perform complex analyses (e.g., finding groups) on videos. Graphs and MLNs can be leveraged to model all the extracted video contents and new algorithms/operators need to be developed to perform interesting analysis on videos.

3 GRAPHS FOR BIG DATA ANALYSIS

Graphs capture relationships between entities in application data using nodes and edges. This representation allows us to perform various analyses based on the graph structure and relationships found in the data.

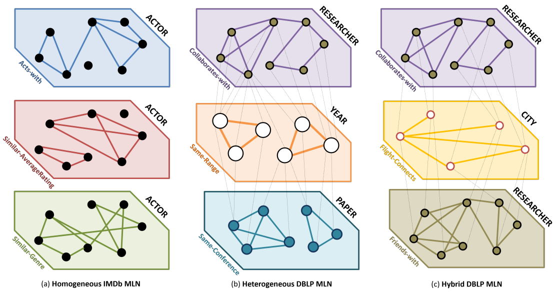
3.1 Graph Types Used as Data Models

A **simple graph** is defined as (V, E) where V is a set of vertices or nodes and E is a set of edges connecting two *distinct* vertices. E is a subset of V × V. The edges are assumed to be unweighted, either directed or undirected, and loops and multiple edges between nodes are not allowed. Typically, vertices have unique numbers, but labels of nodes and edges need not be unique. These graph models are widely used for modeling and analyzing applications.

An **attributed graph** (also called a multigraph) is defined as (V, E, ϕ) where V is a set of vertices or nodes, E is a set of edges connecting two distinct vertices, and ϕ is a function mapping of E to $\{\{x,y\} \mid x,y \in V \; and \; x \neq y\}$. If the distinctness of nodes is removed, loops will be allowed as well. The main advantage of a multigraph or attributed graph from a modeling viewpoint is that it captures multiple entities and multiple relationships between entities. Multiple labels can be associated with nodes and entities. With the attributed graph model, it is possible to include relevant information from the data description as labels and hence is more expressive as a model than a simple graph model.

An **MLN** is a *network of simple graphs* (or forests). In this model, every layer represents a distinct relationship among entities with respect to a single (or combination of) feature(s). The sets of entities across layers, which may or may not be of the same type, can be related to each other too.

An MLN can be used to separate entities and corresponding relationships from an attributed graph into separate layers where each layer is a simple graph. This provides more clarity in understanding and processing. MLNs are widely used for modeling complex data sets with multiple types of entities and multiple relationships between the same types of entities. They can also capture relationships between different types of entities.



(a) Homogeneous IMDb MLN (b) Heterogeneous DBLP MLN (c) Hybrid DBLP MLN

Figure 3: Multilayer Network Types.

Based on the type of relationships and entities, MLNs can be classified into three types. Layers of a **homogeneous MLN (HoMLN)** are used to model different relationships among the **same entity types** like movie actors who are linked based on co-acting (i.e., they act together in a movie) or have similar average rating or have worked in similar genres (Figure 3(a)). Thus, $V_1 = V_2 = \ldots = V_n$ and inter-layer edge sets are empty as no relations across layers are necessary. Relationships among **different types of entities** like researchers (connected by co-authorship), research papers (connected if published in the same conference), and year (related by predefined ranges/eras) are modeled through **heterogeneous MLN (HeMLN)** (Figure 3(b)). The inter-layer edges represent the relationship across layers like writes, published-in, and active-in. In addition to being collaborators, researchers may be social media friends. Thus, to model multi-feature data that capture **multiple relationships within and across different types of entity sets**, a combination of homogeneous and heterogeneous MLNs is used, termed **hybrid MLN (HyMLN)**, as shown in Figure 3(c). Here, the first and the third layer have the same node types (researchers) linked to the city nodes they reside in, which are in turn connected based on the flight network (second layer).

The above graph types and MLN variants provide alternatives for matching modeling and analysis needed for application data. Further, MLNs provide clarity in understanding the data set. Additionally, the availability of algorithms for a specific graph model also plays a key role in the choice of the graph model. For instance, there are not many algorithms available for attributed graphs in contrast to simple graphs. There is considerable ongoing research in developing algorithms for the MLNs (Boden et al., 2012, Santra et al., 2017) due to the clarity of the model. Hence, MLNs are preferred for modeling complex data sets.

3.2 EER Modeling Extensions

In contrast to the relational data model, a principled approach to convert application requirements into a chosen graph model (simple, attributed, or MLN) is lacking. However, recently there has been some work in this regard (Komar et al., 2020, Santra et al., 2022) leading to the wider use of MLNs. Broadly, the entities in the EER diagram dictate the formation of layers with the entity instances as layer nodes and the binary self relationship defining the intra-layer edges. The binary non-self relationships define the inter-layer edges. Some relationships are self-explanatory and can be easily mapped into edges like friendships, siblings, direct flights, and so on. However, some relationships are non-explicit like “two actors working in *similar* genre of movies” for which the EER model needs to have a *parameter* attribute for the relationship that defines the similarity metric and threshold. The value of these parameter attributes will be used to generate the edges in the MLN. Currently, we are developing algorithms for converting EER to any type of graph, not just MLN. More research is needed in this area to make analysis easier.

4 MLNs FOR BIG DATA ANALYSIS

Depending on the analysis requirements, the Google Knowledge Base (GKB) data set can be modeled as different types of MLNs. For instance, there exist multiple relationships among the same set of people - whether they are married to each other or have the same birth state or studied in the same university, and so on. This gives rise to a homogeneous GKB MLN with the same set of nodes being connected differently in each layer (Figure 4(a)). Similarly, Figure 4(b) shows an HeMLN where both layers have different sets of entities - person, and company.

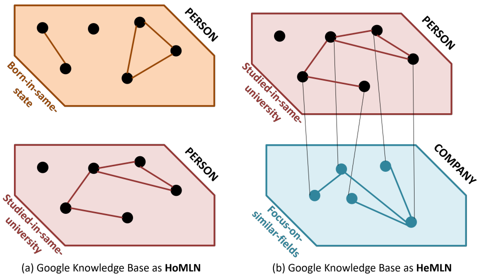

PERSON
Born-in-same-state
PERSON
Studied-in-same-university
PERSON
Studied-in-same-university
COMPANY
Focus-on-similar-fields
(a) Google Knowledge Base as HoMLN
(b) Google Knowledge Base as HeMLN

Figure 4: Google Knowledge Base modeled as MLNs.

The person nodes are connected if they studied in the same university, the company nodes are connected if they focus on similar fields, and the person nodes are connected to the company nodes that they founded/established through inter-layer edges. This may also be extended to Hybrid MLNs if two different person layers are connected to a company layer.

4.1 MLN: Multiple Analysis Choices

Figure 5 shows three MLN analysis alternatives. Figure 5(a) shows an MLN conflated into a simple graph by aggregating layers. These aggregation approaches, termed type-independent (Domenico et al., 2014) and projection-based (Berenstein et al., 2016), ignore type information. Hence, they do not support structure and semantics preservation without elaborate mappings as they aggregate or collapse layers into a simple graph in different ways. As observed in the literature, *without additional mappings*, currently-used aggregation approaches are likely to result in some information loss, distortion of properties, or hide the effect of different entity types and/or different intra- or inter-layer relationships (Kivelä et al., 2013, De Domenico et al., 2014). At the other end of the spectrum, Figure 5(c) shows the same MLN layers and result computation by traversing the MLN as is.

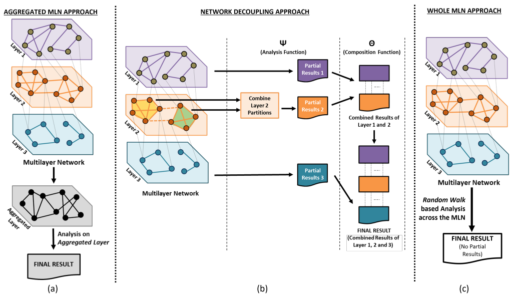


Figure 5: (a) Lossy Vs. (b) Decoupling Vs. (c) Whole MLN approaches (Santra et al., 2022).

Figure 5(b) on the other hand proposes an approach, termed **networking decoupling**, where network property for each layer is computed independently (possibly in parallel) in the analysis (Ψ) phase and compose them using a binary operator Θ. This approach has been shown to be effective and can be done using Boolean operations for HoMLNs and HeMLNs without losing type information. Furthermore, it is more efficient than the approaches shown in Figure 5(a) or (c). Finally, the clarity of modeling using MLNs is retained as well.

5 USE OF MLNs IN LIEU OF GRAPHS

Based on daily life interactions (education, social media platforms, restaurant check-ins, healthcare check-up appointments, etc.) different facts are available on the web. In terms of knowledge base, “different facts” about a person are captured in the GKB. Freebase captures such information for famous personalities: birth place and residence, education institutions attended, birth and death date (if available), companies worked in/founded, family-based relationships and so on (Bollacker et al., 2008). Here, people, universities, companies, and states are related to each other based on explicitly available interactions or relationships. Some interesting analysis objectives can be:

- *(GKB-O1)* Find frequently occurring patterns among states, based on university locations and place of company headquarters for the entrepreneurs.
- *(GKB-O2)* Find groups of people who were born in the same state and have studied in the same university.
- *(GKB-O3)* For each group of founders who have studied in the same university, find out the most popular focus field among the group of similar companies that they have founded.

Although objective *(GKB-O1)* can be computed using traditional graph models, MLNs are needed for objective *(GKB-O2)* and others similar to that. The HoMLN shown in Figure 4(a) is required to address *(GKB-O2)*. In this case, we need to “Find **groups** of people who were *born in the same state* **and** have *studied in the same university*”. Here “grouping” keyword means that we need to compute communities among the people nodes, followed by AND composition (due to the “and” keyword). For AND composition, here the CE-AND composition algorithm is used that intersects the community edges, then perform a connected component analysis to obtain the group of nodes that are tightly connected in both the layers (Santra et al., 2022, Santra et al., 2017). Thus, the analysis expression based on the decoupling approach can be expressed as:

```
Expression: Ψ(PERSON-Born-in-same-state) Θ
Ψ(PERSON-Studied-in-same-university);
where Ψ = Community; Θ = CE-AND (composition)
```

Similarly, for *(GKB-O3)*, the HeMLN shown in Figure 4(b) is used. Here, “For each **group** of founders who have *studied in the same university*, we need to find out the **most popular** focus *field* among the **group** of *similar companies* that they have founded.” Thus, communities need to be detected in both person and company layers, which become meta-nodes in the bipartite graph. The number of inter-layer edges between the constituent nodes of each pair of meta nodes will define the edge weight. Finally, maximal weighted matching (MWM) will give us the required optimal pairing of person and company communities (Santra et al., 2022). The analysis expression is as follows:

```
Expression: Ψ(PERSON-Studied-in-same-univer- sity)
Θ Ψ(COMPANY-Focus-on-similar-fields);
where Ψ = Community; Θ = MWM (bipartite maximum
weighted matching)
```

6 GRAPHS/MLNs FOR VIDEO ANALYSIS

Our goal, as part of big data analysis, is to handle different data types (4 Vs of big data) in the same way we handle structured and tabular data. If videos (or extracted contents) can be modeled using graphs/MLNs, the same life cycle approach can be applied for video analysis, enabling the modeling and analysis of video data alongside other data types. As discussed in Sec. 2, the existing custom approaches for video analysis require new software/algorithm/retraining to perform a new analysis. Hence, this approach does not lend itself to the holistic analysis required for big data. In contrast, if big data analysis were to include video analysis in mainstream data processing, a different approach would be needed.

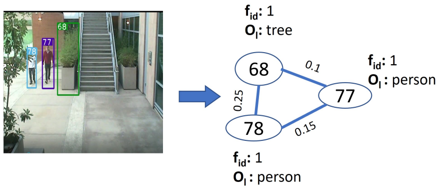
Some works in the literature used graphs for video analysis as explained in Sec. 2. Recently, (Billah et al., 2024) proposed a novel approach for video analysis that has the potential to advance big data analysis to include videos. This approach is novel as video contents are extracted once (using existing Video Content Extraction (VCE) algorithms), modeled, and then analyzed to identify a variety of situations from them. This approach has several advantages: i) video contents are **extracted only once**, ii) it is possible to model these extracted contents completely, iii) several analysis expressions can be formulated and computed on them, iv) both "ad hoc" and "what if" analysis can be supported, and v) most importantly, this can be extended for **real-time analysis**.

A workflow of open-source VCE algorithms can be used for extracting object bounding boxes and class labels (with a confidence score) using object detection (YOLO (Wang et al., 2024)) algorithm, unique identifier (object_id) for each object and feature vectors using object tracking (Bot-sort (Aharon et al., 2022)) algorithm, and pose coordinates using pose estimation (HRNet (Wang et al., 2020)) algorithm.

Modeling of Extracted Video Contents: The different types of extracted video contents can be modeled in multiple ways. Two promising models that are being explored in the literature are the extended relational model (Billah and Chakravarthy, 2024) and the graph model (Billah et al., 2024). If it is modeled using an extended relational model, Continuous Query Language (CQL) (an extension of the widely-used Structured Query Language (SQL)) can be used. If the extracted contents are modeled as graphs, different graph analysis techniques can be used. The rationale for using multiple models is that some analysis may be easier in one model as compared to the other. For example, clustering of objects is easier using the graph model than the extended relational model. We will focus on the graph model as this paper is about the utility of graphs and MLNs for big data analysis.

To represent extracted video contents as graphs, nodes and edges need to be identified and other related information (e.g., the feature vectors, bounding boxes, etc.) needs to be associated properly for computation. Many analyses involve objects. Hence, objects are represented as nodes and *Object_id* as node id in the literature. There are multiple choices to create edges (e.g., the distance between objects, and their spatial relationship in a frame, etc.). Figure 6 shows a graph representation of a sample video frame with nodes with two labels: frame id (f_{id}) and object class label (O_l) and edges (based on the objects bounding box centroid distance).

A spectrum of alternatives exists for the graph representation, each with different advantages and disadvantages. It is possible to model the *entire video as one graph* (model M_1) using object_id for nodes (with a large amount of information with each node). It is also possible to create *a graph for each frame* (model M_F) (shown in Figure 6), where the number of graphs will be equal to the number of non-empty frames F in the video. Options in-between are also possible where a forest of g ($1 \leq g \leq F$) graphs (model M_g) can be generated by aggregating the consecutive frames into a graph based on some constraints, with varying numbers of graphs for different videos. The in-between alternatives allow us to compress node labels and edges in different ways reducing the storage required and can also reduce computational complexity as the graphs are generated in some logical manner.



(a) Sample video frame with f_{id} = 1

(b) Graph Representation of a video frame

Figure 6: Graph representation of a sample video frame.

Video Content Analysis Using the Graph Models: Below, we indicate video analysis examples using graph models from the literature.

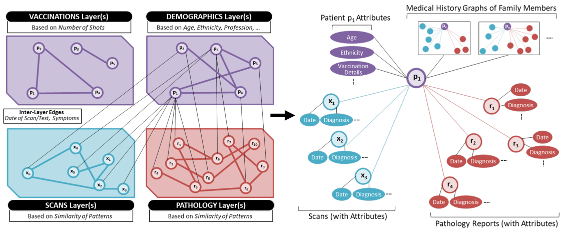
1. **Identifying Groups (Billah et al., 2024):** In assisted living environment videos, it is useful to identify isolated individuals (not participating in group discussions, etc.) This analysis has been reported in (Billah et al., 2024) to cluster individuals in video frames by leveraging K-Means clustering on model M_F where nodes are objects and edges are the object bounding box centroid distances.
2. **Identifying if a Parking Slot is Occupied (Yadav et al., 2020):** In surveillance videos, it is often important to know which parking spaces are occupied. This analysis has been reported in (Yadav et al., 2020) using model M_F, where nodes are objects and edges are spatial bounding box relationships (e.g., overlap, inside, etc.) between objects in a frame. Their proposed algorithm identifies a parking lot as occupied if the parking lot and a car's bounding box overlap over a threshold.

In summary, extracted video contents are shown to be modeled and analyzed using alternative graph models and analysis algorithms. MLNs come in handy to model multiple graphs (or videos) as different layers and perform combined analysis. HoMLNs can be used by connecting object_ids from different graphs generated from the same video or by connecting object_ids from different videos if their feature vectors match. Once modeled appropriately, interesting analysis (e.g., groups of objects entering and exiting a premise after n minutes of each other) can be performed using graphs/MLNs.

7 MLNs FOR INFORMATION FUSION

Analysis of a *single modality/data type* has been the major focus until now, be it structured (e.g., stream data processing (Barbieri et al., 2010)) or unstructured data (e.g., image and video analysis (Zhang et al., 2023, Yadav et al., 2020), text and natural language processing (Otter et al., 2020)). Yet, when all or a subset of these data types must be analyzed holistically, several challenges emerge. These problems have been categorized under various headings, such as data fusion, multi-modal data analysis, and others which are limited in scope and context (Atrey et al., 2010). The challenges originate due to the lack of approaches that can effectively perform information fusion both at the modeling and analysis stages. Therefore, the holistic approach needs to accommodate modeling, and analysis techniques for objectives for performing knowledge discovery. In our view, MLNs with their modeling and analysis advantages provide a path to explore information fusion. Many applications, such as cybersecurity, healthcare, and surveillance can benefit from this. We illustrate this with an example.

Sample Application – Healthcare: Patient data is collected in diverse formats by different specialists over time. This data constitutes the patient's medical records including demographics, hospital/doctor visits, vital signs, medications, progress notes, allergies, radiology images, and laboratory results, and can be further enriched by exercise data, etc. This data is both spatial and temporal. When all this data is accumulated, holistic knowledge discovery over an individual and the population is possible. This application with big data characteristics can be used for personalized care using querying, searching, and mining. MLNs can be used for effectively modeling this data and for flexible analysis. Layers that can be identified are: i) **Demographics Layer(s)**: Patient nodes are connected by edges based on demographics (age, ethnicity, profession, education level, etc.), ii) **Image/Video Layer(s):** Patient nodes are connected based on the similarity of patterns present in them (X-rays, MRIs, EKG, and CT Scans), iii) **Pathology Layer(s):** Patient nodes are connected based on the similarity of indicators (e.g., high sugar, high/low BP, etc.), iv) **Vaccination Layer(s):** Person nodes are connected based on the number of doses and type of shots. These layers can be generated for county/city/state as needed.



(a) Hybrid Healthcare Multilayer Network (b) Extracted Medical History Graph of Patient p_1

Figure 7: Healthcare MLN.

Figure 7(a) illustrates 4 possible layers of the hybrid healthcare MLN, with the inter-layer edges. For example, the demographics layer can be linked with scan/pathology layers based on whom the report belongs to with the test report date and symptoms as the label information. From this MLN, it is also possible to extract graphs for an individual or a select group for different types of analysis (shown in Figure 7(b) for patient p_1 and his/her family). This model with the extracted graph(s) allows us to query, search, and analyze to discover knowledge using all or a subset of layers in various ways. Few examples are - using collective information of an individual and family, a physician can draw holistic inferences which may not be possible without a model that represents multi-source, multi-type data (**personalized/customized holistic diagnosis/inference**), find group(S) of people for a specific demographics who had lung problems and other co-morbidity (e.g. diabetes) and contracted Covid (**aggregate analysis using homogeneous and heterogeneous community detection on multiple layers**), people who did not have any history of lung issues but contracted Covid (**mining on a subset of layers using Boolean NOT operation**).

8 CONCLUSIONS

In this position paper, we argue for MLNs as **a** viable alternative for big data analytics. We have discussed the versatility of MLN models and their ability to model diverse data, the recent work on MLN model generation using the EER approach, and efficient MLN algorithm development for analysis. Based on MLN work in the literature, we have argued for their use for modeling and analyzing complex data sets including images, videos, and other data types (e.g., natural language). There is an ongoing effort to apply MLNs for information fusion/integration as well.

ACKNOWLEDGMENTS

This work was supported by NSF awards #1955798 and #1916084.

REFERENCES

(2014). Gephi - The Open Graph Viz Platform . http://gephi.org/.

Aharon, N., Orfaig, R., and Bobrovsky, B. (2022). Bot-sort: Robust associations multi-pedestrian tracking. *CoRR*, abs/2206.14651.

Angles, R. (2018). The property graph database model. In *AMW*.

Atrey, P. K., Hossain, M. A., El Saddik, A., and Kankanhalli, M. S. (2010). Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16(6):345–379.

Barbieri, D. F., Braga, D., Ceri, S., VALLE, E. D., and Grossniklaus, M. (2010). C-sparql: a continuous query language for rdf data streams. *International Journal of Semantic Computing*, 4(01):3–25.

Berenstein, A., Magarinos, M. P., Chernomoretz, A., and Aguero, F. (2016). A multilayer network approach for guiding drug repositioning in neglected diseases. *PLOS*.

Billah, H. and Chakravarthy, S. (2024). Video situation monitoring using continuous queries. In *DEXA,2024*, volume 14911 of *LNCS*, pages 125–141. Springer.

Billah, H., Santra, A., and Chakravarthy, S. (2024). Leveraging video situation monitoring in assisted living environment. In *PETRA, 2024*, pages 307–315. ACM.

Boden, B., Günnemann, S., Hoffmann, H., and Seidl, T. (2012). Mining coherent subgraphs in multi-layer graphs with edge labels. KDD '12, pages 1258–1266.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Chen, P. P.-S. (1976). The entity-relationship model—toward a unified view of data. *ACM transactions on database systems (TODS)*, 1(1):9–36.

Das, S., Santra, A., Bodra, J., and Chakravarthy, S. (2020). Query processing on large graphs: Approaches to scalability and response time trade offs. *Data Knowl. Eng.*, 126:101736.

De Domenico, M., Solé-Ribalta, A., Gómez, S., and Arenas, A. (2014). Navigability of interconnected networks under random failures. *Proc. of Ntl. Acad. of Sciences*.

Domenico, M. D., Nicosia, V., Arenas, A., and Latora, V. (2014). Layer aggregation and reducibility of multilayer interconnected networks. *CoRR*, abs/1405.0425.

Holder, L. B., Cook, D. J., and Djoko, S. (1994). Substucture Discovery in the SUBDUE System. In *Knowledge Discovery and Data Mining*, pages 169–180.

Ji, J., Krishna, R., Fei-Fei, L., and Niebles, J. C. (2020). Action genome: Actions as compositions of spatio-temporal scene graphs. In *CVPR*, pages 10236–10247.

Kim, J. and Lee, J. (2015). Community detection in multi-layer graphs: A survey. *SIGMOD Record*, 44(3):37–48.

Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., and Porter, M. A. (2013). Multilayer networks. *CoRR*, abs/1309.7233.

Komar, K. S., Santra, A., Bhowmick, S., and Chakravarthy, S. (2020). Eer→mln: EER approach for modeling, mapping, and analyzing complex data using multi-layer networks (mlns). In *ER 2020*, pages 555–572.

Magnani, M., Hanteer, O., Interdonato, R., Rossi, L., and Tagarelli, A. (2021). Community detection in multiplex networks. *ACM CS.*, 54(3):48:1–48:35.

Melamed, D. (2014). Community structures in bipartite networks: A dual-projection approach. *PloS one*, 9(5):e97823.

Otter, D. W., Medina, J. R., and Kalita, J. K. (2020). A survey of the usages of deep learning for natural language processing. *TNNLS*, 32(2):604–624.

Ou, Y., Mi, L., and Chen, Z. (2022). Object-Relation Reasoning Graph for Action Recognition. In *CVPR*, pages 20133–20142.

Padmanabhan, S. and Chakravarthy, S. (2009). HDB-Subdue: A Scalable Approach to Graph Mining. In *DaWaK*, pages 325–338.

Pavel, H. R., Roy, A., Santra, A., and Chakravarthy, S. (2023). Closeness centrality detection in homogeneous multilayer networks. In *IC3K 2023, KDIR*.

Roy-Hubara, N., Rokach, L., Shapira, B., and Shoval, P. (2017). Modeling graph database schema. *IT Professional*, 19(6):34–43.

Samant, K., Memeti, E., Santra, A., Karim, E., and Chakravarthy, S. (2021). Cowiz: Interactive covid-19 visualization based on multilayer network analysis. In *ICDE 2021*, pages 2665–2668. IEEE.

Santra, A., Bhowmick, S., and Chakravarthy, S. (2017). Efficient community re-creation in multilayer networks using boolean operations. In *ICCS 2017*, pages 58–67.

Santra, A., Komar, K., Bhowmick, S., and Chakravarthy, S. (2022). From base data to knowledge discovery–a life cycle approach–using multilayer networks. *DKE*, 141:102058.

Shi, C., Li, Y., Zhang, J., Sun, Y., and Philip, S. Y. (2017). A survey of heterogeneous information network analysis. *IEEE Trans. Knowl. Data Eng.*, 29(1):17–37.

Solé-Ribalta, A., De Domenico, M., Gómez, S., and Arenas, A. (2014). Centrality rankings in multiplex networks. In *Procds. of 2014 ACM conf. on Web science*, pages 149–155. ACM.

Sun, Y. and Han, J. (2013). Mining heterogeneous information networks: a structural analysis approach. *ACM SIGKDD Explorations Newsletter*, 14(2):20–28.

Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., and Ding, G. (2024). Yolov10: Real-time end-to-end object detection. *CoRR*, abs/2405.14458.

Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al. (2020). Deep high-resolution representation learning for visual recognition. *PAMI*, 43(10):3349–3364.

Yadav, P., Salwala, D., Das, D. P., and Curry, E. (2020). Knowledge Graph Driven Approach to Represent Video Streams for Spatiotemporal Event Pattern Matching in Complex Event Processing. *IJSC*, 14(03):423–455.

Yan, X. and Han, J. (2002). gSpan: Graph-Based Substructure Pattern Mining. In *IEEE International Conference on Data Mining*, pages 721–724.

Zhan, Q., Zhang, J., Wang, S., Philip, S. Y., and Xie, J. (2015). Influence maximization across partially aligned heterogenous social networks. In *PAKDD (1)*, pages 58–69.

Zhang, E., Daum, M., He, D., Haynes, B., Krishna, R., and Balazinska, M. (2023). Equi-vocal: Synthesizing queries for compositional video events from limited user interactions. *VLDB*, 16(11):2714–2727.

Route Recommendation Based on POIs and Public Transportation

Ágata Palma[1][a], Pedro Morais[1][b] and Ana Alves[1,2][c]

[1]*Polytechnic University of Coimbra, Rua da Misericórdia, Lagar dos Cortiços, S. Martinho do Bispo, 3045-093 Coimbra, Portugal*

[2]*CISUC, LASI, University of Coimbra, Polo II, Pinhal de Marrocos, 3030-290, Coimbra, Portugal*

{*a2023113935, a21280686, aalves*}*@isec.pt*

Keywords: GIS, Information Retrieval, Ambient Intelligence, Clustering, Route Recommendation, POIs.

Abstract: With the rapid advancement of technology in today's interconnected world, Ambient Intelligence (AmI) emerges as a powerful tool that revolutionizes how we interact with our environments. This article delves into the integration of AmI principles, Python programming, and Geographic Information Systems (GIS) to develop intelligent route recommendation systems for urban exploration. The motivation behind this study lies in the potential of AmI to address challenges in urban navigation, personalized recommendations, and sustainable transportation solutions. The objectives include optimizing travel routes, promoting sustainable transportation options, and enhancing user experiences. This research will contribute to advancing AmI technologies and their practical applications in improving urban living standards and mobility solutions.

1 INTRODUCTION

Ambient Intelligence (AmI) represents a new paradigm in computing that aims to embed intelligence into everyday environments. It involves the integration of computational capabilities into ordinary objects, allowing them to interact with users and each other in a natural and intelligent manner. AmI emphasizes the presence of humans alongside smart interfaces that can adapt to human emotions, behaviors, and expectations. This concept envisions the creation of smart environments, such as smart homes, smart healthcare facilities, and smart cities, where everyday objects are seamlessly connected and capable of enhancing daily living experiences. AmI is seen as a significant societal and cultural shift, with the potential to transform the way people interact with technology and their surroundings (Thankachan, 2023).

Since AmI takes advantage of sensors and Internet of Things (IoT) devices to gather information about the surrounding environment, it is a useful tool to make inferences based on proximity, intent, and behavioral patterns. This facilitates personalized experiences, as for example, receiving location-based alerts when reaching points of interest (POIs) in a new city(Mahmood et al., 2023).

Ambient Intelligence enhances the accuracy and relevance of environmental data by incorporating Geographic Information Systems (GIS) and Information Retrieval techniques. GIS offers spatial analysis and mapping to understand user interactions geographically, while Information Retrieval efficiently extracts relevant data for context-aware recommendations. Clustering techniques group similar data points to identify patterns in user behavior, aiding route recommendation systems by predicting optimal paths based on historical data and preferences. These technologies enable AmI to create intelligent environments that anticipate and respond to user needs, providing seamless and enriched interactions.

The motivation behind this study lies in the practical application of advanced geospatial technologies and algorithms to enhance urban navigation. The aim is to develop an intelligent system that can provide efficient, customizable routing solutions tailored to individual preferences, particularly in urban environments where efficient navigation and personalized experiences are crucial to navigate busy zones and discover POIs. Therefore, these intelligent systems can offer context-aware recommendations and optimize routes tailored to user preferences and sustainable transport options, ultimately promoting efficient travel and contributing to sustainable urban mobility.

[a] https://orcid.org/0009-0009-4450-700X

[b] https://orcid.org/0009-0003-5962-0386

[c] https://orcid.org/0000-0002-3692-338X

Palma, Á., Morais, P. and Alves, A.
Route Recommendation Based on POIs and Public Transportation.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 391-398
ISBN: 978-989-758-716-0; ISSN: 2184-3228

This work explores the integration of AmI principles, Python programming, and GIS to develop intelligent route recommendation systems for urban exploration based on POIs and available public transportation. Given a city and, optionally, a category, the system will reply with a list of POIs and a suggested route to visit the largest number of POIs in the shortest route possible using public transportation. The base of this work is a variant of the Travelling Salesman Problem (TSP), which involves finding the shortest possible route that visits a set of given locations exactly once and then returns to the starting point (Özcan and Kaya, 2018). The challenge in this research is similar to the one on TSP: determine the most efficient route between multiple POIs while minimizing the total travel distance.

The structure of the article includes a review of related works and AmI principles and their relevance in urban navigation, followed by a discussion of technical aspects such as data integration, route optimization algorithms, and visualization techniques using platforms like Quantum GIS (QGIS).

Practical implications, potential extensions, and the broader impact of AmI-driven solutions on urban mobility and city planning are also addressed in the discussion and conclusions sections. This work aims to contribute to the advancement of technology that enhances user experiences and promotes sustainable and efficient mobility solutions in urban settings.

2 RELATED WORK

In this section, a brief literature review is presented, focusing on existing applications, systems, and studies that share similar objectives or themes related to AmI and intelligent route recommendation systems for urban exploration based on POIs.

Based on the TSP, (Özcan and Kaya, 2018) aimed to create a new tourist guide app using OpenStreetMap (OSM). To achieve this, the study involved various tasks using OSM tools, libraries, and frameworks. These tasks included real-time area drawing on OSM, path computation, selection of POIs, and map understanding. The app intends to determine the shortest route between user-selected destinations, optimizing travel time and displaying the route visually on the map. The Hill Climbing Algorithm (HCA), known for its memory efficiency and local search approach, was used for the TSP.

On the itinerary recommendation variant, (Panagiotakis et al., 2022) proposed a method to personalize itinerary recommendation (PIR) with POIs categories, for tourists tours. The authors' method was based on the Expectation Maximization (EM) algorithm, and solves, sequentially, the PIR problem by selecting POIs that maximize a suitable objective function, such as user satisfaction, user time budget, POIs opening hours, POIs category and spatial constraints. In a similar scope, (Lou, 2022) focused on categorizing POIs but with an improved k-means algorithm to be applied to intelligent tourism route planning. The proposed scheme considers tourists' preferences and aims to find the shortest route between desired locations within a selected area.

(Mahdi et al., 2023) also redirected their research focus towards POIs. They applied regression models to analyze the data obtained from Google Popular Times (GPT) to predict the amount of time people would spend at POIs. With this contribution, a similar process would be possible to improve the route generation plan when time constraints are a variable.

Besides the prediction of the time spent at a POI, when planning a route based on public transportation, it is also crucial to take into account the time spent from one point to another. (Zhang et al., 2022) state the importance of improving travel time prediction. The study highlights the importance of real-time, accurate, reliable and low-cost multi-source data for better predictions. The authors affirm that the traditional methods for predicting travel time are deficient and a new approach based on intelligent technology would improve the prediction accuracy. In order to accomplish this, a prediction model based on the Kalman filter - high accuracy in one-step prediction - was designed. For this model, two sub-modules were created: the Route Travel Time Prediction Model - predicts travel time for an entire bus route - and the Stop Dwell Time Prediction - predicts the time spent at bus stops. In this study, the data sources used included GPS (Global Positioning System), AFC (Automatic Fare Collection), and IC (Integrated Circuit) and the models were validated using Automatic Vehicle Location (AVL) from real world scenarios. The results indicate the prediction model meets accuracy requirements for travel time prediction.

(Sarridis et al., 2022) proposed a personalized route recommendation system that balances the tradeoff between distance and POIS using hypergraph models. Their framework considers tourist satisfaction and leverages both visual and geographical data to optimize the shortest path algorithm through POI images embedded in a hypergraph. Similarly, (Karantaidis et al., 2021) applied multi-stage optimization learning in hypergraph structures for image and tag recommendations, dynamically updating hypergraph structures and hyperedge weights to achieve higher accuracy in POI ranking and recommendations.

The contribution of (Li et al., 2021) to this work is based on a solution for another problem. Instead of the common questions such as “find the k nearest POIs around me” or “give me the bus plan from *s* to *d*”, the authors proposed a method to answer the “give me *k* POIs that I can reach earliest within one transfer by bus”. In the public transportation network (PTN), the users’ primary concern is “which POI can be reached with the least travel time under some specified transfer numbers considering the different departure time and frequency of buses”. To answer the proposed question, the k-nearest neighbor (kNN) query should be applied. Given a set of information—POIs, a PTN, a location, departure time, and a transfer number constraint—the kNN query returns the k POIs that meet these conditions.

Another possible method for location-based systems, besides kNN, is the Multi-Cost Transportation Network-constrained skyline query (MCTN-CSQ). (Gong et al., 2020) implemented the CSQ System, the first of its kind, as a web application supporting constrained skyline query on multi-cost transportation networks. Users input query points and receive skyline answer-objects reachable via transportation networks, superior on at least one dimension. For example, lets assume a user needs to book a room for the night but he has more constraints about the desired room: it can be reached by taking public transportation, and the transportation fare and the travel time should be reasonable - the query processing component of the CSQ System handles the query execution. “The system is implemented as a web application, which allows users to input a query point from a web interface, get the skyline result by using several algorithms, and display the result on the web interface” (Gong et al., 2020).

3 SYSTEM ARCHITECTURE

The application is designed to provide a comprehensive solution for route planning and analysis within the QGIS environment. The system architecture is composed of several elements, interconnected to facilitate preprocessing, route generation, spatial analysis, visualization, and user interaction.

The user interface, implemented using the QGIS interface in the first phase and a plugin in QGIS in the second phase, serves as the entry point for users to select the city and visualize the suggested route, as well as to specify POI categories. The route generation engine employs an algorithm to compute the optimal route, connecting different POIs based on user-defined parameters. The aim is to find the shortest path in the road network integrating public transport routes and stops.

In the first phase, a proof of concept is achieved by working with the available processing tools and plugins in QGIS, such as ORS (OpenRouteService) tools. The system uses data provided for the development of this project, specifically POIs and roads in Portugal and public transportation in Coimbra. The spatial visualization is handled by the QGIS environment, taking advantage of HeatMaps and route overlays to visually present analysis results, as well as to produce a georeferenced PDF.

For the second phase, a custom plugin is developed to provide the user with a more friendly and intuitive interface. Using the user’s input for a location and category of POIs, and the processing of POIs with machine learning methods, a route is drawn using the shortest path possible across the region with the most of these POIs.

4 DATA SOURCES

To populate the application with pertinent data concerning POIs, the primary source relies on OpenStreetMap for the second step, retrieved through the “osmnx” python package.

For the first step, data containing POIs and roads of Portugal in shapefiles format were provided in the context of this project, as well as public transportation data for Coimbra, with routes and stops for SMTUC (Serviços Municipalizados de Transportes Urbanos de Coimbra). Additionally, the application integrates (1) the QGIS plugin QuickOSM to retrieve the boundaries of Coimbra city and (2) the module Quick Map Services (QMS) to procure a standardized raster layer of OSM.

5 MACHINE LEARNING

To enhance the performance of the application, Clustering is applied, which allows the identification of groups of POIs that are geographically close to each other. Through this unsupervised learning method, the most concentrated area of POIs is identified and a route is established within that zone. A density-based cluster analysis algorithm, the Density-Based Spatial Clustering of Applications with Noise (DBSCAN), is applied due to its robustness and effectiveness in handling spatial data. As noted in (Lou, 2022), DBSCAN has the great advantage of clustering dense datasets of any shape and is “sensitive to the selection of initial values, but insensitive to noise points and has certain noise immunity". Another advantage is the unnecessary need to predefine the number of clusters. The initial DBSCAN parameters are:

- **eps** (ε)**.** The maximum distance between two points to be considered as part of the same neighborhood. This parameter defines the radius of the neighborhood around each point.
- **minPts.** The minimum number of points required to form a dense region. A point is considered a core point if it has at least *minPts* within its *eps* radius.

These parameters are crucial for the performance of the DBSCAN algorithm. In this study, *eps* and *minPts* are fine-tuned based on the spatial distribution of POIs in the dataset.

The clustering process entails loading the road network graph using *OSMnx*, projecting the POIs to align with the Coordinate Reference System (CRS) of the graph, and generating a distance matrix based on network distances. Subsequently, the DBSCAN algorithm is employed to detect clusters, and the outcomes are assessed using metrics like Silhouette Score and Davies-Bouldin Index.

The Silhouette Score measures how similar a point is to its own cluster compared to other clusters. Higher values indicate well-defined clusters with clear separation between them. The Davies-Bouldin Index assesses the average similarity ratio of each cluster with its most similar cluster, where lower values indicate better-defined clusters with less overlap. These metrics provide a quantitative evaluation of the clustering quality, ensuring that the clusters formed are meaningful and accurate.

6 VISUALISATION OF DATA

Throughout the first and second phases, different approaches are utilized for collecting, treating, and displaying the results. Both approaches are addressed, demonstrating the evolution and refinement of the methods to achieve a more automated response in custom route generation.

6.1 Phase I

Taking advantage of the already present module in QGIS, QMS, the standardized raster layer is retrieved, providing a comprehensive and detailed map background in *EPSG:4326*. This CRS, also known as *WGS 84*, is widely used in geographic coordinate systems and is the one that the ORS API expects in the requests.

To commence data analysis, the shapefiles of Portugal's POIs and roads are imported, along with the SMTUC General Transit Feed Specification (GTFS) containing route information and bus stops, facilitated by the GTFS GO plugin. Furthermore, the polygon delineating the region of Coimbra is imported using the QuickOSM plugin. Additionally, a new polygon is drawn within the Coimbra region to delimit the analysis area.

A new layer, named *Coimbra_POIS* is created through the extraction by location of elements that intersect or are contained within the area of the polygon. This layer is subsequently utilized as the foundation for generating a HeatMap, providing a visual representation of the concentration of POIs within the delimited area. Since meters are preferred over degrees for measurements, the layers are re-projected to *EPSG:3763*. This adjustment enables the proper configuration of parameters for the DBSCAN algorithm (ε: 200 meters; minPts: 4), using the *Coimbra_POIS* layer as the data source. The outcomes demonstrate a clear separation of clusters, indicating a satisfactory fit, as shown in Figure 1. To enhance visualization and delineate cluster regions more distinctly, concave hulls are employed for each cluster. A concave hull is a shape that closely wraps a set of points, capturing the boundaries of the points more accurately. The result provided a collection of polygons encompassing the points within each cluster, as depicted in Figure 2.

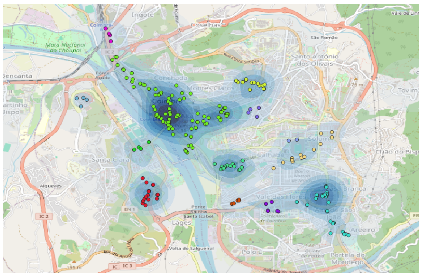


Figure 1: Heatmap with DBSCAN clustered POIs.

In this phase, the simulation involves a user who wishes to travel from point A to point B, as depicted in Figure 2, utilizing the shortest path and public transportation services. With this goal in mind, a manual approach is adopted for route construction. Using the ORS tools, a few points are manually selected as coordinates to create two custom routes, employing the shortest path and driving-car preferences. After re-projecting both custom and SMTUC routes and stops, each route is segmented into sections of approximately 500 meters, resulting in the creation of two new layers: (1) SMTUC sections intersecting the custom routes and (2) SMTUC stops along the custom route. Additionally, leveraging ORS tools, a new layer with isochrones is created, as seen in Figure 3. An isochrone is a line or boundary on a map that connects points representing equal travel time or distance from a particular location. This new layer provided a visual analysis of the area accessible from each SMTUC stop along the route within 2, 5, and 7-minute thresholds.

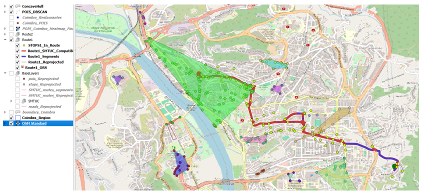


Figure 2: Route 1.

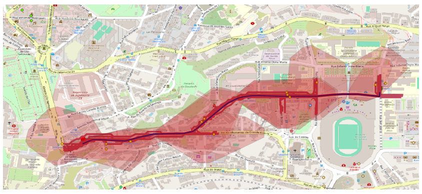


Figure 3: Route 2 with isochrones for 2 minutes.

Given the limitations of the ORS API for isochrones, no route with them is created to the center of the largest cluster of POIs. Nonetheless, through visual analysis, it can be confirmed that using this approach could indeed provide an effective tool for route customization based on POI concentration and public transportation.

6.2 Phase II

In the second phase, the system allows users to select POIs from any geographic location. These POIs are organized into top-level categories, each containing subcategories. For example, the Tourism category includes Hotels and Museums, the Amenity category features Bars and Cafes, and the Shop category encompasses Malls.

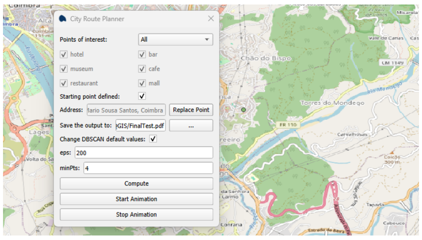
The main objective of this stage is to provide a more automated response to the challenge presented in the first phase of this project. A plugin for QGIS has been developed - Optimal Custom Route - which offers an intuitive and efficient tool for route planning and visualization.

The development environment includes *OSMnx* for geographic data handling, *OpenRouteService* for routing, *gpxpy* for GPS Exchange Format (GPX) file manipulation, *scikit-learn* for clustering, and *geopy* for geocoding. These libraries provide the necessary tools for implementing the plugin's core functionalities and can be installed using the following commands:

```
$pip install osmnx
$pip install openrouteservice
$pip install gpxpy
$pip install scikit-learn
$pip install geopy
```

The next step focuses on designing and implementing the user interface, developed using PyQt5. This interface comprises two main windows:

1. The first window allows users to input the name of the city for which they want to plan a route.
2. The second window is dedicated to route customization details, as shown in Figure 4.



City Route Planner

Points of interest: All

☑ hotel ☑ bar
☑ museum ☑ cafe
☑ restaurant ☑ mall

Starting point defined: ☑

Address: lario Sousa Santos, Coimbra [Replace Point]

Save the output to:)GIS/FinalTest.pdf [...]

Change DBSCAN default values: ☑

eps: 200

minPts: 4

Figure 4: Customization window with a starting point defined and customized DBSCAN parameters.

To handle geographic data, the plugin utilizes "OSMnx" to collect and process map data from OpenStreetMap. The process commences with geocoding the city name to acquire geographic coordinates. These coordinates are subsequently utilized to import the relevant map tiles into QGIS as layers, thereby offering users a visual representation of the area of interest.

For selecting POIs, users can choose from various categories, such as tourism, amenities, and shops. The plugin dynamically generates checkboxes for each subcategory, allowing for detailed selection. Once the POIs are selected, the plugin retrieves the corresponding data from OpenStreetMap and saves the data in a new layer with the name, category, and subcategory of the POI. Then, it proceeds to clustering using the pre-defined values of ε = 200 meters and minPts = 4 or custom values chosen by the user.

Clustering analysis plays a crucial role in the plugin, focusing on grouping POIs based on geographic proximity. To use the collected data, a transformation is needed. In this phase, the coordinates of the POIs are converted to a suitable coordinate system to ensure accurate distance measurements. Typically, the Universal Transverse Mercator (UTM) projection is used because it provides a more accurate representation of distances compared to latitude and longitude. This is essential for spatial data analysis, as the DBSCAN algorithm operates on distances between points. To ensure consistency in distance calculations, the POIs data is projected into the same CRS as the road network graph generated by *OSMnx*. This CRS transformation is important for aligning the POIs with the graph, allowing for accurate integration and subsequent analysis.

Once the data is transformed, a road network graph is created using *OSMnx*. This graph represents the road network within the specified place, where nodes correspond to intersections, and edges represent road segments connecting these intersections. The road network graph is truncated to retain only the largest connected component. This step is needed to avoid isolated nodes that do not contribute to the main network, ensuring a coherent and comprehensive road network for analysis. The truncated graph provides a strong foundation for mapping POIs and calculating network distances.

With the road network graph prepared, the POIs are projected into the same CRS as the graph to maintain consistency, and the nearest nodes in the road network graph are found for each POI. This way, distances between POIs can be calculated within the context of the road network.

The clustering process involves calculating a distance matrix, which is essential for applying the DBSCAN algorithm. Initially, a pairwise Euclidean distance matrix is calculated between the nodes representing the POIs. However, for more accurate distance measurements that account for the road network, this Euclidean distance matrix is converted into a network distance matrix using Dijkstra's algorithm. This algorithm computes the shortest path between nodes based on the actual road network distances. By using the network distance matrix, the DBSCAN algorithm can accurately cluster POIs based on real-world distances, rather than straight-line distances. The DBSCAN algorithm is then applied to this network distance matrix. The eps parameter, which is used in meters, defines the maximum distance between two points for them to be considered part of the same cluster. The minPts parameter specifies the minimum number of points required to form a dense region. The metric *precomputed* is used to indicate that the distance matrix has already been calculated.

After the clustering is performed, the results are processed to extract meaningful clusters. Noise points, which are points labeled as -1 by DBSCAN, are excluded from further analysis and the largest and densest clusters are identified. Subsequently, the outcomes are assessed using Silhouette Score and Davies-Bouldin Index.

These clusters are then visualized within the QGIS environment, providing an intuitive and comprehensive view of the spatial distribution of POIs. This visualization aids in identifying key areas of interest, as shown in Figure 5 and supports the generation of an optimal route.

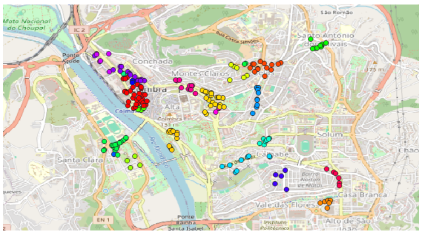


Figure 5: Clustered POIs.

Based on the clustered POIs, the route generation process initiates a request to the ORS API to compute the optimal route. Users can specify the starting point either manually or by utilizing the center of the largest cluster (the default by omission). Subsequently, the plugin selects waypoints from the largest cluster, applies a greedy TSP solver to determine the optimal order of waypoints, and generates the route using ORS. The resulting route is then visualized in QGIS (see Figure 6), providing users with an interactive map display. To enhance the user experience, the plugin includes a route animation feature, which reads GPX data and animates the movement along the route on the QGIS map canvas. Finally, it also supports exporting the created route and layers to a PDF, as an image.

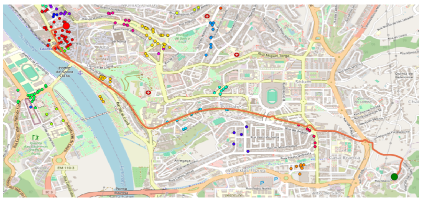


Figure 6: Route visualization in QGIS.

The integration of clustering analysis and advanced route generation techniques in the second phase represents a significant advancement in developing intelligent route recommendation systems. By allowing users to select POIs from a variety of categories and subcategories, the system provides highly personalized and efficient routing solutions. The use of DBSCAN for clustering POIs based on real-world distances ensures accurate and meaningful groupings, while the ORS API facilitates the generation of optimized routes. The inclusion of a user-friendly interface, interactive map displays, and features such as route animation and export options enhances the overall user experience, making the system a powerful tool for urban exploration and navigation.

7 EVALUATION

To assess the success of this work, the system is evaluated on functionality, user experience, performance, clustering effectiveness, and accuracy. The system's ability to accurately recommend routes based on user-selected cities and POI categories is assessed, as well as the effectiveness of the route generation engine in optimizing factors like distance (for phase II) and available public transportation options (for phase I), in real-time route recommendations. Each project phase meets expectations, demonstrating flexibility and efficiency in using user inputs to recommend routes within concentrated areas of interest. This aligns with the motivations described by (Mahmood et al., 2023), who highlights the importance of context-aware recommendations and optimized routes tailored to user preferences.

Regarding performance evaluation, the system demonstrates high efficiency under varying loads. In Phase I, a significant number of POIs are retrieved without delay. In Phase II, although fewer POIs are processed, more complex operations are executed in sequence (API calls followed by clustering analysis, display, and data export) within a few seconds. This real-time response capability underscores the practical application of advanced geospatial technologies and algorithms in urban navigation, as discussed in the introduction.

The reliability of clustering effectiveness in identifying concentrated areas of POIs and generating optimized routes within those zones is also assessed. To ensure clustering effectiveness, two metrics are used for validation: the Silhouette Score and the Davies-Bouldin Index. These metrics provide quantitative evaluations of clustering quality, confirming that the system effectively identifies meaningful clusters of POIs. However, some challenges are encountered in clustering effectiveness in phase II, particularly with results suggesting an overlap of clusters when using the same parameters as in phase I. This can be visualized in the layers and in the Silhouette Score with values ranging from -0.7 to -0.4 and the Davies-Bouldin Index with values from 1 to 2 or 3, depending on the parameter values. This cluster overlap could be attributed to the presence of various sources and categories for the POIs. In the first step, all the retrieved POIs are used for the clustering process, and in the second step, only a few categories are processed. Nonetheless, the overall results are promising. This finding corroborates the work of (Lou, 2022), who emphasizes the importance of accurate clustering for intelligent tourism route planning.

The system's clustering process benefits from the use of the DBSCAN algorithm, known for its robustness in handling spatial data and noise, as noted by (Lou, 2022). The application of DBSCAN, along with the conversion of Euclidean distance matrices into network distance matrices using Dijkstra's algorithm, allows for accurate clustering based on real-world distances. This approach is consistent with the findings of (Zhang et al., 2022), who highlights the importance of accurate distance measurements and intelligent technology in improving travel time predictions and route optimization.

In terms of user experience, the development of a custom QGIS plugin [1],"Optimal Custom Route", provides an intuitive and efficient tool for route planning and visualization. The user interface, designed using PyQt5, offers a seamless and interactive experience for selecting POIs and generating routes. The integration of clustering analysis, route optimization, and visualization within the QGIS environment enhances the system's usability and practicality, aligning with the envisioned AmI principles of creating smart environments that enhance daily living experiences (Thankachan, 2023).

In conclusion, the application achieves its primary goals of generating optimal routes that connect different POIs within selected cities, demonstrating high accuracy in visualization and spatial analysis results. By identifying areas with high concentrations of POIs (in both phases) and public transportation coverage (in Phase I), the system successfully provides personalized and efficient navigation solutions. Future work will focus on enhancing clustering techniques, integrating real-time data, optimizing performance, incorporating user feedback, and expanding the range of POIs categories to further improve the system's functionality and applicability.

8 CONCLUSIONS

This study successfully integrates AmI principles, Python programming, and GIS to develop intelligent route recommendation systems for urban exploration. The developed systems optimize travel routes, promote sustainable transportation options, and enhance user experiences. The research demonstrates the potential of AmI to address challenges in urban navigation and personalized recommendations, contributing to sustainable urban mobility.

The development process is divided into two phases. In the first phase, existing QGIS tools and plugins are utilized to manually create and ana-

[1] https://github.com/AgataPalma/OptimalCustomRoute

lyze routes based on POIs and public transportation data. This phase demonstrates the feasibility of using geospatial technologies to optimize urban navigation. The second phase involves the creation of a custom QGIS plugin, “Optimal Custom Route,” providing an automated and user-friendly interface for route planning and visualization. This phase leverages advanced machine learning techniques, specifically DBSCAN clustering, to identify dense areas of POIs and generate optimized routes.

The system’s performance is evaluated based on functionality, user experience, performance, clustering effectiveness, and accuracy. The results indicate that the system accurately recommends routes based on user-selected cities and POI categories, efficiently handles varying loads, and generates well-defined clusters of POIs. However, some challenges are encountered, particularly in clustering effectiveness when dealing with different sources and categories of POIs, which will need further refinement.

8.1 Future Work

While the current system shows promising results, several areas for future work can enhance its functionality and applicability:

- **Enhanced Clustering Techniques.** Future research could explore more advanced clustering algorithms and parameter tuning to improve clustering effectiveness, particularly when dealing with diverse categories of POIs.
- **Integration with Real-time Data.** Incorporating real-time data from public transportation systems, traffic conditions, and user location can enhance the system’s ability to provide dynamic and real-time route recommendations.
- **Extended POI Categories:** Expanding the range of POI categories and integrating additional data sources can provide more comprehensive and personalized route recommendations.
- **Mobile Application Development.** Developing a mobile app of the system can make it more accessible to users on the go, providing seamless and interactive route recommendations.
- **Sustainability Metrics.** Incorporating sustainability metrics, such as carbon footprint reduction and energy efficiency, into the route optimization process can further promote sustainable urban mobility solutions.

In conclusion, this research demonstrates the significant potential of integrating AmI, Python programming, and GIS in developing intelligent route recommendation systems. By addressing the identified challenges and exploring future research directions, ongoing advancements in AmI technologies and their practical applications can continue to improve urban living standards and mobility solutions.

REFERENCES

Gong, Q., Liu, J., and Cao, H. (2020). Csq system: A system to support constrained skyline queries on transportation networks. In *Proc. of Intl. Conf. on Data Engineering*, volume 2020-April, pages 1746–1749. IEEE Computer Society.

Karantaidis, G., Sarridis, I., and Kotropoulos, C. (2021). Adaptive hypergraph learning with multi-stage optimizations for image and tag recommendation. *Signal Processing: Image Communication*, 97.

Li, J., Zhang, L., Ni, C., An, Y., Zong, C., and Zhang, A. (2021). Efficient k nearest neighbor query processing on public transportation network. In *Proc. of 20th Intl. Conf. on Trust, Security and Privacy in Computing and Communications, TrustCom*, pages 1108–1115. IEEE.

Lou, N. (2022). Analysis of the intelligent tourism route planning scheme based on the cluster analysis algorithm. *Computational Intelligence and Neuroscience*, 2022:3310676.

Mahdi, A. J., Tettamanti, T., and Esztergar-Kiss, D. (2023). Modeling the time spent at points of interest based on google popular times. *IEEE Access*, 11:88946–88959.

Mahmood, M. R., Kaur, H., Kaur, M., Raja, R., and Khan, I. A. (2023). *Ambient intelligence and internet of things: An overview*, chapter 1, pages 1–32. John Wiley & Sons, Inc., 1 edition.

Panagiotakis, C., Daskalaki, E., Papadakis, H., and Fragopoulou, P. (2022). The tourist trip design problem with poi categories via an expectation-maximization based method.

Sarridis, I., Karantaidis, G., and Kotropoulos, C. (2022). Image driven optimal personalized route recommendation. In *IEEE IVMSP 2022*. Institute of Electrical and Electronics Engineers Inc.

Thankachan, D. (2023). *Introduction to Ami and IoT*, chapter 12, pages 361–381. John Wiley & Sons, Inc., 1 edition.

Zhang, X., Lauber, L., Liu, H., Shi, J., Xie, M., and Pan, Y. (2022). Travel time prediction of urban public transportation based on detection of single routes. *PLoS ONE*, 17:e0262535.

Özcan, S. and Kaya, H. (2018). An analysis of travelling salesman problem utilizing hill climbing algorithm for a smart city touristic search on openstreetmap (osm). In *2nd Intl Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pages 1–5.

Hyperparameter Optimization for Search Relevance in E-Commerce

Manuel Dalcastagné and Giuseppe Di Fabbrizio
VUI, Inc., Boston, U.S.A.
manuel.dalcastagne@gmail.com, difabbrizio@gmail.com

Keywords: Hyperparameter Optimization, Differential Evolution, e-Commerce Search Relevance Optimization.

Abstract: The tuning of retrieval and ranking strategies in search engines is traditionally done manually by search experts in a time-consuming and often irreproducible process. A typical use case is field boosting in keyword-based search, where the ranking weights of different document fields are changed in a trial-and-error process to obtain what seems to be the best possible results on a set of manually picked user queries. Hyperparameter optimization (HPO) can automatically tune search engines' hyperparameters like field boosts and solve these problems. To the best of our knowledge, there has been little work in the research community regarding the application of HPO to search relevance in e-commerce. This work demonstrates the effectiveness of HPO techniques for optimizing the relevance of e-commerce search engines using a real-world dataset and evaluation setup, providing guidelines on key aspects to consider for the application of HPO to search relevance. Differential evolution (DE) optimization achieves up to 13% improvement in terms of NDCG@10 over baseline search configurations on a publicly available dataset.

1 INTRODUCTION

Modern e-commerce platforms rely on search engines to help customers find relevant products from catalogs containing millions of items. Configuring these platforms is challenging and requires carefully modeling the query intent, product attributes, customer behavior, and other factors influencing relevance. Most search engines have numerous hyperparameters that can significantly impact both retrieval and ranking of results. Traditionally, these options are tuned manually in a time-consuming and often irreproducible process as the queries, products, and customer preferences evolve continuously over time.

In recent years, hyperparameter optimization (HPO) techniques have been successfully used to configure automatically many types of algorithms as well as complex machine learning models (Feurer and Hutter, 2019; Eggensperger et al., 2019). HPO employs a class of models usually called black-box or derivative-free, as no mathematical closed-form formulation of an objective function is necessary and the only requirement is a metric for numerical estimation. These techniques search through a multi-dimensional space of possible hyperparameter configurations to find the settings that optimize a performance metric such as NDCG (Wang et al., 2013).

To the best of our knowledge, there has been little work in the research community on e-commerce applications of HPO for search relevance. One notable exception is the work by Cavalcante et al. (Cavalcante et al., 2020), who used Bayesian Optimization to tune the ranking function of a customer support search application on a private dataset. However, their work did not explore different query structures, field boosting, query intent or query classification (Di Fabbrizio et al., 2024). Our work is one of the first to systematically apply HPO techniques to optimize relevance in e-commerce and to provide guidelines regarding the application of HPO to this context.

The main contributions of this work are: 1) application of differential evolution (DE) on a publicly available e-commerce dataset for search relevance optimization; 2) analysis of the dataset's label distribution impact on search relevance; 3) tuning of precision and recall-oriented Elasticsearch queries, and variants thereof, observing improvements up to 13% in terms of NDCG@10; 4) insights into the impact of field boosting, query structure, and query understanding on relevance; 5) guidelines on key aspects to consider when applying HPO to search relevance, such as the characteristics of the search space, multifidelity, or the use of multiple metrics for multi-objective optimization.

The remainder of this paper is structured as follows. Section 2 provides the problem definition. Section 3 introduces HPO and DE. Section 4 describes the WANDS evaluation dataset. Section 5 presents

Dalcastagné, M. and Di Fabbrizio, G.
Hyperparameter Optimization for Search Relevance in E-Commerce.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 399-407
ISBN: 978-989-758-716-0; ISSN: 2184-3228

setup, results, and analysis of the experiments. Finally, Section 6 concludes the paper and outlines potential future research.

2 PROBLEM DEFINITION

Users traditionally search by typing natural language queries that define what they are looking for (*user's intent*). As a response, a search engine retrieves and ranks a set of relevant documents from a corpus of possibly multiple document types, whose specifics are determined in dedicated document *schemas*. A *document type* is represented as a collection of named fields, also known as attributes or features, that are employed to build ranking signals quantifying the relevance of each field with respect to search queries.

2.1 Index Time and Query Time

Modern search engines index and query documents at separate times, but decisions taken at index time might impact on both the performance and quality of results retrieved at query time. At index time, the fields of each document are analyzed and indexed: each feature is divided into tokens, mapped to a type (e.g., string, numeric, date), processed and transformed in one or more fields that are indexed (i.e., according to the signals to be modeled). Also, details about the keyword and vector algorithms to be used for ranking are usually defined at this point. For example, if using BM25 (Robertson and Zaragoza, 2009) as a ranking algorithm, its b and k_1 hyperparameters could be optimized during this phase.

Although the application of HPO is possible at both stages, doing so at index time is significantly more expensive from a computational perspective - changes to the index usually require the reindexing of the whole corpus. This work focuses only on query-time applications of HPO, but the same techniques can be applied to optimize hyperparameters with impact at index time.

2.2 HPO for Search Relevance

The application of HPO involves two steps. First, define the hyperparameters to tune (i.e., type, range, relationships with other hyperparameters) and a budget to spend for the optimization process (e.g., number of function evaluations). Second, run an optimization loop where a search algorithm iteratively explores the space defined previously to find the best possible configuration of the hyperparameters by using some user-defined metric to evaluate each configuration.

In search relevance optimization (SRO), hyperparameters correspond to properties of the search engine query (e.g., values of field boosts, type of logical operators), while user-defined metrics are information retrieval (IR) metrics like precision, recall, or NDCG. Therefore, in order to evaluate a retrieval and ranking strategy over a corpus of documents, a dataset should contain a representative set of search queries and a collection of sets of relevance labels, defining the relevance of each document that could appear in the top results of each user query.

More precisely, let $\mathcal{D}$ be a dataset of triplets (q,d,y) where q is a search query, d is a document and y is a relevance label that defines the relevance of d for q, and let $\mathcal{S}$ be a search engine with a given index structure, whose output depends on a vector of hyperparameters $\theta \in \Theta^t$ that define an optimization search space of dimension t. The optimization goal is to heuristically find the best possible configuration θ^* by using a training dataset $\mathcal{D}_{train}$ to estimate the performance of $\mathcal{S}$ during the optimization and a validation dataset $\mathcal{D}_{val}$ to prevent overfitting, so that θ^* generalizes to a test dataset $\mathcal{D}_{test}$ that was not employed during the optimization. Ideally, all these datasets should be large enough to ensure statistically sound decisions. If $\mathcal{D}$ is not large enough, methods like k-fold-cross-validation can be used to split available data in folds to be combined as k training and test sets. Therefore, the performance of any θ is estimated as $p_{train,\theta} = \mathcal{S}(\theta, \mathcal{D}_{train})$ and, at the end of the optimization, the quality of θ^* is estimated as $p_{test,\theta^*} = \mathcal{S}(\theta^*, \mathcal{D}_{test})$. Finally, to evaluate the contribution of the optimization, the whole process is repeated k times and the optimized performance of $\mathcal{S}$ is estimated as

$$p_{test} = \frac{1}{k}\sum_{i=1}^{k} \mathcal{S}(\theta_i^*, \mathcal{D}_{test,i}) \qquad (1)$$

where θ_i^* is the best configuration found at optimization i by using $\mathcal{D}_{train,i}$ as training dataset and $\mathcal{D}_{test,i}$ as test dataset from the i-th split. It is important to highlight that all splits are based on folds coming from the same initial randomized sampling process. As a result, the repeated estimation and averaging over multiple splits results in an estimate of generalization error with lower variance (Kohavi, 1995).

3 OPTIMIZATION

HPO algorithms are usually classified as model-free (e.g., variants of stochastic search like *differential evolution*) or model-based (e.g., Bayesian optimization), where a model is used to estimate the response of the objective function to be optimized. Both approaches have advantages and disadvantages, and picking the right algorithm for the problem at hand depends on multiple factors that include search space characteristics (i.e., size, type of hyperparameters) or latency requirements (Feurer and Hutter, 2019; Bischl et al., 2023).

3.1 HPO Search Space and Latency

Due to the curse of dimensionality (Bellman, 1966), the size of the search space has a large influence on the optimization. The larger the size, the harder it is for the algorithm to find well-performing configurations of the hyperparameters. Furthermore, not all algorithms are able to scale with the number of dimensions. For example, standard Bayesian optimization (BO) based on Gaussian processes is not usually efficient on problems with more than 20 dimensions, but it excels in continuous spaces (Eggensperger et al., 2013; Frazier, 2018). In contrast, BO based on random forests and evolutionary algorithms like DE are not as efficient. Still, they are able to handle larger search spaces based on mixed hyperparameters as well.

When performance evaluations are computationally expensive, which can happen when the objective function requires training on large datasets, it might be helpful to consider multi-fidelity algorithms like Successive Halving (Jamieson and Talwalkar, 2016) or Hyperband (Li et al., 2017) to schedule monotonically the use of low-fidelity (less expensive) and high-fidelity (more expensive) evaluations during the optimization, to spend the budget efficiently. For example, DEHB (Awad et al., 2021) uses Differential evolution (DE) as an optimization algorithm to search θ in combination with a variant of *hyperband*, performing better than the more famous BOHB (Falkner et al., 2018) on a wide range of problems, including the tuning of deep learning networks.

Optimization algorithms differ as well in their parallelizability capabilities. In fact, model-free algorithms are usually more scalable since model-based methods are less parallelizable due to the presence of a common model that must be iteratively updated. For more details, refer to (Feurer and Hutter, 2019; Bischl et al., 2023).

3.2 Differential Evolution

The optimization algorithm used in the experiments is Differential evolution (Storn and Price, 1997), which is an evolutionary algorithm inspired by the concepts of biological evolution and natural selection, specifically by how the offspring inheriting the best traits of a population evolve over generations.

At the beginning of the process, a population $p_0 = (\theta_1, \dots, \theta_n)$ is randomly sampled from Θ^t. Until some user-defined optimization budget b is consumed, DE works iteratively in three steps: mutation, crossover, and selection. During the mutation phase, each member θ of the population p_i at the current iteration i is evaluated by computing $\mathcal{S}(\theta, \mathcal{D}_{train})$. Then, a new set of n offsprings is generated by applying a scaled perturbation to each dimension of a new offspring θ_{new} resulting from the combination of randomly picked parents from p_i. A crossover operator combines each member of p_i with one of the new offsprings θ_{new}, by picking for each dimension with some probability which value from the two vectors should be used for the mutant configuration θ_{mutant}. Finally, θ_{mutant} is compared with θ, and θ_{mutant} possibly takes place of θ if its quality is better.

4 EVALUATION DATASET

The Wayfair Annotation DataSet (WANDS) is an open-source e-commerce product dataset designed to evaluate the relevancy of e-commerce product search engines (Chen et al., 2022). As described in Table 1, the WANDS dataset contains:

- 480 search queries sampled from real search logs of Wayfair, a major e-commerce retailer, with two features: query text and class. For example, a query like *smart coffee table* belongs to the *Coffee & Cocktail Tables* class. The queries were stratified sampled to cover various dimensions such as popularity, seasonality, and whether they led to customer purchases. This ensures the query set is representative of real customer search behavior.
- 42,994 products sampled from Wayfair's catalog, with nine features of which only the following five textual features were used for field boosting in the experiments: product name, class, description, category hierarchy and list of features. For example, a product named *solid wood platform bed* belongs to the *Bed* class within a category hierarchy like *Furniture / Bedroom Furniture / Beds* and has a list of features that contains information like color, material, size or weight.

  For each query, Wayfair selected a set of potentially relevant products using a combination of customer click logs, lexical search systems, and neural retrieval models. Specifically, the dataset authors employed two strategies to construct the product pool:

1. They leveraged user engagement data (clicks and add-to-cart events), hypothesizing that products users clicked on are a good approximation of potentially relevant products, while products users clicked on but didn't add to the cart could be hard negatives or almost-relevant products.
2. They further mined the product catalog using an open-source lexical search engine (Apache Solr) and a neural product retrieval system inspired by (Nigam et al., 2019). The two systems provide complementary ways to retrieve relevant products, removing the bias related to the use of a single lexical retrieval source. Moreover, this hybrid approach ensures the product set contains both obviously relevant products as well as more challenging cases that can help discriminate between different retrieval systems.

- 233,448 (query, product) pairs assigning one out of three relevance labels to the match of query and product: exact (1.0) if the product is completely relevant to the query, partial (0.5) if the product matches some but not all aspects of the query, and irrelevant (0.0) if the product is not relevant to the query.

Note that the statistics are based on the most recent version available on GitHub[1] which is slightly different from the version in (Chen et al., 2022).

A group of trained human annotators provided the labels following a rigorous set of annotation guidelines. Each *(query, product)* pair was judged by up to 3 annotators, and the ratings were aggregated using a majority vote. The WANDS dataset was constructed through multiple rounds of annotation and refinement. The inter-annotator agreement, measured by Cohen's Kappa (Cohen, 1960), improved from a moderate 0.467 in the initial months to a substantial 0.826 after a few iterations of guideline refinement and annotator training. This indicates the dataset labels are of high quality and consistency.

A key feature of WANDS is that it aims for *completeness* - i.e., for a given query, the dataset tries to include relevance labels for *all* the relevant products from the catalog subset, not just the top few results. This is achieved through an iterative "cross-referencing" process during dataset construction that identifies potentially relevant products that were not covered in the initial labeling. Completeness is important for unbiased offline evaluation as it avoids missing relevant products that could unfairly penalize certain retrieval systems. The complete, multi-graded relevance labels allow for a robust evaluation of the ranking quality of search engines using metrics like NDCG.

To evaluate the difficulty of the search relevance task in the WANDS dataset, we analyzed the distribution of relevance labels (exact match, partial match, irrelevant) across the queries. The goal was to understand how many queries have products labeled as only exact matches, only partial matches, only irrelevant, or a mixture of these labels. This analysis provides insights into the difficulty of ranking the search results for each query.

Assuming that, on average, each query contains the same proportion of exact, partial, and irrelevant labels as the overall distribution in the dataset, we found that:

- 0 queries have products with only the Exact label, 24 queries have products with only the Partial label, 1 query has products with only the Irrelevant label
- 33 queries have products with only Exact and Partial labels, 11 queries have products with only Exact and Irrelevant labels, 76 queries have products with only Irrelevant and Partial labels

This analysis reveals that 25 queries do not have an impact on NDCG, and 11 queries should have results that are relatively easy to rank. Around 100 queries are of medium difficulty, while the rest are more challenging. However, the distribution of labels across queries is not balanced, which is an important consideration for learning to rank (LtR) models (Goswami et al., 2018). If the number of labels per type is imbalanced, the model may be more prone to overfitting. For example, queries with a highly skewed distribution of exact and partial matches are easier to achieve a good NDCG score compared to queries which have a more balanced distribution of exact and partial matches.

This analysis highlights the importance of considering the distribution of relevance labels when evaluating the difficulty of the search relevance task and the potential impact on the performance of ranking models. The WANDS dataset provides a diverse set of queries with varying levels of difficulty, making it a valuable resource for evaluating and comparing different search engines and ranking algorithms in the e-commerce domain.

5 EXPERIMENTS AND RESULTS

We provide experimental results to demonstrate how hyperparameter optimization can be leveraged to

[1] https://github.com/wayfair/WANDS

Table 1: Summary of key data statistics about the WANDS dataset.

Feature	**Value**
Number of queries	480
Number of products	42,994
Number of (query, product) relevance labels	233,448
Relevance label scale	0-2
Relevance label distribution	Exact: 25,614 Partial: 146,633 Irrelevant: 61,201
Search queries from	Real search logs of Wayfair
Products sampled from	Wayfair's catalog
Annotators per (query, product) pair	Up to 3
Inter-annotator agreement (Cohen's Kappa)	0.826

automate solutions to many information retrieval and search problems commonly encountered in e-commerce. The focus is on optimizing hyperparameters of search queries and ranking signals used by search engines in keyword search. Elasticsearch is used as an experimental framework, but the techniques mentioned in this section are applicable to any other engine that supports the manual tuning of its components.

Experiments start from the consideration that both TF-IDF and BM25 have some ranking strategy limits, which can be partially addressed through the use of optimization for field boosting. It is worth mentioning that well-tuned boosts are critical not only to rank the expected importance of different signals but also to balance the range of the respective BM25 scores.

5.1 BM25 Limits and Field Boosting

Scores based on TF-IDF have some shortcomings, which are partially solved by the BM25 formulation. TF-IDF's score for a term in a corpus is computed as the product of term frequency and inverse document frequency. A problem comes from the unconstrained impact of term frequency on the score (i.e., a term that appears *n* times in a document implies that a document is *n* times more relevant than another document without any occurrence). Also, the length of a document does not weight the relevance of its terms (e.g., if a term appears once in a document containing 10 words, it is considered to be as relevant as if the term appears once in a document containing 1000 words). BM25's *b* parameter restrains the degree to which term frequency can impact the score, determining a penalty for documents longer than the average, and the influence of common terms on the score is saturated by BM25's k_1 parameter.

Nonetheless, the scores of fields can be on different scales due to distribution differences of frequencies and document lengths and are, therefore, not directly comparable. Also, by definition, these scores are biased towards information, usually against users' needs (i.e., rare matches within a document score higher, while users usually look for popular items). Field boosting helps counterbalance the aforementioned problems by prioritizing and balancing signals from different fields. In fact, a search query usually contains more than one string and possibly multiple concepts. It does not come as a surprise that the information required to return relevant results is often stored in multiple fields.

Elasticsearch tries to solve some of TF-IDF's problems by changing how token frequencies are combined to compute scores during a multi-field search by considering the frequencies coming from multiple fields at the same time. In particular, field-centric search (e.g., `multi_match best_fields` and `most_fields`) focuses towards precision by promoting results which satisfy criteria based on the signals which are expected to match the user's search, while term-centric search (e.g., `cross_fields`, `combined_fields`) focuses towards recall, by selecting all possibly relevant search results (Turnbull and Berryman, 2016). The use of either conjunctive (AND) or disjunctive operator (OR) further pushes these queries towards precision or recall, respectively.

The combination of recall-oriented and precision-oriented clauses in a *stratified* query improves the ranking of the results returned to the user (Turnbull and Berryman, 2016). In Elasticsearch, this can be achieved using a boolean query, which matches documents satisfying boolean combinations of other queries (e.g., `multi_match` queries), where some clauses provide a recall-oriented base score that is improved by other precision-oriented clauses. For example, the base score may come from a `multi_match cross_fields` query searching in all text fields, while other scores may come from `multi_match best_fields` or `most_fields` queries based on high-quality signals.

5.2 User's Intent

Understanding the user's intent is another critical signal that significantly improves search relevance in e-commerce. This involves classifying whether the user is searching for a specific product category or asking a broader, more informational question. By using a machine learning model to predict the user's intent based on query structure, search patterns, and historical behavior, the system can adjust its ranking strategy to deliver more relevant results. For example, when seeking a specific product, search results can prioritize relevant items from the desired category. Conversely, if the user's query is informational, the system can prioritize results such as FAQs, reviews, or other informative content. Incorporating intent prediction into the search optimization process allows for more accurate recommendations and a highly personalized shopping experience.

5.3 Multi-Objective Optimization

In the experiments, we use NDCG@10 over the labeled dataset to evaluate the relevance performance of a given search engine configuration θ. The normalized discounted cumulative gain (NDCG) (Wang et al., 2013) measures the relevance of the top-ranked results, putting more emphasis on the relevance of results at higher ranks (Järvelin and Kekäläinen, 2000). This aligns well with users' behavior and preferences on e-commerce search result pages, who tend to focus mainly on the first page of results. Still, while a single metric is a good starting point for assessing the quality of search relevance performance, it might only tell part of the story.

NDCG assumes that labeled documents are uniformly distributed in the ranked list, which is usually untrue. In Section 4, we showed that even a well-built dataset like WANDS falls in more extreme situations where not all relevance labels are found for more than 100 use queries, and in some cases, only one class of relevance labels might be retrieved. A metric like NDCG cannot detect such scenarios and would return a perfect value even if some queries were evaluated, for example, only on irrelevant documents. To obtain robust evaluations, one should combine at least an order-aware metric like NDCG or Mean Reciprocal Rank with an order-unaware metric like Precision or Recall. For further details about these indicators or variants thereof, please refer to (Valcarce et al., 2018).

The optimization of multiple equally important but conflicting objectives is named multi-objective optimization, where solutions that optimize all objectives simultaneously usually do not exist (Helfrich et al., 2023). In this scenario, heuristic algorithms try to find efficient, non-dominated solutions concerning the defined objectives. An alternative solution is to employ scalarization techniques to systematically approximate a multi-objective optimization problem into a regular single-objective optimization problem with the help of additional parameters such as weights and use regular optimization problems to solve the resulting scalarization. For further details about multi-objective HPO algorithms, please refer to (Feurer and Hutter, 2019; Bischl et al., 2023).

5.4 Experimental Setup

Text fields from WANDS were indexed using Elasticsearch's English analyzer, without any additional preprocessing steps. In particular, all experiments were run on Elasticsearch 8.8.2 and Python 3.10. To ensure replicability and improved comparison of results, all splits and optimization runs were carried out multiple times with a common set of random seeds. This ensured that the evaluations utilized to build estimators were paired. In addition, we computed random ranking values based on 5 repetitions, similar to how k-fold cross-validation was employed with $k = 5$. According to the experiment, the search space size varies from 8 to 27 dimensions, and each optimization run is executed up to a budget b of 400 function evaluations. Results on the test set are considered only for evaluation purposes at $b = 50$, 100, 200, and 400. Unless explicitly defined, the experiments' optimized hyperparameters were defined as in Table 2. For further details about the role of these hyperparameters in multi-field queries, please refer to Elasticsearch documentation. Finally, the DE implementation used in the experiments is the default version available on GitHub[2] from the Python package created by the authors of DEHB.

5.5 Random and Standard Baselines

In this work, we consider both random and standard ranking as baselines against which to evaluate the contribution of HPO. The random ranking provides a ranking baseline for the problem, by assigning to each document from the set of results of a retrieval strategy a pseudorandom number in the range [0,1]. As a result, it is possible to compute any performance metric on the resulting ranked list. For example, if using NDCG, higher values imply easier ranking problems. Similar considerations can be achieved analyzing the distribution of relevance labels across the dataset.

[2] https://github.com/automl/DEHB

Table 2: Hyperparameter used in the experiments.

Name	Type	Range	Default value
operator	categorical	{and, or}	or
type	categorical	{best_fields, most_fields, cross_fields}	none
minimum_should_match	ordinal	{0%, 20%, 40%, 60%, 80%, 100%}	none
tie_breaker	float	[0, 1]	0
boost	float	[0, 100]	1

Standard ranking quantifies how the standard configuration of a search engine's ranking strategy performs with respect to a completely random ranking strategy. Unlikely the general HPO scenario, where good initial configurations of hyperparameters are usually unknown, search engines come with default values that work well on average. As a consequence, the corresponding ranking performance should be considered as well as a baseline.

5.6 Optimization Improvements

The contribution of the optimization to ranking strategies is empirically estimated by showing the improvement that DE is able to achieve with respect to the standard ranking of multiple retrieval queries with a fixed structure and increasing difficulty. Results show that the optimization contributed, on average, to an improvement of approximately 0.05 in terms of NDCG@10 on 12 cases. Our optimization strategy was not only employed to fully optimize both retrieval and ranking parts of each type of Elasticsearch query used in the experiments, but it also proved its effectiveness. It was able to reach comparable results with respect to its optimized counterparts with a fixed retrieval structure, providing reassurance about its success.

Results were built on three main types of Elasticsearch queries that were increasingly difficult. In the first set of experiments (Table 3, top), basic types of multi-field query are used distinctively in combination with both conjunctive and disjunctive operators. On average, the optimization achieves an improvement of 0.07. Once optimized, precision-oriented queries achieve the same results, and therefore, only one of the two is going to be considered in the following experiments. A Boolean query is employed to build a stratified query in the second set of experiments (Table 3, middle). On average, the optimization achieves an improvement of 0.05. The best results are interestingly achieved by combining a recall-oriented query based on the conjunctive operator and a precision-oriented query based on the disjunctive operator. In the third set of experiments (Table 3, bottom), the best stratified query from the previous experiments is extended with an additional multi-field query that considers user intent. On average, the optimization achieves an improvement of 0.04, and the introduction of user intent contributes approximately 0.03 - 0.04 with respect to the best results from the previous sets of experiments.

5.7 Retrieval Relaxation Improvements

All results show that, on average, queries using the conjunctive operator perform worse than queries adopting the disjunctive operator. In particular, random ranking results allow us to infer that performance values can be improved by relaxing the matching requirements and retrieving more potentially relevant documents that could be otherwise excluded from further ranking refinement. This behavior aligns with modern multi-stage IR systems that rely on multiple ranking phases, where the first phase focuses on recall and successive steps towards precision (Dang et al., 2013; Zhou and Devlin, 2021).

6 CONCLUSIONS

This work demonstrates the potential for HPO techniques to substantially improve the search relevance of e-commerce engines with minimal human effort in a reproducible and automatic process, providing insights into the impact of field boosting, retrieval query structure, and query understanding on relevance, as well as guidelines on the application of HPO to search relevance in e-commerce.

By leveraging the WANDS evaluation dataset and DE as HPO algorithm, we automatically optimized both retrieval and ranking strategies of Elasticsearch queries, improving NDCG@10 up to 13% with respect to baseline configurations. The introduction of the user's intent in the search strategy, defined as correspondence between the category of user query and document, brought an improvement of up to 4 %. Finally, results showed that the relaxation of the retrieval strategy led to significantly better results. Default search engine configurations leave significant room for relevance improvements that can be un-

Table 3: Best results from the first set (top), the second set (middle), and the third set of experiments (bottom). All performance metrics are expressed as averaged NDCG@10 with standard deviation, and results with highest average are in bold for each column.

Query type	**Operator**	**Space Size**	**Random**	**Standard**	**Optimized**
cross_fields	OR	8	0.53 ± 0.01	0.60 ± 0.00	0.73 ± 0.02
cross_fields	AND	8	0.49 ± 0.00	0.52 ± 0.01	0.59 ± 0.02
best_fields	OR	8	0.54 ± 0.01	0.60 ± 0.01	0.73 ± 0.01
best_fields	AND	8	0.46 ± 0.00	0.48 ± 0.01	0.52 ± 0.04
most_fields	OR	8	**0.60 ± 0.00**	**0.69 ± 0.00**	0.73 ± 0.01
most_fields	AND	8	0.49 ± 0.00	0.51 ± 0.01	0.52 ± 0.04
optimized	optimized	10	/	/	**0.75 ± 0.02**
stratified	OR, OR	17	0.62 ± 0.00	0.71 ± 0.00	0.74 ± 0.02
stratified	OR, AND	17	0.59 ± 0.00	0.64 ± 0.00	0.74 ± 0.03
stratified	AND, OR	17	**0.64 ± 0.00**	**0.72 ± 0.00**	**0.75 ± 0.02**
stratified	AND, AND	17	0.52 ± 0.00	0.55 ± 0.01	0.58 ± 0.02
optimized	optimized	21	/	/	0.74 ± 0.02
stratified, most_fields	AND, OR, AND	21	0.65 ± 0.00	0.74 ± 0.00	0.77 ± 0.02
stratified, cross_fields	AND, OR, OR	21	0.65 ± 0.00	0.74 ± 0.00	0.78 ± 0.03
optimized	optimized	27	/	/	**0.78 ± 0.02**

locked with HPO, through a reproducible process that does not keep humans in the never-ending loop of manual search relevance optimization.

Picking the best algorithm for search relevance optimization depends on various factors including the size and type of hyperparameters, as well as multi-fidelity and multi-objective requirements. Evolutionary algorithms like DE are capable of handling large mixed search spaces, but unless the size of the search space goes beyond hundreds of dimensions, random-forests-based BO is another possible option. Furthermore, when performance evaluations are expensive due to the need for large datasets, options such as multi-fidelity HPO algorithms should be considered. Finally, to obtain robust configurations, one should consider multi-objective HPO algorithms to optimize for both order-aware and order-unaware metrics, or to create a scalarization of such metrics to apply regular HPO algorithms like DE.

While the optimal configuration will vary for each search application, this work establishes a general framework, methodology, and best practices for applying HPO to improve search relevance. With the increasing availability of easy-to-use HPO libraries and their integration with popular search engines, we believe this is a highly promising direction to improve the search experience for e-commerce customers with less manual effort and greater reproducibility.

This work focuses on optimizing keyword-based search, but it is worth noting the complementary role of dense vector search using learned semantic representations (Mitra and Craswell, 2018). In many search use cases where user queries primarily consist of named entities like product names or brands, exact keyword matching remains critical and even preferable. However, modern search engines offer hybrid search capabilities that combine the strenghts of sparse keyword-based retrieval with dense vector search. This hybrid approach is commonly used in retrieval augmented generation (RAG) architectures, as purely semantic search can miss obvious keyword matches needed for accurate product retrieval in e-commerce and for more strongly grounded factual knowledge retrieval (Lewis et al., 2020).

Finally, other several exciting avenues for future work in this area include:

- Exploration of the benefits that HPO can bring to hybrid search, such as improvements to the fine-tuning process of embedding models used in dense vector search or the configuration of other hyperparameters used in multi-stage IR systems;
- Application of multi-objective optimization to jointly optimize multiple metrics that measure different aspects of the results;
- Investigation of possible interactions as well as differences between HPO and LtR techniques for search relevance.

REFERENCES

Awad, N. H., Mallik, N., and Hutter, F. (2021). DEHB: Evolutionary hyperband for scalable, robust and efficient hyperparameter optimization. *Proceedings of IJCAI*.

Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., et al. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*.

Cavalcante, L., Lima, U., Barbosa, L., Gomes, A. L., Éden Santana, and Martins, T. (2020). Improving Search Quality with Automatic Ranking Evaluation and Tuning. In *Anais do XXXV Simpósio Brasileiro de Bancos de Dados*, Brasil.

Chen, Y., Khrennikov, D., Ferrer, I., and Verberne, S. (2022). WANDS: A Dataset for Web-based Product Search. In *European Conference on Information Retrieval*, pages 61–75. Springer.

Cohen, J. (1960). A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*.

Dang, V., Bendersky, M., and Croft, W. B. (2013). Two-stage learning to rank for information retrieval. In *Advances in Information Retrieval*. Springer.

Di Fabbrizio, G., Stepanov, E., and Tessaro, F. (2024). Extreme Multi-label Query Classification for E-commerce. In *The SIGIR 2024 Workshop on eCommerce*, Washington, D.C., USA.

Eggensperger, K., Feurer, M., Hutter, F., Bergstra, J., et al. (2013). Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *NIPS workshop on Bayesian Optimization in Theory and Practice*, Nevada. Curran Associates, Inc.

Eggensperger, K., Lindauer, M., and Hutter, F. (2019). Pitfalls and best practices in algorithm configuration. *Journal of Artificial Intelligence Research*, 64:861–893.

Falkner, S., Klein, A., and Hutter, F. (2018). Bohb: Robust and efficient hyperparameter optimization at scale. In *International conference on machine learning*.

Feurer, M. and Hutter, F. (2019). *Hyperparameter Optimization*, chapter 1, pages 3–38. Springer, Cham.

Frazier, P. I. (2018). Bayesian optimization. In *Recent advances in optimization and modeling of contemporary problems*, pages 255–278. Informs.

Goswami, A., Zhai, C., and Mohapatra, P. (2018). Learning to rank and discover for e-commerce search. In *14th International Conference on Machine Learning and Data Mining in Pattern Recognition (MLDM 2018)*, pages 331–346, Germany. Springer.

Helfrich, S., Herzel, A., Ruzika, S., and Thielen, C. (2023). Using scalarizations for the approximation of multiobjective optimization problems: towards a general theory. *Mathematical Methods of Operations Research*, pages 1–37.

Jamieson, K. and Talwalkar, A. (2016). Non-stochastic best arm identification and hyperparameter optimization. In *Artificial intelligence and statistics*.

Järvelin, K. and Kekäläinen, J. (2000). IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd Annual International Conference on Research and Development in Information Retrieval*, New York, NY, USA. Association for Computing Machinery.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, page 1137–1143, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2017). Hyperband: a novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.*, 18(1):6765–6816.

Mitra, B. and Craswell, N. (2018). An introduction to neural information retrieval. *Foundations and Trends in Information Retrieval*.

Nigam, P., Song, Y., Mohan, V., Lakshman, V., Ding, W. A., Shingavi, A., Teo, C. H., Gu, H., and Yin, B. (2019). Semantic product search. In *Proceedings of KDD*, New York, NY, USA. Association for Computing Machinery.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Storn, R. and Price, K. (1997). Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359.

Turnbull, D. and Berryman, J. (2016). *Relevant Search: With applications for Solr and Elasticsearch*. Manning Publications Co., USA.

Valcarce, D., Bellogín, A., Parapar, J., and Castells, P. (2018). On the robustness and discriminative power of information retrieval metrics for top-n recommendation. In *Proceedings of the 12th ACM conference on recommender systems*, pages 260–268.

Wang, Y., Wang, L., Li, Y., He, D., and Liu, T. (2013). A Theoretical Analysis of NDCG Type Ranking Measures. In *COLT 2013 - The 26th Annual Conference on Learning Theory*.

Zhou, G. and Devlin, J. (2021). Multi-vector attention models for deep re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5452–5456.

A Framework for Self-Service Business Intelligence

Rosa Matias and Maria Beatriz Piedade
Computer Science and Communications Research Centre (CIIC), School of Technology and Management (ESTG),
Polytechnic University of Leiria, Portugal
{rosa.matias, beatriz.piedade}@ipleiria.pt

Keywords: Self-Service Business Intelligence, Data Visualization, Data Storytelling.

Abstract: Building an effective Business Intelligence solution involves several key steps. Recently, low-code software tools have allowed casual users - those with domain-specific knowledge of a case study - to develop custom solutions independently of IT teams. This is the era of Self-Service Business Intelligence. However, some drawbacks have been identified due to casual users' lack of Business Intelligence expertise. In response, a framework is proposed, introducing the role of casual power users and specifying the Business Intelligence knowledge they should possess. Additionally, the framework aims to integrate Business Intelligence methodologies more cohesively with data visualization and data storytelling development cycles. As a proof of concept, the framework was applied to develop a solution for monitoring class attendance at a higher education institution. In this case study, a casual power user is able to identify, early in the semester, which classes require adjustments to improve resource management and pedagogical outcomes. The contextualization provided by the framework enabled that user to successfully uncover critical insights.

1 INTRODUCTION

For quite some time, the digital technological expansion has contributed to the accumulation of data. Depending on the amount of data, the size of datasets may vary from small to medium or even enormous. Regardless of dataset size, data analysis is essential for gaining insights, as datasets consist solely of facts. Data analysis facilitates the transformation of these facts into information, that together with the users' background knowledge about the domain of analysis, enables wisdom and consequentially impactful decisions. Data analysis can be utilized to explore historical data, forecast future events, and recommend actions to achieve optimal outcomes. The first is called descriptive analysis, the second predictive and the third prescriptive (Sharda, Delen, & Turban, 2018). Business Intelligence (BI) solutions are data-driven systems created to help organizations gather, organize, and present data, from multiple systems, providing insights that facilitate informed decision-making. It may also support data analysis as part of a larger process. In organizations such a solution comprises a set of methods, processes, architectures, applications, and technologies that collectively transform raw data into insights (Evelson, & Norman, 2008), facilitating operational, tactical, or strategic decision-making.

Lennerholt, Van Laere, and Söderström (2021) identify 2 types of BI users: power users and causal users (Lennerholt, Van Laere, & Söderström, 2021). The first type has technical and theoretical knowledge to develop BI solutions but lacks problem-domain specific knowledge. The second type has no technical or theoretical knowledge to develop BI solutions but possesses problem-domain specific knowledge about the scenario under study.

Until recently, BI solutions were typically developed by users with technical and theoretical expertise and teams integrates users with domain knowledge for requirements gathering (for instance).

The emergence of tools such as Power BI (Microsoft, 2024), Tableau (Tableau,2024), and Qlik (Qlik, 2024) enable the development of Self-Service Business Intelligence (SSBI) solutions by casual users that usually do not possess technical expertise (Arnaboldi, Robbiani, & Carlucci, 2021). Those tools intend to be interactive, visual oriented, user-friendly, with low code features. As a result, casual users, with their background knowledge, are now apparently capable of interacting with BI tools and building their own dedicated solutions.

Matias, R. and Piedade, M.
A Framework for Self-Service Business Intelligence.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 408-415
ISBN: 978-989-758-716-0; ISSN: 2184-3228

By definition, a SSBI solution consists of a set of processes and tools that enables non-technical users to obtain, integrate, analyse, and visualize data without the need for traditional BI solution (Lennerholt, Van Laere, & Söderström, 2018).

Nevertheless, despite the possibility of developing a SSBI solution with low code, there are studies that identify difficulties in doing so. Suprata (2019) stated that several companies struggle to develop impactful data-driven dashboards due to complex datasets, inadequate dashboard design, and ineffective data storytelling (Suprata, F. ,2019).

This work proposes a new SSBI role: the casual power user. It is a user with domain knowledge about the scenario under analysis who also possesses fundamental theoretical and technical knowledge about BI. A framework is proposed for them to serve as a guide for developing SSBI solutions. It aims to bridge the gap between causal users and the concepts associated with BI. The framework considers well-known stages in BI methodologies but presents simplified, dependent, and interrelated stages for SSBI. Usually, BI methodologies have sequential steps, with some stages done in parallel. To the best of our knowledge, there is no framework capable of helping causal power users in the development of SSBI solutions with simplified, dependent and interrelated stages. The framework intends to incorporate in a stricter way traditional BI methodologies with dashboard and data storytelling development cycles.

As a proof of concept, a prototype has been built by a casual power user using the proposed framework. The case study aims to discover patterns regarding class attendance in a higher education institution. The project objectives are to assist managers in identifying classes with the highest and lowest number of student and to determine attendance behaviour over the weeks. With the identified insights, they can make informed decisions about class rearrangements and improve resources management.

The structure of the document is organized as follows: first, theoretical background; then, the proposed framework; and next, the case study. Finally, considerations are discussed, and conclusions are drawn.

2 THEORICAL BACKGROUD

This section provides an overview of concepts associated with BI and SSBI and about dashboards and data storytelling.

2.1 From Business Intelligence to Self-Service Business Intelligence

Bost by activities such as day-to-day events, social media or IoT sensors the data is being generated at impressive velocity, with variety and high volume. The raw data hide patterns with valuable insights. Generally, users want to explore it in an agile, interactive, and efficient manner. Regardless of the type of organization users belong to, or the role they play, at any given time, they want to access summarized data, drill down into the details, and study it from diverse perspectives. Also enrich it with data from external sources. In organizations one of the solutions to analyse the data and to monitor performance indicators is through BI systems. Therefore, organizations from various sectors have begun to adopt them, and they are now widely used for multiple purposes. For instance, in sectors such as education, health, commerce, industry, government, among others.

A BI system may include a data warehouse structured in agreement with the dimensional model, which supports data querying and data exploration. It is also be supported by extract, transform and load (ETL) processes and dedicated applications (Kimball, 2016).

In a BI project it is essential the stakeholders' background knowledge regarding subjects to ensure that the objectives are properly understood and addressed. With the rise of SSBI they turn into casual users and usually do not possess technical and theoretical expertise to build common a BI solution. Nevertheless, they have an enormous knowledge about the scenario or case under study. The advent of SSBI solutions has gaining popularity because the stakeholders have now interactive and low-code applications capable of certain independence from power users. The necessity for SSBI is unavoidable, as it enables businesses to extract information as needed and make informed decisions. (Zaghloul, Ali-Eldin, & Salem, 2013). It is a democratization process where users have the possibility and independence of building their own BI solutions (Arnaboldi, Robbiani, & Carlucci, 2021). A SSBI allows non-technical users to independently utilize BI tools, reducing their dependence on technical support (Lennerholt, Van Laere, & Söderström, 2021). Consequently, the role of casual users has changed (Dedić, & Stanier, 2017) and at present they have dedicated tools to perform specific analysis as required and on-the-fly.

Recently, Olaoye, & Potter (2024) stated the key components that work together in a BI environment are: data integration, data warehousing, reporting and dashboards, data visualization, advanced analytics, self-service analytics, data governance and collaboration (Olaoye, F., & Potter, K., 2024).

2.2 From Data Visualization to Data Storytelling

A dashboard is a visual tool that displays the key information required to accomplish the organization's goals. The data is organized on a screen, enabling easy and immediate monitoring (Few, 2006; Schwendimann et al., 2016). In agreement with the well-known enterprise organizational pyramid, the dashboards are classified as operational, tactical or strategic (Few, 2006). The first to monitor day-to-day activities, the second to take medium term decisions and the third to perform long-term strategic decisions by senior management executives.

A useful starting point for organizing the elements in a dashboard is the following Information Visual Mantra (Shneiderman, 2003): overview first, zoom and filter, then details on demand. Additionally, when building a dashboard, it's important to consider the appropriate visual elements to effectively display the relevant data. Suprata (2019) gives an overview about the relationship between charts and specific display proposes (Suprata, 2019).

Chokki et al. (2022) specifies the stages to build a dashboard, for instances, pick the metrics, collect the data, ensure quality, consider the audience, choose the best visualization practices, choose the best charts, provide easy to use tools, provide clear presentation, context and data interpretation, think of the audience, ensure data is up to date, allow access to data source and privacy, provide interactive support and allow customization (Chokki et al., 2022).

Sorour & Atkins (2024) propose a data cycle to develop dashboards with following steps: metrics choice, data collection, data processing, data analysis, building the dashboard layout, integrating visualizations in the layout and deployment (Sorour & Atkins, 2024).

The development of dashboards is a main factor for good stories as they are based on frames and pictures obtained from them.

For all the times humans use stories to gain attention, communicate and pass knowledge (Dykes, 2019). As so data storytelling has become an essential step in BI. The data only speaks if the right message is passed to users. Suprata (2019) proposed the following approach to develop a data story (Suprata, 2019): define the audience, frame insights, establish setting or context, focus on the story elements and consolidate and practices.

3 THE PROBLEM

Although casual users can interact with SSBI tools, they may lack the fundamental theoretical and technical concepts necessary to develop an effective SSBI solution. A SSBI software tool is a simplified version of traditional BI software, specifically designed for casual users. However, it is essential that these users also possess theoretical and technical knowledge to build a high-quality solution. In an SSBI tool, the user interaction is apparently easy, but this does not necessarily mean that data will be handled correctly. Indeed, it may happen that casual users interact with visual elements without being aware of all the available features and their relationships to underlying concepts. As above-mentioned SSBI is a simplified version of BI, but it does not decline important ideas necessary for efficient and effective solutions and results. Such question highlights the fact that in practice, implementing SSBI is not as easy as expected (Lennerholt, Van Laere, & Söderström, 2021).

Also, traditional BI methodologies (Kimball, 2016; Inmon, 2006) already contemplate briefly the development of dashboards and data communication. However, since they were developed with a focus on data integration and building a centralized repository, they are not as oriented toward the latest developments in data visualization and storytelling. Dashboard development and data storytelling are gaining attention but generally as separated approaches with their own development cycles (Suprata, 2019; Zhang, et al. 2022).

The Kimball methodology is a widely used bottom-up approach that remains relevant for developing BI systems. The methodology was developed in the 80s and it is considered a guide for experts. On the other hand, data storytelling in the context of data visualization is gaining momentum, and it is not fully considered in that methodology. Kimball and Inmon are well known authors of approaches to development BI. They gave importance to aspects such as data integration and to the develop of centralized sources. But cloud computing changed the paradigm and cloud providers currently enable to store enormous data volumes in structured, unstructured or semi-structured formats. Data may be stored in the cloud providers supported by databases, data warehouses or data lakes. An SSBI can consume data from those cloud platforms at any given time and as needed. Despite, many organizations struggle to utilize the potential of SSBI and experience implementation challenges (Lennerholt, Van Laere, & Söderström, 2018).

Nevertheless, data exploration by stakeholders is essential, it may happen a BI solution is deployed, and data specialists provide models without communicating potential data insights. At other times, data specialists supply only a set of charts that may be more or less impactful.

The dashboards and data storytelling stages are gaining attention since when well-done discoveries are communicated effectively as they transmit and highlight insights. As part of the project, effective communication of results should be combined with contextualized narratives for guidance and higher quality insights. Although stakeholders have rich background knowledge, they may not understand how to sculp the data, how to model the data, how to organize layouts and communicate insights effectively. Also, to access and use several data sources for analysis and decision-making is not easy as expected and different challenges arise for SSBI (Alpar & Schulz, 2016; Lennerholt, Van Laere, & Söderström, 2021). It requires technical skills that not all users possess, such as data cleaning, data modelling, knowledge about layouts arrangements and choosing the right charts, among others.

It is considered essential that users with background knowledge in a subject and who wish to analyse data themselves acquire concepts of BI to develop SSBI. In this way, more proactive SSBI projects can be built with more data quality and more impactful decisions. Therefore, a symbiosis between casual users and power users is fundamental, as their roles complement each other - power users with technical skills and casual users with background knowledge about the case under study.

In SSBI the casual users should have minimum knowledge such as capacities to connect to the data sources, clean and transform the data, build a data model, choose the right charts and build dedicated layouts and communicate the data stories. As so casual users should be promoted to a role designated by casual power users. These are users with background knowledge regarding the case study and some technical and theoretical expertise about BI.

Also recently, the paradigm started to be on data visualization and data communication. How can traditional BI methodologies be integrated with the dashboards and data storytelling approaches for easy and flexible data consumption and in a light but stricter manner?

4 THE FRAMEWORK

Authors propose to build a framework to help casual power users to the development more quality SSBI solutions.

4.1 Specification

The framework has 5 constraints each specify the minimum knowledge requirements that they should possess:

1. Knowledge about requirements.
2. Knowledge about modelling.
3. Knowledge about data integration.
4. Knowledge about data visualization.
5. Knowledge about data storytelling.

The constraints have subitems, some of which are interrelated and dependent on each other. The interrelated subitems are developed together due to their interdependence. For instance, the requirements constraint is closely linked to the data visualization and data storytelling constraints. When identifying requirements, is necessary to establish the audience, specify performance indicators, and design the dashboard layouts and charts to be used. Conversely, during dashboard layout design, new requirements may also emerge. Similarly, the integration constraint is strongly connected to the data modelling constraint, as integration, for example, is not feasible without identifying the metadata. Below, the main stages and their subitems are outlined.

In the **knowledge about requirements constraints** casual power users grasp both the context and the audience, along with their profiles. The discovery of relevant questions is crucial, as they influence problem comprehension, leading to problem resolution. For instance, the Specific, Measurable, Achievable, Relevant, and Time-Bound (SMART) criteria (Doran, 1981) may be applied. The goal is to develop sets of relevant questions and to identify performance indicators, as these enable the organization to monitor and control its operations. Every organization has a specific strategy to achieve certain goals and uses these indicators as references for decision-making (Balon, 2024). On the other hand, design the dashboard *mockups* to organize previously the data to display.

In the **knowledge about the modelling constraints** casual power users discover the data source metadata and design an appropriate model for data exploration. Kimball (2016) proposes an approach for designing a multidimensional model. This model is considered significant because it allows flexible combinations when querying the data. The dimensional model and the associated star schema enhance data exploration capabilities, providing the system with a powerful mechanism for data analysis.

The dimensional model is expressed by fact tables surrounded by dimension tables. The fact tables store business measures, while the dimension tables contain axes of analysis describing measures. The Kimball process has the following stages (Kimball, 2016): select the business processes, declare the grain, identify the dimensions, identify the facts, design the star schema, define the data, handle slowly changed dimensions, implement and test the model, iterate and refine.

In the **knowledge about data integration constraints,** it is important to understand how to connect to a diverse group of data sources and how to clean and transform the data. Many SSBI tools come equipped with features that allow connection to various types of sources, along with easy-to-use data cleaning and transformation capabilities.

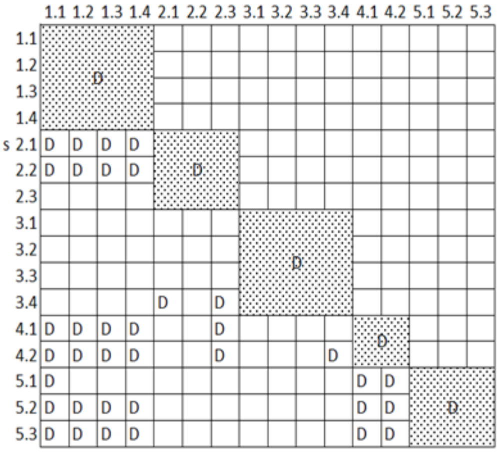
In the **knowledge about data visualization constraints,** casual power users implement the layouts designed in the first stage, utilizing the identified layouts and charts to convey the appropriate messages.

In the **knowledge about data storytelling constraints,** the narratives are built by identifying the most appropriate episodes for each taking into considerations the audience previously identified. They then express these narratives using guiding threads.

4.2 SSBI Framework

In this section the subitems of each constraint are described.

1. **SSBI_KR** Knowledge about requirements
   - 1.1. Identify the context and the audience
   - 1.2. Identify analysis questions
   - 1.3. Describe performance indicators
   - 1.4. Gather functional requirements
2. **SBI_KM** Knowledge about modelling
   - 2.1. Identify metadata from data sources
   - 2.2. Build a dimensional model
   - 2.3. Implement the dimensional model
3. **SSBI_KI** Knowledge about data integration:
   - 3.1. Connect to data sources
   - 3.2. Infer the data profile
   - 3.3. Clean and transform the data
   - 3.4. Load the data tables (dimensions and facts)
4. **SSBI_KV** Knowledge about data visualization:
   - 4.1. Design the layouts and identify the best charts in agreement with the data visualization objectives
   - 4.2. Implement the dashboards
5. **SSBI_KS** Knowledge about data storytelling
   - 5.1. Identify the context
   - 5.2. Identify narratives
   - 5.3. Build narratives guiding threads



	1.1	1.2	1.3	1.4	2.1	2.2	2.3	3.1	3.2	3.3	3.4	4.1	4.2	5.1	5.2	5.3
1.1	D															
1.2																
1.3																
1.4																
s 2.1	D	D	D	D	D											
2.2	D	D	D	D												
2.3																
3.1								D								
3.2																
3.3																
3.4					D		D									
4.1	D	D	D	D			D					D				
4.2	D	D	D	D			D				D					
5.1	D											D	D	D		
5.2	D	D	D	D								D	D			
5.3	D	D	D	D								D	D			

Figure 1: Dependence between subitems.

Figure 1 highlights the dependence between its subitems, and Figure 2 displays the graphic representation of the framework. In the framework there are high dependences between the requirements phase and the data visualization and data storytelling phases. In the data visualization dashboards are built as they serve as a support to the data storytelling phase. As so they influence each other.

5 CASE STUDY: CLASS ATTENDANCE

The framework was utilized by a casual power user and applied to a case study in a higher education institution. The institution needs to make decisions regarding the management of the classes. The casual power user is a manager aware of the problem under analysis and uses a SSBI tool (Power BI). Next, the problem is contextualized, and later, the framework is applied to solve the problem.

5.1 Contextualization

A higher education institution has a transactional and operational digital platform to control class attendance. In the school, each course has a set of subjects with enrolled students. Students are divided into groups. A class is a lesson conducted by a teacher for a group of students. After each class the teachers register the number of attending students in the digital platform. There is a need for a data-driven solution to monitor student attendance throughout the semester.

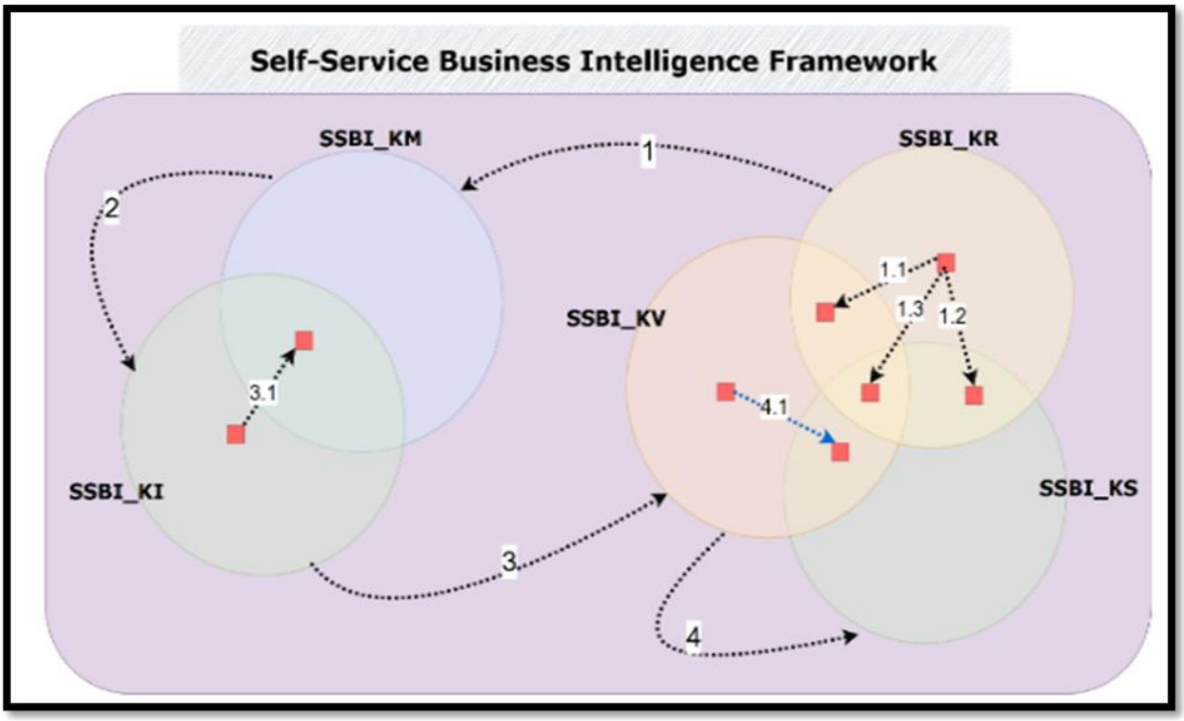


Figure 2: The SSBI framework with interconnected stages.

It has two main objectives: the first is to discourage school dropouts and prevent early course withdrawals, while the second is to identify attendance patterns to support decisions about rearranging classes for students over the semesters. In some cases, classes may have low or high attendance. Low attendance is undesirable, as resources are underutilizing while high attendance is not pedagogically effective.

The objective is to build a small-scale SSBI system to support decision-making regarding the classes rearrangement.

5.2 Applying the Framework

5.2.1 Requirements

In initial meetings with other project sponsors the objectives of the project were identified together with the questions to respond. The project sponsors are the managers who also have the responsibility to rearrange the groups of students. The questions were formulated using the SMART criteria (Table 1).

Table 1: For managers.

Q1	What is the number of courses, subjects, teachers, and enrolled students in the current academic year to assess resource allocation?
Q2	How many students are attending each class over the weeks during the current academic year to help determine necessary adjustments?
Q3	Which classes have the highest and lowest attendance rates over the weeks and may require rearrangement?
Q4	What is the attendance rate for all the classes of a specific subject, both daily and weekly?
Q5	What is the impact on the attendance rate of academic events?

The project main objective is to analyse the student's attendance evolution over the weeks and identify classes that need to be rearranged, more concretely, classes to be closed and classes to split.

5.2.2 Data Modelling

A star model was built resulting from the steps early mentioned in section 4.1. The casual power user contextualize itself with the approach.

The casual power user background knowledge with the problem has facilitate the modelling phase, since the data and the terminology are well-known. The dimensions such as teachers, courses, subjects and classes were identified. Additionally, it was recognized the need for the academic date dimension since the it was considered important to observe the impact of some academic events in the students' class attendance behaviour. The following facts were identified: attendance and enrolment.

Table 2 presents the relationship between the facts and dimensions. In Figure 3 the achieved data model may be analysed.

Table 2: Relationships between facts and dimensions.

	Subjects and Courses Enrolment	Classes Attendance
Course	X	X
Subject	X	X
Teacher		X
Date		X
Academic Date	X	X

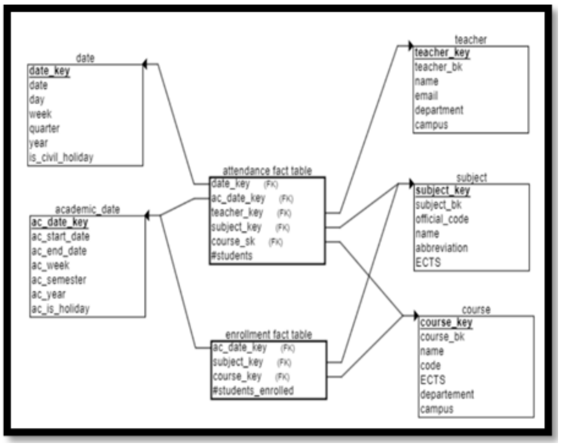


Figure 3: The star schema.

5.2.3 Data Integration

The data sources are consumed using RESTful Web Services. After it, data is visually shaped (cleaned, integrated and transformed) step by step. The result is a group of connected tables. In PowerBI both Power Query and Data Analysis eXpressions (DAX) are used to perform the ETL process. The first to load and accomplish initial transformations and DAX for additional operations and to load data to the final dimension and fact tables.

5.2.4 Data Visualization

All at all dashboards classified as tactical have been developed for decisions. Despite charts being generally used daily it was considered a challenge the identification of the most suitable charts for data display.

5.2.5 Data Storytelling

The narratives were elected. For instances, the narrative of the 4th week semester was told since in that period the identification of classes attendance is a main concern (since it is still the beginning of the semester). The narrative starts to contextualize the courses, the subjects and teachers. Then it highlights the classes with the lowest and highest number of students. The narrative was build using elected frames extracted from dashboards. Additionally, the frames were organized, and context was assigned creating a movie. The audience in this case were the managers.

6 CONSIDERATIONS

The most expensive tasks that the casual power user reported was the development of the dimensional data model and the development of dashboards with an effective layout. However, later the dimensional model was considered fundamental to support data combination and data exploration. Nevertheless, the familiarization with the problem in analysis also has contributed to assist in the development of that model. Additionally, there has been reported some versions of *mockups*. Thinking about the layout and their visual elements was considered fundamental to a more effective design. The casual power user established the requirements and design the layouts in conjunction. In modelling as the data was gathered and its profile obtained from the web services the dimensional model was elaborated. The developed solution enables the understanding of class attendance behaviour during an academic semester, facilitating necessary adjustments. By the 4th week, the casual power users could identify classes to merge and classes to split. The custom solution developed by its own is now capable of telling him and others about the need of changes. Casual power user stated that the model with the appropriate connections were a main resource to build a set of filter segmentation components and to identify the rate attendance in classes.

7 CONCLUSIONS

SSBI aims to enable the agile development of BI solutions using low-code features and facilitating the creation of custom data-driven systems. For casual users, this is useful, as they are the ones who best understand the primary objectives of the analysis and may what to independently build a personalized solution. However, it has been reported that some poor solutions have resulted from these users' lack of conceptual understanding. Although SSBI tools provide visual artifacts, there is still a need to know the underlying concepts. This work introduces the role of casual power users: individuals who are familiar with the case study and have a general understanding of BI concepts. The authors present a framework with five interconnected knowledge constraints for developing SSBI solutions. These constraints were applied to a case study conducted by a casual power user, who uncover insights for decision-making. In the future, the authors plan to apply the framework to additional case studies.

ACKNOWLEDGEMENTS

This work was supported by the Fundação para a Ciência e a Tecnologia (FCT), I.P., under Project UIDB/04524/2020.

REFERENCES

Sharda, R., Delen, D., & Turban, E. (2018). *Business intelligence, analytics, and data science: a managerial perspective*. pearson.

Evelson, B., & Norman, N. (2008). Topic overview: Business intelligence. *Forrester research, 61*.

Lennerholt, C., Van Laere, J., & Söderström, E. (2018). Implementation challenges of self-service business intelligence: A literature review.

Microsoft. (2024). *Microsoft*. Retrieved July 15, 2024, from *Power BI*. https://powerbi.microsoft.com/

Tableau Software. (2024). *Tableau* Retrieved July 15, 2024, from . https://www.tableau.com/

Qlik. (2024). *Qlik*. Retrieved July 15, 2024, from https://www.qlik.com/

Arnaboldi, M., Robbiani, A., & Carlucci, P. (2021). On the relevance of self-service business intelligence to university management. *Journal of Accounting & Organizational Change*, *17*(1), 5-22.

Suprata, F. (2019). Data storytelling with dashboard: accelerating understanding through data visualization in financial technology company case study. *Jurnal Metris*, *20*(01), 1-10.

Kimball, R., & Ross, M. (2016). *The kimball group reader: Relentlessly practical tools for data warehousing and business intelligence remastered collection*. John Wiley & Sons

Zaghloul, M. M., Ali-Eldin, A., & Salem, M. (2013). Towards a self-service data analytics framework. *International Journal of Computer Applications*, *80*(9).

Lennerholt, C., Van Laere, J., & Söderström, E. (2021). User-related challenges of self-service business intelligence. *Information Systems Management*, *38*(4), 309-323.

Dedić, N., & Stanier, C. (2017). Measuring the success of changes to Business Intelligence solutions to improve Business Intelligence reporting. *Journal of Management Analytics*, *4*(2), 130-144.

Alpar, P., & Schulz, M. (2016). Self-service business intelligence. *Business & Information Systems Engineering*, *58*, 151-155.

Inmon, B. (2006). DW 2.0; Architecture for the Next Generation of Data Warehousing. *Information Management*, *16*(4), 8.

Olaoye, F., & Potter, K. (2024). *Business Intelligence (BI) and Analytics Software: Empowering Data-Driven Decision-Making* (No. 12550). EasyChair.

Shneiderman, B. (2003). The eyes have it: A task by data type taxonomy for information visualizations. In *The craft of information visualization* (pp. 364-371). Morgan Kaufmann.

Chokki, A. P., Simonofski, A., Frénay, B., & Vanderose, B. (2022). Engaging citizens with open government data: The value of dashboards compared to individual visualizations. *Digital Government: Research and Practice*, *3*(3), 1-20.

Zheng, J. G. (2017). Data visualization in business intelligence. In *Global business intelligence* (pp. 67-81). Routledge.

Doran, G. T. (1981). There's a SMART way to write managements's goals and objectives. *Management review*, *70*(11).

Balon, U. (2024) KEY PERFORMANCE INDICATORS (KPIs) IN THE QUALITY MANAGEMENT SYSTEM. *International Journal for Quality Research*, *18*(2), 473-486.

Few, S. (2006). *Information dashboard design: The effective visual communication of data*. O'Reilly Media, Inc.

Schwendimann, B. A., Rodriguez-Triana, M. J., Vozniuk, A., Prieto, L. P., Boroujeni, M. S., Holzer, A., ... & Dillenbourg, P. (2016). Perceiving learning at a glance: A systematic literature review of learning dashboard research. *IEEE transactions on learning technologies*, *10*(1), 30-41.

Shneiderman, B. (2003). The eyes have it: A task by data type taxonomy for information visualizations. *In The craft of information visualization* (pp. 364-371). Morgan Kaufmann.

Sorour, A., & Atkins, A. S. (2024). Big data challenge for monitoring quality in higher education institutions using business intelligence dashboards. *Journal of Electronic Science and Technology*, 100233.

Zhang, Y., Reynolds, M., Lugmayr, A., Damjanov, K., & Hassan, G. M. (2022, September). A visual data storytelling framework. In *Informatics* (Vol. 9, No. 4, p. 73). MDPI.

Dykes, B. (2019). Effective data storytelling: how to drive change with data, narrative and visuals. *John Wiley & Sons*.

Knowledge Graphs Can Play Together: Addressing Knowledge Graph Alignment from Ontologies in the Biomedical Domain

Hanna Abi Akl[1,2][a], Dominique Mariko[3], Yann-Alan Pilatte[3], Stéphane Durfort[3], Nesrine Yahiaoui[3] and Anubhav Gupta[3]

[1] *Data ScienceTech Institute (DSTI), 4 Rue de la Collégiale, 75005 Paris, France*
[2] *Université Côte d'Azur, Inria, CNRS, I3S, France*
[3] *Yseop, 4 Rue de Penthièvre, 75008 Paris, France*
hanna.abi-akl@dsti.institute, {dmariko, ypilatte, sdurfort, nyahiaoui, agupta}@yseop.com

Keywords: Knowledge Graphs, Ontologies, Natural Language Processing, Information Extraction.

Abstract: We introduce DomainKnowledge, a system that leverages a pipeline for triple extraction from natural text and domain-specific ontologies leading to knowledge graph construction. We also address the challenge of aligning text-extracted and ontology-based knowledge graphs using the biomedical domain as use case. Finally, we derive graph metrics to evaluate the effectiveness of our system compared to a human baseline.

1 INTRODUCTION

In the era of Large Language Models (LLMs), Knowledge Graphs (KGs) have resurfaced to play an important role, whether as complements to LLM-based technology to enhance predictions in Retrieval Augmented Generation (RAG) models, or as standalone systems that more faithfully capture factual information (Pan et al., 2023b), (Peng et al., 2023), (Vogt et al., 2022). The ongoing problem of hallucinations in LLMs draws a line on their reliability and questions the interpretability and explainability of their outputs. The inability to trust the responses of these deep learning models leads to much hesitation in implementing and deploying them in production, especially in sensitive domains such as healthcare (Pan et al., 2023a).

KGs, on the other hand, have demonstrated their staying power by circumventing the black-box mechanism of LLMs and offering open and traceable representations of domain information (Pan et al., 2023a). Their staying power is also strengthened by their integration with both deep learning solutions and more classical frameworks like ontologies that provide formal representations of knowledge (Pan et al., 2023b), (Vogt et al., 2022). A major weakness they exhibit however is their difficulty in integrating and aligning new knowledge. Unlike LLMs, which benefit from fine-tuning to add new knowledge, ontologies, and KGs by extension, require a lot of work in order to enrich their representations in a single domain or extend to a new one (Van Tong et al., 2021). This weakness makes these technologies less transferable on their own which is why they are often utilized as components in larger systems that can benefit from their advantages (Peng et al., 2023).

In this work, we present DomainKnowledge, a system comprised of a workflow of information extraction (IE) from unstructured text leading to the construction of a consolidated domain KG. We showcase strategies in our implementation to combine domain knowledge from ontological sources and amount to a generalized domain-specific KG mapping input text entities to higher-order concepts. We also introduce metrics inspired from graph theory to evaluate our system. The rest of the work is structured as follows. Section 2 presents related work in the literature. Section 3 describes our methodology. In section 4 we present our experimental setup. Section 5 discusses our findings with an analysis of our results. Finally, we conclude with future directions of work in section 6.

2 RELATED WORK

This section covers the literature pertaining to text-to-graph extraction techniques as well as KG alignment methods.

[a] https://orcid.org/0000-0001-9829-7401

Abi Akl, H., Mariko, D., Pilatte, Y-A., Durfort, S., Yahiaoui, N. and Gupta, A.
Knowledge Graphs Can Play Together: Addressing Knowledge Graph Alignment from Ontologies in the Biomedical Domain.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 416-423
ISBN: 978-989-758-716-0; ISSN: 2184-3228

2.1 Knowledge Graph Construction from Text

Research in IE shows different methods to construct KGs from text. In their work, (Liu et al., 2022) surveyed different methods for text information extraction from relation triples. They explored and compared systems that capture relations in the form of triples, spans and clusters using symbolic and deep learning techniques at the syntactic and semantic levels. (Kamp et al., 2023) compared rule-based open IE engines to machine learning extraction systems and found a trade-off between implementation and precision. While rule-based systems exhibited better overall performance in identifying and extracting relations, they were much harder and more exhaustive to implement than off-the-shelf machine learning models.

Natural language processing (NLP) methods like sentence chunking, domain entity classification, relation classification and sentence-to-graph techniques to manipulate text directly through graph properties have achieved promising results that exploit syntactic and semantic text attributes through models built on robust rule engines. These techniques, while capable of controlling the type of information to extract, have shown limitations when it comes to extending them to cover more exhaustive knowledge (Chouham et al., 2023), (Dong et al., 2023), (Motger and Franch, 2024), (Yu et al., 2022). Other research geared toward machine and deep learning technology combines these methods with classical NLP techniques for better results. In their work, (Qian et al., 2023) proposed an IE pipeline that combines pattern-based, machine learning and LLM extractions that undergo rule-based and machine learning scoring to decide on keeping or discarding extracted information. Transformer-based approaches have also been applied to leverage embeddings information and transform them to node properties in graphs constructed from text (Friedman et al., 2022), (Melnyk et al., 2022). These methods have showcased a better ability at capturing text properties as node representations. Novel hybrid systems making use of the availability of LLM technology leveraged their prompting abilities to provide domain annotations for better information extraction (Dunn et al., 2022), combine them with other sources of knowledge like ontologies (Mihindukulasooriya et al., 2023), (Wadhwa et al., 2023) for better coverage, and even use text generation techniques as a comparative benchmark to identify viable relation candidates for extraction (Hong et al., 2024).

2.2 Knowledge Graph Alignment

Several research avenues explore graph-based techniques for KG alignment. (Zeng et al., 2021) survey distance-based and semantic matching scores for effective entity alignment in KGs. In their work, (Zhang et al., 2021) propose systems based on stacked graph embeddings of different graph components like neighboring entities and predicates to improve entity alignment. Other methods focus on integrating deep learning models to better express graph component properties and answer the graph alignment problem. (Chaurasiya et al., 2022), (Dao et al., 2023) and (Fanourakis et al., 2023) show that graph neural networks performed well in aligning different graph entities when paired with distance-based graph features and embeddings. In their work, (Yang et al., 2024) show that LLMs could be leveraged to decompose the alignment problem into multiple choice questions referring to sub-tasks to approximate the alignment of entities with respect to neighboring nodes. (Trisedya et al., 2023) propose a system composed of an attribute aggregator and a node aggregator to combine both node and relation properties and get better alignment predictions. (Zhang et al., 2023) showcase a similar method aggregating property, relationship and attribute triples to get a more complete representation of entities and aid the entity alignment process.

Finally, neuro-symbolic systems aiming to combine both classical rule-based techniques with sub-symbolic architectures have also been proposed to tackle the graph alignment problem. (Cotovio et al., 2023) survey neural network architectures and reinforcement learning methods for better entity alignment predictions. In their work, (Xie et al., 2023) convert different KGs into vector space embeddings and combine them with graph neural networks to create transitions and better delimit the best alignment for a node entity. (Abi Akl, 2023) show the benefits of using logic neural networks as reasoners with a rule-set derived from upper ontologies in a hybrid system to align entities from different KGs.

3 METHODOLOGY

The DomainKnowledge system proposes a data acquisition and transformation pipeline that leverages NLP and graph techniques to extract meaningful relationships from raw text and store them in graph structures to create a domain vocabulary. It consists of the following components:

- An IE pipeline which handles the relationship extraction. The IE component depends on the document or text extraction process that precedes it, which should be capable of extracting raw text and transforming it into a list of sentences, since the IE pipeline identifies relationships at sentence level.

- A knowledge storage system which references the graph database storage and KG construction.

The system workflow can be summarized in the following steps:

- Initiate a generic pipeline to identify and extract relations from raw text
- Define a ruleset for meaningful relations
- Prune relations to conserve only meaningful ones
- Export relations into semantic graph structures
- Generate the domain vocabulary from graph relationships

3.1 System Overview

The user provides a number of documents from the same domain (e.g., Pharmaceutical). The documents are processed one by one as raw texts. The DomainKnowledge pipeline analyzes the texts as sentences and extracts relationships as triples of the form *(subject,relation,object)*. Relationships are identified with the help of a domain ontology that emphasizes important domain words to look out for, e.g., MedDRA for Pharmaceutical. Once relationships are extracted, a set of rules is applied to prune the bad ones. These rules can vary from simple, e.g., eliminating relationships with missing elements in the triples, to more complex, e.g., evaluating the nature of the relation like verbal versus non-verbal. The ruleset can also be aided by the reference ontology to drop relationships that contain no relevant terms in the subject and/or object entities of the triple. The relationship matrices are then concatenated into one matrix containing all the relevant relationships from all the documents. The matrix is then formatted into several files and exported in a way to preserve the following information:

- Each relationship is unique and is assigned a unique identifier
- Each relationship triple has a clear subject, predicate and object
- Each relationship clearly references the sentence it is extracted from
- Repeated triples are kept
- Each relationship clearly references the document it is extracted from
- Each document is unique and is assigned a unique identifier

The exported information is then ingested into a graph database that conserves the above-mentioned information in a graph network. The graph network is modeled as a subject/object node KG where nodes are subject and object entities and edges are the relation of the triple. Each node has properties associated with it like its unique identifier, the sentence it is extracted from, the name of the document it is extracted from, the unique identifier of the document, the type of the document (e.g., Clinical Study Report, Protocol) and the domain of the document (e.g., Pharmaceutical). Figure 1 shows the high-level architecture of our system.

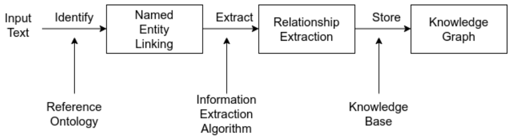


Figure 1: DomainKnowledge pipeline.

3.2 System Modules

3.2.1 Extractor

A plain text extractor keeping document layout based on MuPDF[1] in Python.

3.2.2 Annotator

The Annotator's output is based on Stanza's[2] dependency parser which provides a standardized way of representing syntactic dependencies between words in a sentence. Our system produces relations from texts of documents using specific dependencies appearing in Stanza's output. Two main relation types were considered for extraction:

1. Verbal Relations: canonical verbal relations take a verb as a cornerstone to build a triple (entity, verb, entity) which can be transformed to (subject, relation, object). The relations are described as follows:
    - *root:* the root of the sentence should usually be the verb that is the main predicate. The root usually has subject(s) and object(s), unless it is intransitive or another verbal dependency interferes.
    - *acl:* behave like roots, but their subject already has a dependency link to another verb (typically, as an object of the root, but not only).

[1] https://shorturl.at/WXmwU

[2] https://stanfordnlp.github.io/stanza/

- *acl:relcl:* adnominal relative clause introduced by relative pronouns, which can either be their subject or object, and reference another subject or object in the context.
- *advcl:* adverbial clauses can have their own subjects and objects, in which case they behave like roots. If they modify nouns and have no subject, they are linked to the verb they modify and its subject.

2. Prepositional Relations: we use OpenIE[3], an open-source relation extraction tool, to build prepositional relations from adpositions (e.g., 'as', 'with', 'for', etc.). Considering prepositions as the pivot of the relation, subjects and objects of verbal relations are split into smaller pieces and can match better with ontology terms. We use the dependency tag of the object entities of these relations to identify them with the *nmod* tag.

The Annotator module is also in charge of constructing triples. Each sentence in the original text is decomposed into entity-relation triples and stored with metadata attributes such as document ID, section ID (from the document layout), sentence ID, tokens positions and tokens POS tags. The triples are sets of nodes and relations to be compared with the value of the string data type available in the UMLS metathesaurus[4]. Figure 2 shows the annotation logic.

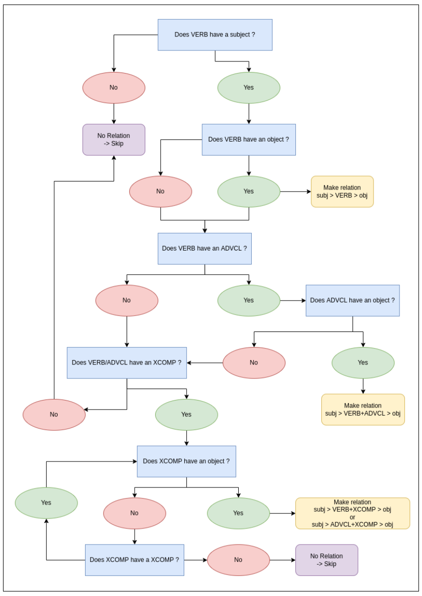

Does VERB have a subject ?
No
Yes
No Relation -> Skip
Does VERB have an object ?
No
Yes
Make relation subj > VERB > obj
Does VERB have an ADVCL ?
No
Yes
Does ADVCL have an object ?
Does VERB/ADVCL have an XCOMP ?
No
Yes
Make relation subj > VERB+ADVCL > obj
No
Yes
Does XCOMP have an object ?
Yes
No
Yes
Make relation subj > VERB+XCOMP > obj or subj > ADVCL+XCOMP > obj
Does XCOMP have a XCOMP ?
No
No Relation -> Skip

Figure 2: Annotation flowchart.

3.2.3 Aggregator

The Aggregator relies on the National Library of Medicine Unified Medical Language System (UMLS® 2022AA) release. We consider the MRCONSO, MRSAB, MRSTY and MRREL data tables and reorganize their content into a graph data model. We follow the UMLS data types as described in the UMLS Metathesaurus Rich Release Format[5] and keep the data objects as nodes in the graph data model. The data model consists of the following nodes:

- AUI: atom
- CUI: concept
- LUI: term
- SUI: unique string
- TUI: semantic type

We preserve the relationships attributes as defined in the original UMLS Metathesaurus[6]. We turn the incoming relationships into direct links to find paths between text NER nodes and UMLS nodes:

- CUI node has an atom node: $CUI \xrightarrow{HAS\_AUI} AUI$
- SUI node has an atom node: $SUI \xrightarrow{HAS\_AUI} AUI$
- SUI node has concept node: $SUI \xrightarrow{HAS\_CUI} CUI$
- CUI node has semantic type node: $CUI \xrightarrow{HAS\_STY} TUI$

The resulting data model is available in Figure 3.

3.2.4 Merger

Outputs from the Annotator, i.e., entity-relation triples, and the Aggregator, i.e., SUI objects, are mapped with measures of semantic similarity using the following algorithm:

- An exact matching measure using the Levenshtein distance to compute a first similarity score.
- A semantic matching algorithm using cosine similarity to compute a more refined evaluation of entities that do not score highly on the exact matching: each entity is mapped to a 512 dimensional dense vector space, so the semantic matching algorithm can draw similarities from the generated vectors to find associations between two entities.

An additional Named Entity Recognition tagger, i2b2[7], is used to map long triples entities and SUI objects to augment the text-to-ontology mapping. Subject and object entities declared in extracted triples

[3] https://shorturl.at/2VNh0

[4] https://shorturl.at/5F0P8

[5] https://shorturl.at/2HYBj

[6] https://shorturl.at/iV97e

[7] https://shorturl.at/aMN80

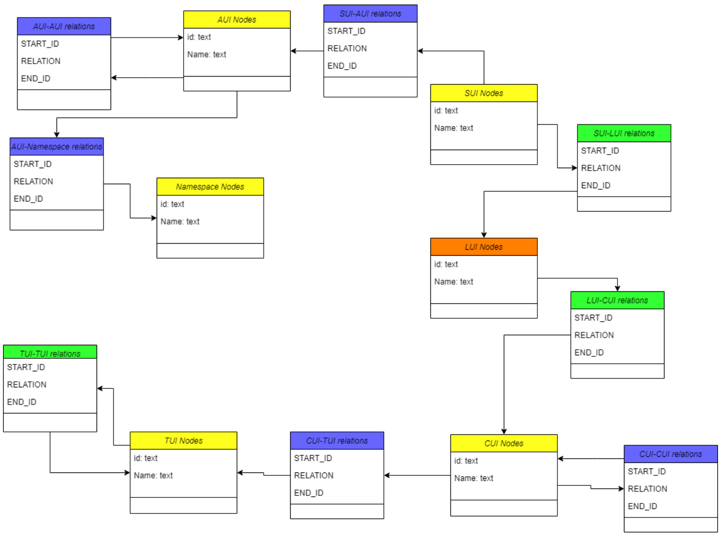


Figure 3: Graph data model.

from sentences are declared as NER nodes in the constructed KG. The Merger outputs a KG construction from a set of pre-configured semantic graphs, consistent with the graph data model, adding the following nodes and edges to the graph:

- Text node linked to another text node: $NER \xrightarrow{TEXT\_LINK} NER$
- Text node matched to SUI node: $NER \xrightarrow{HAS\_LEXICAL} SUI$

  An example graph is presented in Figure 4.

3.3 Metrics

We define the following evaluation metrics:

- Coverage (CVRG). Let *DT* be the set of domain tokens, i.e., any extracted entity from a given text that is also linked, i.e., sharing a direct relation in our KG, to an ontological concept from the domain. Let *TT* be the set of text tokens, i.e., any extracted entity from the same text. The Coverage is defined as

$$\frac{|DT|}{|TT|} \times 100 \tag{1}$$

- Mapping (MAPG). Let *CT* be the set of concept tokens, i.e., any extracted entity from a given text sharing the same syntactic (and semantic) name as an ontological concept from the domain. The Mapping is defined as

$$\frac{|CT|}{|DT|} \times 100 \tag{2}$$

- Alignment (ALGT). Let $r_{NER \to TUI}$ be a direct link from any extracted entity (NER) from a given text to an ontological semantic type (TUI). Let r_{TUI} be a link from any source node to a TUI node, i.e., $r_{TUI} = r_{NER \to TUI} + r_{CUI \to TUI}$. The Alignment is defined as

$$\frac{count(r_{NER \to TUI})}{count(r_{TUI})} \times 100 \tag{3}$$

4 EXPERIMENTS

Experiments were performed on 52 Clinical Study Reports (CSR) with the objective of finding direct relationships between text entities, i.e., subject or object of a triple (NER nodes), and ontological concepts (CUI nodes) and semantic types (TUI nodes). We perform two experiments, each testing an algorithmic approach using the DomainKnowledge pipeline to obtain an alignment from NER nodes to TUI nodes. All experiments were hosted on an instance of Neo4j AuraDB[8]. The first experiment focuses on building sentence clusters based on a sentence similarity score calculated from the triples forming the sentences. The intuition is that similar sentences will very likely be paraphrases or rewording and will trace back to the same higher-order ontological concepts. Grounding these concepts makes the task of aligning NER and TUI nodes easier. The experiment can be broken down to the following steps:

1. The node2vec[9] embeddings is calculated for every NER node.

[8] https://tinyurl.com/yzxneyy5

[9] https://tinyurl.com/55jc525f

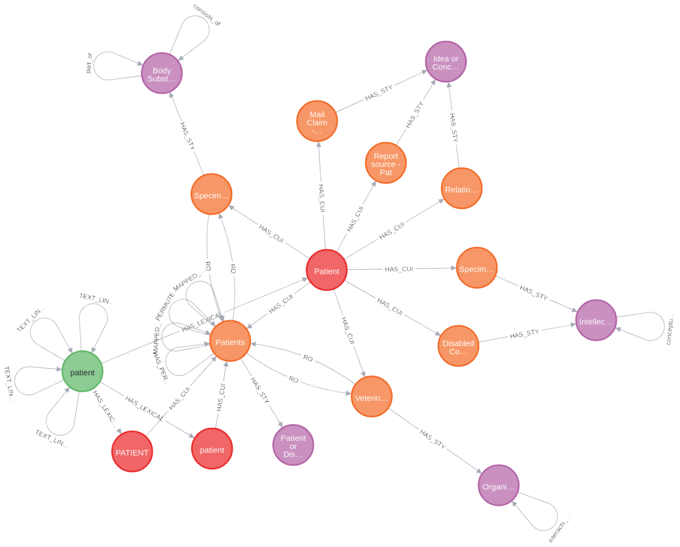


Figure 4: Sample output graph.

2. A K-Nearest Neighbors (KNN)[10] clustering algorithm is used to create pairwise clusters of NER nodes using the node2vec embeddings as property.
3. The resulting KNN similarity score *knn_score* for each pair of NER nodes is appended to the relation in their triple if and only if they share a triple.
4. We define the sentence score for a sentence as
$$sentence\_score = \sum_{i=1}^{n} knn\_score_i \tag{4}$$
where n is the number of relations of the triples extracted from the sentence. All sentences are compared and grouped based on the sentence_score. Sentences with equal sentence_scores underline similar sub-graphs from NER to TUI nodes.

The final step is extracting the relevant NER and TUI nodes from the different sentence group sub-graphs. While this experiment shows promising results on a small batch of sentences, we lacked the resources to handle the computational complexity of the procedure on our ensemble of documents. We therefore did not report results for this method. The second experiment targets ontology alignment directly using the paths between NER, CUI and TUI nodes. The experimental setup is as follows:

1. The degree centrality DC[11] measure is calculated for every NER, SUI, CUI and TUI node. Relations between SUI and AUI nodes are also considered for the SUI degree centrality calculation as they are considered additional information on the representation of a concept. The calculations are based on the following directed graph orientations:
   - $NER \longrightarrow SUI \longrightarrow AUI$ (a)
   - $NER \longrightarrow SUI \longrightarrow CUI \longrightarrow TUI$ (b)

   The aim of this measure is to identify popular nodes.
2. we define the weight w of a relation between 2 nodes A and B as the sum of their degree centralities DC_A and DC_B respectively. Formally, $w_{AB} = w_{BA} = w = DC_A + DC_B$.
3. For each NER node, we traverse the closed sub-graphs respecting the path in (b) while opting for the maximum total weight
$$W = \sum_{i=1}^{n} w_i \tag{5}$$
where n is the total number of relations between a NER node and a CUI node in a closed sub-graph.
4. We apply the same traversal algorithm to identify the best direct relation between a CUI node and a TUI node.
5. We finally use the results from the previous two steps to find the best direct relation between a NER node and a TUI node.

We evaluate our DomainKnowledge pipeline against a human baseline consisting of clinical analysts from the biomedical domain who manually perform the alignment on the same dataset and report our results.

[10] https://tinyurl.com/yzx7dxv9
[11] https://tinyurl.com/bddhh9e7

5 RESULTS

Of the 7407 extracted sentences over all documents, a total of 172836 tokens were identified. From these tokens, 131625 were relevant domain tokens covered in triples, representing 76.16% of text coverage into triples. To ensure these triples are viable, the relations binding subject and object tokens had to be either verbal or prepositional to rule out unusable triples. 16051 verbal or prepositional relations were extracted over the text, which resulted in 13821 unique triples representing approximately 10.50% of the total set of extracted triples. This figure signifies that domain concepts make up roughly 10% of a CSR, whereas the remaining 90% are the different context windows in which the domain vocabulary is used. A summary of the triple extraction process from our pipeline is detailed in Table 1. From the extracted triples, 4002 NER objects are domain vocabulary that can be mapped to UMLS concepts. An additional 3417 objects tagger by the NER tagger means a total of 53,67% of the extracted triple objects can be mapped to UMLS concepts. The final KG yields 7151 indirect links from NER to TUI nodes. Indirect links encompass any direct link from NER to CUI ot NER to TUI directly. The calculations from our graph traversal algorithm identify 1533 direct links from NER to TUI, resulting in an alignment of 21,40%. Table 2 shows the details of the NER node alignment to the domain ontology. Table 3 shows the performance of our pipeline with respect to the human baseline. The results show promise for our pipeline: it beats the human baseline on all metrics while retaining a good domain coverage of the text. The mapping score indicates the over half the extracted triples contains pertinent nodes that can be traced back to the domain ontology, showcasing the effectiveness of our annotation and extraction methods. The alignment score, while relatively low, is encouraging when it comes to finding higher-level concepts linked to the initial document text. This opens the possibility to a wider integration between domain ontologies and domain texts, with potential possibilities to enhance the latter with the former using the links between NER and TUI to semi-automatically generate in-context text templates and enrich the document. It is worth noting that the noticeable discrepancy in scores between the metrics suggests issues that need to be addressed at annotation and extraction level. Our pipeline still performs poorly on adjectival relationships, identifying acronyms (e.g., *human arm* versus *ARM*) and specific wordings (e.g., *6 cycle* versus *cycle 6*) which explains the drops in scores between metrics.

Table 1: Triple extraction summary.

Object	**Count**
Sentences	7407
Tokens in sentences	172836
Tokens covered in triples	131625
Verbal or prepositional relations	16051
Unique triple objects	13821

Table 2: Alignment Summary.

Object	**Count**
Unique NER objects linked to UMLS	4002
I2b2 NER objects linked to UMLS	3417
NER to CUI/TUI indirect links	7151
NER to TUI direct links	1533

Table 3: Comparative results of our methodology.

Method	**CVRG**	**MAPG**	**ALGT**
Baseline	68.00	40.00	10.00
Our Pipeline	**76.16**	**53.67**	**21.40**

6 CONCLUSION

We introduce a system for domain information abstraction from text and ontology alignment for a more effective KG creation. Our method has the advantage of providing good text-to-triple coverage while maintaining strict semantic consistency for overlapping tokens, which allows better mapping and alignment to higher-order domain ontologies. Our experiments show the need to expand the annotation and extraction processes of our system in order to handle edge cases in unstructured text and capture triples more faithfully. In future work, we will target enhancing the triple extraction process from text by making the annotator more flexible with handling edge cases like acronyms or sentence rewordings. We will integrate features like coreference resolution to capture more fine-grained triples and improve KG construction. we will also aim to evaluate our system against other architectures like LLMs and widen the scope of our experimentation to include other types of biomedical documents (e.g., Protocols) as well as extend it to other domains like finance.

REFERENCES

Abi Akl, H. (2023). The path to autonomous learners. In *Science and Information Conference*, pages 808–830. Springer.

Chaurasiya, D., Surisetty, A., Kumar, N., Singh, A., Dey, V., Malhotra, A., Dhama, G., and Arora, A. (2022). Entity alignment for knowledge graphs: progress, challenges, and empirical studies. *arXiv preprint arXiv:2205.08777*.

Chouham, E. M., Espejel, J. L., Alassan, M. S. Y., Dahhane, W., and Ettifouri, E. H. (2023). Entity identifier: A natural text parsing-based framework for entity relation extraction. *arXiv preprint arXiv:2307.04892*.

Cotovio, P. G., Jimenez-Ruiz, E., and Pesquita, C. (2023). What can knowledge graph alignment gain with neuro-symbolic learning approaches? *arXiv preprint arXiv:2310.07417*.

Dao, N.-M., Hoang, T. V., and Zhang, Z. (2023). A benchmarking study of matching algorithms for knowledge graph entity alignment. *arXiv preprint arXiv:2308.03961*.

Dong, K., Sun, A., Kim, J.-J., and Li, X. (2023). Open information extraction via chunks. *arXiv preprint arXiv:2305.03299*.

Dunn, A., Dagdelen, J., Walker, N., Lee, S., Rosen, A. S., Ceder, G., Persson, K., and Jain, A. (2022). Structured information extraction from complex scientific text with fine-tuned large language models. *arXiv preprint arXiv:2212.05238*.

Fanourakis, N., Efthymiou, V., Kotzinos, D., and Christophides, V. (2023). Knowledge graph embedding methods for entity alignment: experimental review. *Data Mining and Knowledge Discovery*, 37(5):2070–2137.

Friedman, S., Magnusson, I., Sarathy, V., and Schmer-Galunder, S. (2022). From unstructured text to causal knowledge graphs: A transformer-based approach. *arXiv preprint arXiv:2202.11768*.

Hong, Z., Chard, K., and Foster, I. (2024). Combining language and graph models for semi-structured information extraction on the web. *arXiv preprint arXiv:2402.14129*.

Kamp, S., Fayazi, M., Benameur-El, Z., Yu, S., and Dreslinski, R. (2023). Open information extraction: A review of baseline techniques, approaches, and applications. *arXiv preprint arXiv:2310.11644*.

Liu, P., Gao, W., Dong, W., Huang, S., and Zhang, Y. (2022). Open information extraction from 2007 to 2022–a survey. *arXiv preprint arXiv:2208.08690*.

Melnyk, I., Dognin, P., and Das, P. (2022). Knowledge graph generation from text. *arXiv preprint arXiv:2211.10511*.

Mihindukulasooriya, N., Tiwari, S., Enguix, C. F., and Lata, K. (2023). Text2kgbench: A benchmark for ontology-driven knowledge graph generation from text. In *International Semantic Web Conference*, pages 247–265. Springer.

Motger, Q. and Franch, X. (2024). Nlp-based relation extraction methods in re. *arXiv preprint arXiv:2401.12075*.

Pan, J. Z., Razniewski, S., Kalo, J.-C., Singhania, S., Chen, J., Dietze, S., Jabeen, H., Omeliyanenko, J., Zhang, W., Lissandrini, M., et al. (2023a). Large language models and knowledge graphs: Opportunities and challenges, 2023. *arXiv preprint arXiv:2308.06374*.

Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., and Wu, X. (2023b). Unifying large language models and knowledge graphs: A roadmap, 2023. *arXiv preprint arXiv:2306.08302*.

Peng, C., Xia, F., Naseriparsa, M., and Osborne, F. (2023). Knowledge graphs: Opportunities and challenges. *Artificial Intelligence Review*.

Qian, K., Belyi, A., Wu, F., Khorshidi, S., Nikfarjam, A., Khot, R., Sang, Y., Luna, K., Chu, X., Choi, E., et al. (2023). Open domain knowledge extraction for knowledge graphs. *arXiv preprint arXiv:2312.09424*.

Trisedya, B. D., Salim, F. D., Chan, J., Spina, D., Scholer, F., and Sanderson, M. (2023). i-align: an interpretable knowledge graph alignment model. *Data Mining and Knowledge Discovery*, 37(6):2494–2516.

Van Tong, V., Huynh, T. T., Nguyen, T. T., Yin, H., Nguyen, Q. V. H., and Huynh, Q. T. (2021). Incomplete knowledge graph alignment. *arXiv preprint arXiv:2112.09266*.

Vogt, L., Kuhn, T., and Hoehndorf, R. (2022). Semantic units: Organizing knowledge graphs into semantically meaningful units of representation [internet].

Wadhwa, S., Amir, S., and Wallace, B. C. (2023). Revisiting relation extraction in the era of large language models. *arXiv preprint arXiv:2305.05003*.

Xie, F., Zeng, X., Zhou, B., and Tan, Y. (2023). Improving knowledge graph entity alignment with graph augmentation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 3–14. Springer.

Yang, L., Chen, H., Wang, X., Yang, J., Wang, F.-Y., and Liu, H. (2024). Two heads are better than one: Integrating knowledge from knowledge graphs and large language models for entity alignment. *arXiv preprint arXiv:2401.16960*.

Yu, B., Zhang, Z., Li, J., Yu, H., Liu, T., Sun, J., Li, Y., and Wang, B. (2022). Towards generalized open information extraction. *arXiv preprint arXiv:2211.15987*.

Zeng, K., Li, C., Hou, L., Li, J., and Feng, L. (2021). A comprehensive survey of entity alignment for knowledge graphs. *AI Open*, 2:1–13.

Zhang, R., Su, Y., Trisedya, B. D., Zhao, X., Yang, M., Cheng, H., and Qi, J. (2023). Autoalign: fully automatic and effective knowledge graph alignment enabled by large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, R., Trisedy, B. D., Li, M., Jiang, Y., and Qi, J. (2021). A benchmark and comprehensive survey on knowledge graph entity alignment via representation learning. *arXiv preprint arXiv:2103.15059*.

Decoding AI's Evolution Using Big Data: A Methodological Approach

Sophie Gvasalia[1], Mauro Pelucchi[1], Simone Perego[1] and Rita Porcelli[2]
[1]*Global Data Science, Lightcast, Italy*
[2]*INAPP – Istituto per l'Analisi delle Politiche Pubbliche, Italy*
{*sophie.gvasalia, mauro.pelucchi, simone.perego*}*@lightcast.io, r.porcelli@inapp.gov.it*

Keywords: Big Data Methodology, Artificial Intelligence Impact, Job Market Analysis.

Abstract: This study presents a novel approach to measuring the impact of Artificial Intelligence on occupations through an analysis of the Atlante del Lavoro dataset and web job postings. By focusing on data preparation and model selection, we provide real-time insights into how AI is reshaping job roles and required skills. Our methodological framework enables a detailed examination of specific labour market segments, emphasizing the dynamic nature of occupational demands. Through a rigorous mixed-method approach, the study highlights the AI impact on sectors such as ICT, telecommunications, and mechatronic, revealing distinct skill clusters and their significance. This innovative analysis not only delineates the convergence of digital, soft, and hard skills but also offers a multidimensional view of future workforce competencies. The findings serve as a valuable resource for educators, policymakers, and industry stakeholders, guiding workforce development in line with emerging AI-driven demands.

1 INTRODUCTION

The integration of artificial intelligence into work processes is set to significantly influence various types of workers, leading to changes in wage structures and skill requirements. Public policies will, therefore, play a critical role in promoting training and ensuring that workers are adequately prepared for the transformations in the labour market. This study aims to provide an innovative analysis of the labour market, focusing on the measure of the impact of AI-related skills across different economic sectors. By utilizing data from the Atlante del Lavoro[1] and online job postings (provided by Lightcast[2]), this study explores the competencies demanded in job postings and characterizes professional profiles, offering a detailed perspective on the emerging labour dynamics in sectors significantly impacted by AI.

To explore this relationship, we apply big data principles, focusing on five key dimensions: volume, by leveraging a large dataset of online job postings for comprehensive labour market analysis; velocity, with near real-time processing to capture dynamic changes in AI skill demands; variety, by combining structured data from Atlante del Lavoro with unstructured job postings for a multifaceted view; veracity, using NLP techniques to ensure data accuracy and reliability; and value, providing actionable insights for policymakers, training institutions, and industry stakeholders on AI-related skills and their economic impact. Through rigorous quantitative analysis and qualitative interpretation, we aim to characterize emerging professional profiles and elucidate the nuanced interplay between AI adoption and skill demand. This approach allows for a granular examination of labour market trends, with particular emphasis on sectors experiencing significant AI-driven transformations.

AI's labour market impact includes job displacement, creation, and transformation. Studies indicate AI automates routine and non-routine tasks, altering work and skill demands. (Lane et al., 2023) highlight how AI transforms roles by automating repetitive tasks, increasing the need for cognitive and socioemotional skills. Generative Pre-trained Transformers (GPT) further revolutionize labour by automating tasks requiring natural language understanding. GPT is widely used in customer service, content creation, and data analysis, enhancing productivity (Eloundou et al., 2023). As GPT reshapes job roles, new training programs are essential for effective AI collaboration. (Squicciarini and Nachtigall, 2021) investigations confirm that the majority of workers developing and maintaining AI possess these specialized skills, although not all workers involved in AI

[1] https://atlantelavoro.inapp.org/

[2] https://lightcast.io

Gvasalia, S., Pelucchi, M., Perego, S. and Porcelli, R.
Decoding AI's Evolution Using Big Data: A Methodological Approach.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 424-432
ISBN: 978-989-758-716-0; ISSN: 2184-3228

have these skills to the same extent. The demand for these specialized AI skills has grown significantly in recent years, particularly in the United States, with similar trends observed in Canada, Singapore, and the United Kingdom. Job advertisements increasingly demand AI skills alongside transversal competencies such as social skills and management abilities, indicating their complementary nature.

Our approach innovatively analyzes the effects of AI on job competencies by integrating large-scale job posting data with the Atlante del Lavoro framework. This method provides a detailed view of evolving skill demands and AI-related skills across sectors. Advanced text mining techniques capture emerging trends in real-time, offering more immediate insights than traditional surveys. The dynamic mapping of AI adoption and skill demand reveals subtle shifts in job roles, often missed by conventional methods.

The study begins with a review of the state of the art in research on AI's impact on the labour market. This section will introduce the current understanding and highlight gaps that this research aims to fill. Following this, section 3 presents the data used in our analysis. The data section also covers the preprocessing steps undertaken to ensure the accuracy and relevance of the data for our study. In the section 4, we detail our innovative approach that integrates both qualitative and quantitative data to evaluate the impact of AI on work activities. Advanced Machine Learning and NLP techniques are employed to estimate AI's impact efficiently. This section provides a comprehensive description of the data preparation and model selection processes, highlighting the nuances of our methodological framework. The results section presents our findings on the impact of AI in specific sectors such as ICT, telecommunications, and mechatronic. The conclusion discusses the implications of our findings, the limitations of the study, and suggests directions for future research.

2 RELATED WORK

Recent studies indicate a significant acceleration in the adoption of Artificial Intelligence (AI) across various sectors of the economy. The PwC AI Barometer[3] reports that 52% of companies have expedited their AI adoption plans, with 86% anticipating AI to become a mainstream technology within their organisations by 2024. This trend is corroborated by the OECD AI surveys (Lane et al., 2023), which found that 24% of businesses across OECD countries are currently utilising AI technologies. Notably, there exists a substantial disparity in adoption rates based on firm size, with large firms being ten times more likely to adopt AI than their smaller counterparts.

(Brynjolfsson and McAfee, 2014) highlight AI's dual effect on the workforce: automation of routine tasks and creation of new roles requiring advanced skills. This underscores the complex interplay between technological innovation and labour market shifts, with both job displacement and new opportunities emerging. (Autor, 2015) delved into the polarization of the labour market caused by technological advancements. The research indicates that AI and automation technologies tend to replace middle-skill jobs that involve routine tasks, while simultaneously increasing demand for both high-skill jobs that require creative and cognitive abilities and low-skill jobs that involve non-routine manual tasks. This polarization highlights the need for targeted educational and training programs to equip workers with the skills necessary to thrive in an AI-driven economy. (Frey and Osborne, 2017) utilize a Gaussian process classifier to estimate automation probabilities for 702 occupations, based on O*NET data. The study highlights sectoral automation risks, though it focuses on current technology and technical feasibility, neglecting future advancements and workforce adaptability.

Recent empirical investigations have delved into the specific competencies. (Alekseeva et al., 2021) conducted a comprehensive analysis of job postings, revealing a marked upsurge in demand for AI-centric skills, encompassing proficiency in programming languages such as python, expertise in big data management, and capabilities in model development. Complementing this work, (Acemoglu et al., 2022) explored the intricate relationship between AI technology adoption and evolving workforce skill demands. Their findings underscore the symbiotic nature of technical proficiencies and soft skills in the contemporary labour landscape, highlighting the complex interplay between technological advancement and human capital development in shaping employment dynamics.

The methodological approaches for analyzing the impact of AI on job postings have also evolved. (Manca, 2023) utilized advanced NLP techniques to parse and analyze large datasets of job advertisements, providing real-time insights into emerging skill demands. This approach enables a more dynamic understanding of labour market trends, as opposed to traditional static analyses.

(Eloundou et al., 2023) integrate expert judgments with datasets from O*NET, ILO, and the World Bank, assessing generative AI's impact on 923 occupations

[3] https://www.pwc.com/AIJobsBarometer

across 199 countries. The broad scope and AI exposure scoring system are strengths, though expert reliance and rapid AI evolution pose limitations. Despite extrapolation challenges, the study offers valuable insights for global labour markets. (Weichselbraun et al., 2024) employs a deep learning-based approach to anticipate future job market demands by assessing the automatability and offshorability of skills. The authors use a combination of Support Vector Machines (SVMs), Transformers, and Large Language Models (LLMs) to classify skills and estimate their future relevance. Their findings highlight the increasing demand for skills related to automation and offshoring, driven by trends like the Gig economy and technological advancements.

3 DATA

The Atlante del Lavoro is the Italian classificatory and informative device for work and qualifications, created based on the descriptive sequences of the Classification of 24 Economic Professional Sectors (SEPs). The Atlante del Lavoro was developed as part of the construction of the National Repository of Education and Training Titles and Professional Qualifications, as stipulated by Legislative Decree No. 13 of January 16, 2013[4]. It aims to systematize and correlate the competencies of qualifications from the public lifelong learning offerings with work activities. The sectors were generated by intersecting two independent ISTAT classifications, both in terms of the object represented and the constructive criteria used: the classification of economic activities (ATECO 2007[5]) and the classification of professions (CP 2011, updated in 2023 with CP 2021[6]).

All the codes constituting the aforementioned statistical classifications, at their maximum extension, have been aggregated in the Atlante del Lavoro Economic Professional Sectors (SEPs) to meet the empirical need to identify a ”perimeter” where sets of work processes and activities with relative internal homogeneity (intra-sectoral) and sufficient external distinction (inter-sectoral) can be placed and ordered in their information field. The Economic Professional Sectors (SEPs) are articulated into work processes, process sequences, activity areas, and individual activities described following the typical logic of the value chain model (Mazzarella et al., 2017). These descriptors are constantly updated to meet the need to track the evolution of constantly changing work activities.

The INAPP[7] study utilized online job advertisements to calculate skill rates and assess the relevance of these skills for the descriptors within the Atlante del Lavoro, which detail each segment of work. Three indicators were defined to measure the evolution of work dynamics in processes, sequences, and area of activities (ADAs), quantifying the skill rate for each system component (Mezzanzanica et al., 2018). These indicators measure the incidence of digital, soft, and hard non-digital skills on the Atlante del Lavoro's descriptive elements:

$$Skill\_Rate_t = \frac{f_t}{f_s + f_d + f_h}$$

Where t denotes the skill type (digital, soft, or hard non-digital), with f_t representing the frequency of type t, f_s the frequency of soft skills, f_d the frequency of digital skills, and f_h the frequency of hard non-digital skills in the dataset. Three indicators were identified, relating to classes of macro-competencies based on ESCO skills[8], extracted from the job postings database. These indicators measure and monitor over time the degree of digitalization, the demand for soft skills, and the demand for technical/hard skills within Economic Professional Sectors (SEPs), Processes, Sequences, and Area of Activities (ADAs).

The working method can be summarized in the following steps:

(i) Job advertisements are used as measurement tools to elaborate all indices on macro-competencies. Each job advertisement is classified according to the CP 2011 standard, at the 5-digit level.

(ii) Ads are linked to Area of Activities (ADAs) through the associated profession. It should be noted that only advertisements classified according to the occupations belonging to the area of activity contribute to the calculation of indicators. During the association between area of activity and job advertisements, the correspondence between a single area of activity and the occupations may not be one-to-one: each profession can be associated with multiple activities. In this case, if the same profession is associated with multiple areas belonging to the same sequence, the job advertisements for this sequence are considered only once.

(iii) Job advertisements associated with each area can be further filtered based on the industry codes

[4] https://www.gazzettaufficiale.it/eli/id/2013/02/15/13G00043/sg

[5] https://www.istat.it/en/classification/ateco-classification-of-economic-activity-2007/

[6] https://www.istat.it/en/classification/classification-of-occupations/

[7] https://www.inapp.gov.it/en/homepage

[8] https://esco.ec.europa.eu/en/classification

associated with the process Sequence to which the area belongs. The industry sectors filter, in some cases, reduces the expressive capacity of the database, but when present, it refines the match.

(iv) Job advertisements associated with each area report the required skills, categorized according to the ESCO classification and grouped by macro-competency classes.

The skills rate, broken down into macro-competence areas, monitors the results within the area of activity, sequence, or process through successive aggregations and it tracks the evolution of jobs over time.

3.1 Job Postings Dataset

The Lightcast database currently consists of over 21 million online job postings for Italy. After thorough data cleaning, it contains more than 8 million validated job postings. These postings, often in semi-structured or unstructured text, require rigorous scientific, methodological, and technical work to extract useful information. Covering the entire national territory, they provide a rich data source for analyzing various dimensions (occupations, industry sectors, regions, and skills) (Vrolijk et al., 2022).

The processing phases are: (i) Data Collection: Extracting job postings via API, bulk extraction, and scraping. (ii) Data Treatment: Structuring data to meet shared standards. (iii) Text Processing: Preparing unstructured texts for classification. (iv) Classification: Extracting professions and skills from job postings.

Occupations are extracted from texts using a combination of machine learning algorithms that train the classifier based on previously classified and expert-validated occurrences. Skills are extracted using feature extraction techniques and mapped to the ESCO standard. Each skill is then associated with a macro-competence class: digital, soft, or hard-no-digital, defined by the working group using ESCO and O*NET classification pillars. Hard skills are specific job abilities, while soft skills are interpersonal and environmental interaction abilities. Digital skills within hard skills include ICT tool usage to complex system design. Soft skills include thinking, social interaction, knowledge application, and attitudes and values. For each occupation, the required competencies are analyzed, and the frequency of soft, hard non-digital, and digital skills is calculated.

(Lovaglio, 2022) introduces a methodology for analyzing labour market trends using web-scraped job vacancies, revealing the growing importance of digital skills across sectors. Despite potential biases in online recruitment, the approach provides real-time insights. Similarly, (Vermeulen and Amaros, 2024) and (Enrique and Matteo, 2024) assess the validity of Lightcast job posting data compared to national statistics across Europe (2019–2022). Their benchmarking highlights discrepancies but emphasizes the complementary value of online postings for tracking labour demand trends.

4 METHODOLOGY

The methodology involves selecting a representative sample from the Atlante del Lavoro using stratified sampling by Economic Professional Sector (SEP). Sector experts then evaluated the AI impact on sampled work activities. Next, NLP techniques were applied to efficiently analyze the data corpus and estimate AI's impact across all activities. Finally, work activities were classified based on AI impact estimates, enabling a clear assessment of sectoral transformations. This methodology partially draws from (Frey and Osborne, 2017) study. A 5% random sample of areas of activity was extracted using stratified sampling to ensure fair sector representation. The sample comprises 48 Area of Activities (ADAs), with each of the 24 Economic Professional Sectors (SEPs) represented by 2 ADAs (see Table 1).

To ensure the reliability and consistency of the labeling effort, we assessed inter-rater agreement among the five industry experts who evaluated the AI impact on various work activities. Each expert independently assigned scores ranging from 1 to 5, reflecting the perceived impact of AI on specific activities. To gauge the consistency of these assessments, we calculated inter-rater agreement using the Fleiss' Kappa method. The calculated Fleiss' Kappa value of 0.2128 suggests fair agreement among the raters. This indicates some level of consistency in their evaluations, but the variability implies differences in their assessment criteria (see Table 2).

To efficiently analyze the impact of AI on all work activities, a methodology based on advanced machine ,earning techniques was adopted. Initially, Sentence BERT (Bidirectional Encoder Representations from Transformers, (Reimers and Gurevych, 2019)), a natural language representation model, was used to create a data corpus containing descriptions of work activities and their related embeddings. Subsequently, the dataset was divided into training and test sets to train and evaluate the ML models. Various models were explored, including XGBoost (Chen and Guestrin, 2016), linear and polynomial regression, neural networks, and support vector machines

Table 1: Sample of the area of activities evaluated by the experts.

Sector	**Area of Activity description**
Chemistry	Operation and control of plants/machines in the production of sterile and non-sterile drugs
Chemistry	Processing of plastics and rubber
Construction	Execution of foundations and tunnels
Construction	Painting works
Extraction of gas, oil, coal, minerals, and stone processing	Environmental recovery of disused extraction areas
Extraction of gas, oil, coal, minerals, and stone processing	Preparation and squaring of blocks
Wood and furniture	Selection and storage of lots
Wood and furniture	Packaging of curtains and drapes
Mechanics, production, and maintenance of machinery, plant engineering	Maintenance and repair of mechanical and structural components of aircraft
Mechanics, production, and maintenance of machinery, plant engineering	Maintenance and repair of household appliances and electrical devices

Table 2: Sample of AI Impact Evaluation on work activities from the experts.

Sector	**Description**	**Expert evaluation**
Social and Health Services	Implementation of clown therapy interventions	1
Tourism Services	Operational management of bathing services	2
Tourism Services	Operational management of ski slopes and implementation of rescue interventions	3
Printing and Publishing	Handcrafted production of prints using lithographic processes	3
Printing and Publishing	Digital archiving of the publishing house's documentary heritage	5

(SVM), in order to estimate the impact of AI on work activities. Among the various models tested (see Figure 1), those that showed the best performance were selected for subsequent analysis. The performances, computed on the test set and reported in table 3 indicate that the Gradient Boosting model performs best, with an R² of 0.417, meaning it explains 41.7% of the variance in the data. The Gradient Boosting model utilizes XGBoost. Key parameters include the number of trees (100), which dictates the ensemble's size, and the learning rate (0.1), controlling how much each tree contributes to the model. The regularization (Lambda: 10) adds penalties to prevent overfitting, and the depth of trees (10) controls tree complexity, balancing model expressiveness and overfitting risk. Subsampling parameters are set to 1.0, meaning the entire dataset and all features are used at each step.

Table 3: Comparison of model performance metrics evaluated on the test set.

Model	**MSE**	**RMSE**	**MAE**	**R²**
Gradient Boosting	0.400	0.633	0.490	0.417
Linear Regression	0.494	0.703	0.641	0.282
Neural Network	0.703	0.838	0.635	-0.022
SVM	0.503	0.709	0.587	0.268

Based on the estimates obtained through models, work activities were classified according to their relative AI impact. Activities were divided into categories, including areas with high impact (score $>$ 3.5), medium impact (score $>$ 2 and $\leq$ 3.5), and low impact (score $>$ 1 and $<$ 2). This categorization provides a clear overview of AI's influence on different work activities and sectors.

5 RESULTS

The results of the analysis provided a detailed overview of the impact of AI on work activities across various economic sectors.

5.0.1 AI Impact

Initially, we examined the number of activities classified based on AI impact, distinguishing between high impact (category A), medium impact (category B), and low impact (category C). The results indicate that the most significant number of activities fall into category B (564 activities), followed by category C (254 activities), while category A includes 84 activities.

Analyzing the industry sectors and their respective areas of activity with significant AI impact, data shows considerable variation across sectors (Table 4). For instance, in the ”Digital Services” sector, most ADA (63.64%) are classified as high impact; in ”Printing and Publishing” and ”Construction” 45.45% and 4.17%, respectively. The strong AI presence in ICT related services drives demand for specific skills such as programming, data analysis, and AI itself. Sectors with medium AI impact, like printing and manufacturing, require professionals to adapt

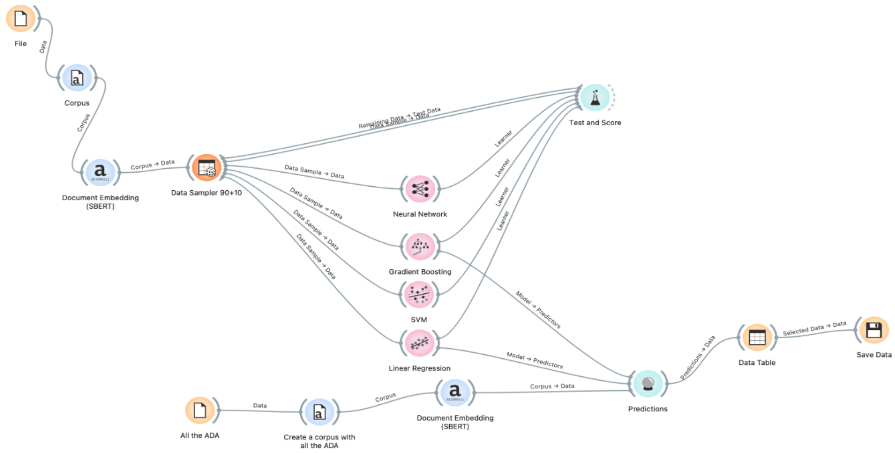


Figure 1: The text mining process to evaluate the AI impact on all the area of activities. The process begins with document embedding using SBERT[9], followed by a data split for training and testing. Four models (Neural Network, Gradient Boosting, Support Vector Machine, and Linear Regression) are trained and evaluated using test data. The best-performing model generates predictions, which are output for further analysis.

their skills to optimize work processes using AI technologies. Even in sectors with limited AI impact, such as education, professional development must address digitalization demands, preparing the workforce for future innovations.

The analysis highlights the need for continuous adaptation of the education and skill development system in response to technological evolution. Training should focus on transversal digital skills, such as critical thinking, problem-solving, and communication, in addition to AI-specific skills and their applications in various sectors (Pedone A. 2024, Conforti D. 2024).

To complete the study, we explored the link between digital skills (using the digital skills rate at the Area of Activity level) and the AI impact on ADA. A polynomial model was used to calculate the coefficient of determination (R²), which measures how much variation in AI impact is explained by digital skills. The R² of 0.2095 indicates that 20% of the variability in AI impact is explained by digital skills. The adjusted R² was 0.2078, suggesting the model is appropriate and does not overfit. While a correlation exists, the relatively low R² suggests that other factors contribute to explaining AI impact variability, indicating the need for further research (Figure 2).

5.0.2 Digital Services

The analysis of the ICT sector highlights several key findings. The sector shows a high susceptibility to AI-driven changes, with a significant portion of activities impacted by AI. Skill demands are shifting towards AI-related competencies, with a notable emphasis on machine learning, NLP, and data analytics. Digital skills dominate the sector, and soft skills are increasingly valued, especially for roles managing AI. These insights stress the need for an adaptable ICT workforce, capable of continuous upskilling to keep pace with AI advancements (Table 5).

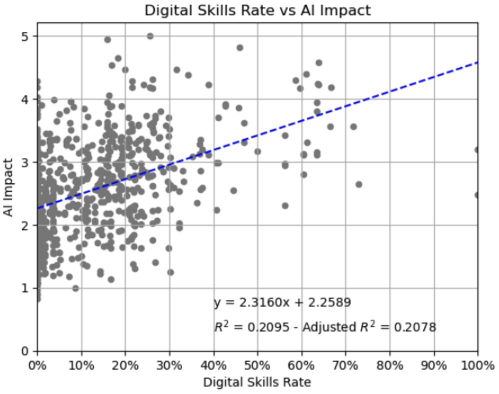


Figure 2: Digital Skills Rate and AI Impact. The digital skills rate is represented on the horizontal axis of this scatter plot, while the vertical axis presents the AI impact.

Job postings in 2023 already highlight AI-related skills such as Apache Spark, Machine Learning, NLP, Computer Vision, PyTorch, Deep Learning, Keras, Generative AI, Cognitive Computing, and Large Language Modeling. To attract talent in the ICT sector, it is important to introduce policies focused on upskilling and reskilling the workforce. These initiatives enhance existing employee skills, making the region more competitive and appealing to potential talent. Continuous learning ensures that the workforce stays adaptable, filling skill gaps and positioning the region as a hub for innovation and professional growth (Gatti et al., 2022).

Table 4: AI Impact on the economic sectors.

Industry	A (High Impact)	B (Medium Impact)	C (Low Impact)
Agriculture, forestry and fishing	2.00%	58.00%	40.00%
Food production	4.76%	69.05%	26.19%
Wood and furniture	0.00%	21.74%	78.26%
Paper and papermaking	0.00%	75.00%	25.00%
Textiles, clothing, footwear and fashion system	0.00%	27.50%	72.50%
Chemistry	0.00%	76.00%	24.00%
Extraction of gas, oil, coal, minerals and stone processing	0.00%	43.33%	56.67%
Glass, ceramics and building materials	0.00%	28.57%	71.43%
Construction	4.17%	41.67%	54.17%
Mechanics, machine production and maintenance, plant engineering	11.32%	63.21%	25.47%
Transport and logistics	7.35%	89.71%	2.94%
Commercial distribution services	0.00%	90.00%	10.00%
Financial and insurance services	2.08%	81.25%	16.67%
Digital Services	63.64%	36.36%	0.00%
Telecommunication and postal services	38.46%	61.54%	0.00%
Public utilities services	18.18%	68.18%	13.64%
Printing and publishing	45.45%	45.45%	9.09%
Education, training and employment services	9.38%	90.63%	0.00%
Social and health services	4.17%	79.17%	16.67%
Personal services	0.00%	29.41%	70.59%
Recreational and sports services	0.00%	62.50%	37.50%
Cultural and entertainment services	11.11%	64.81%	24.07%
Tourism services	9.68%	83.87%	6.45%
Common area	20.55%	73.97%	5.48%

Table 5: ADA SEP – High AI Impact Digital Services and corresponding digital skills rate.

Area of activity	AI Impact	Digital Skills Rate
Engineering ICT Systems	4.57	64%
Improving ICT Processes	4.46	20%
Innovation in ICT	4.4	61%
Data Science and Analytics	4.38	34%
Sustainability Management in ICT	4.29	59%
Monitoring Technological Trends	4.23	64%
Information and Knowledge Management	4.23	64%
Problem Management in ICT	4.18	67%
Defining IT Strategy and Aligning with Business	4.16	60%
User Experience Design	3.95	63%
Application Development	3.8	63%
Developing Cybersecurity Strategy	3.62	47%
Providing ICT Services	3.57	66%
Supporting System Changes and Evolutions	3.55	72%

5.0.3 Telecommunications

In the telecommunications sector, AI significantly impacts work activities, as shown by the digital skill rate of area of activity analyzed for this SEP (Table 6). The highest AI impact is seen in Network Architecture Design and Planning (5.00), requiring a 26% digital skill rate. Installation, Configuration, and Testing of TLC Systems have a high AI impact (4.64) with an 18% digital skill rate. Network Management and Supervision also show significant AI impact (4.18) with a 26% digital skill rate. Conversely, Online Shipping Service Programming has a lower AI impact (3.88) with a 19% digital skill rate, while TLC System Maintenance Assistance has a moderate AI impact (3.87) with an 8% digital skill rate.

Job postings in the telecommunications sector highlight the demand for AI-related skills, such as Machine Learning, K-Means Clustering, Deep Learning, and Natural Language Processing. Technologies like Apache Spark, TensorFlow, PyTorch, and Keras are widely adopted, emphasizing the need for skills in data analysis, ICT system management, and project management methodologies.

Table 6: ADA SEP – Telecommunications and Postal Services with High AI Impact and Corresponding Digital Skills Rate.

Area of activity	AI Impact	Digital Skills Rate
Network Architecture Design and Planning	5.00	26%
Installation, Configuration, and Testing of TLC Systems	4.64	18%
Management, Supervision, and Control of TLC System Components and Networks	4.18	26%
Programming and Control of Online Shipping Services	3.88	19%
Assistance/Maintenance of TLC Systems	3.87	8%

5.0.4 Mechatronic

In the mechatronic, AI integration is revolutionizing several operational activities (Table 7). Key areas include programming and automating electronic systems, utilizing AI platforms to optimize production processes, and improving assembly line efficiency. AI solutions in electrical/electronic installation on boats integrate smart sensors and control algorithms, enhancing system safety and reliability. Aerospace sector AI optimizes production of components and vehicles through advanced modeling and virtual simulations. Building automation systems use machine learning algorithms to optimize energy consumption and occupant comfort.

Job postings in this sector frequently mention roles such as electromechanics, industrial engineers, telecommunication technicians, and aerospace engineers. Skills in Machine Learning, Apache Spark, Computer Vision, and Natural Language Processing are highly sought after, indicating a growing need for AI competencies to enhance automation, safety, and efficiency in industrial and electronic systems.

6 CONCLUSIONS

The analysis investigated AI's impact across various labour segments in the Atlante del Lavoro using online job vacancies. Results revealed differing AI impacts, categorized as high, medium, and low, with a correlation between digital skills and AI impact on work activities. Analysis of selected sectors focused on high-impact area of activities, identifying key professions and AI-related skills. A clear distinction was found between AI application roles, which require digital literacy and domain-specific expertise, and AI development roles, which demand specialized skills like Python, SQL, and machine learning. These findings underscore the varied skillsets needed across AI's influence on the labour market.

In Digital Services, AI automates repetitive tasks, requiring new skills for designing intelligent applications like Robotic Process Automation (RPA) and data analysis tools. In telecommunications, AI automates network management, enhances customer interactions through NLP, and improves predictive analytics for network maintenance. In the mechatronics sector, AI-driven robots boost efficiency, and machine learning predicts equipment failures, while simulators optimize production processes and predictive maintenance reduces downtime. In addressing the feasibility of satisfying the required shift in skills, it is important to acknowledge the gap between the demand for advanced AI skills, such as deep learning, and the realistic capacity for most workers to acquire them. While reskilling efforts can help non-technical workers adopt AI-related competencies, expecting a significant portion of the workforce to master complex areas like deep learning is unrealistic. Recent studies suggest that non-technical workers are better suited to focus on skills such as AI collaboration, data literacy, and problem-solving, which are more attainable and still highly relevant in an AI-driven environment (see (Whelan and Redmond, 2024)).

The study has limitations, such as potential biases from relying on online job vacancies and overlooking future AI advancements. The analysis is not exhaustive across sectors, and digital skills measurement may miss emerging trends. Expert evaluations introduce subjectivity, and the study's focus on a specific timeframe and region limits broader applicability. Economic shifts and sectoral AI adoption rates are not fully considered. However, the study emphasizes the need for investment in training programs, identifying two key skill areas: technical competencies for AI system development and general skills for AI adoption and interaction.

ACKNOWLEDGEMENTS

Simone Perego handled data preprocessing and preparation for online job posting datasets. Sophie Gvasalia performed data extraction and linked the Atlante del Lavoro dataset with job postings. Mauro Pelucchi implemented data mining and model selection. Rita Porcelli curated the manuscript and synthesized policy-relevant insights.

Table 7: ADA SEP – Mechatronic with High AI Impact and Corresponding Digital Skills Rate.

Area of activity	**AI Impact**	**Digital Skills Rate**
Programming Electronic Systems for Automation Control	4.53	17%
Installation of Electrical/Electronic Systems on Boats	3.82	7%
Manual and Automated Machine Forming	3.75	0%
Installation and Repair of TV Reception and Signal Systems	3.75	0%
Designing Renewable Energy Source (RES) Systems	3.73	27%
System Integration for Optimizing Aerospace Components and Vehicles Production	3.73	25%
Customer Installation, Commissioning, and Testing	3.71	12%
Design of Thermohydraulic Systems (e.g., civil, industrial, HVAC)	3.69	27%
Installation/Maintenance of Industrial Electrical Systems	3.61	14%
Management and Improvement of Aerospace Production Processes and Logistics	3.61	21%
Building Automation Systems Setup and Management	3.59	26%
Installation/Maintenance of Civil and Commercial Electrical Systems	3.57	11%

REFERENCES

Acemoglu, D., Autor, D., Hazell, J., and Restrepo, P. (2022). Artificial intelligence and jobs: Evidence from online vacancies. *Journal of Labor Economics*, 40(S1):S293–S340.

Alekseeva, L., Azar, J., Giné, M., Samila, S., and Taska, B. (2021). The demand for ai skills in the labor market. *Labour economics*, 71:102002.

Autor, D. H. (2015). Why are there still so many jobs? the history and future of workplace automation. *Journal of economic perspectives*, 29(3):3–30.

Brynjolfsson, E. and McAfee, A. (2014). *The second machine age: Work, progress, and prosperity in a time of brilliant technologies*. WW Norton & Company.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Eloundou, T., Manning, S., Mishkin, P., and Rock, D. (2023). Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130.*

Enrique, F. M. and Matteo, S. (2024). Skewed signals? confronting biases in online job ads data. Technical report, Joint Research Centre.

Frey, C. B. and Osborne, M. A. (2017). The future of employment: How susceptible are jobs to computerisation? *Technological forecasting and social change*, 114:254–280.

Gatti, A. C., Colombo, E., Magrini, E., Perego, S., and Pelucchi, M. (2022). Understanding talent attraction using online job ads: the impact of artificial intelligence and green jobs. In *The Relevance of Artificial Intelligence in the Digital and Green Transformation of Regional and Local Labour Markets Across Europe*, pages 129–164. Nomos Verlagsgesellschaft mbH & Co. KG.

Lane, M., Williams, M., and Broecke, S. (2023). The impact of ai on the workplace: Main findings from the oecd ai surveys of employers and workers.

Lovaglio, P. G. (2022). Do job vacancies variations anticipate employment variations by sector? some preliminary evidence from italy. *Labour*, 36(1):71–93.

Manca, F. (2023). Six questions about the demand for artificial intelligence skills in labour markets.

Mazzarella, R., Mallardi, F., and Porcelli, R. (2017). Atlante lavoro: un modello a supporto delle politiche dell'occupazione e dell'apprendimento permanente. *Sinappsi*, 7:2–3.

Mezzanzanica, M., Mercorio, F., and Colombo, E. (2018). Digitalisation and automation: Insights from the online labour market. In *Developing Skills in a Changing World of Work*, pages 259–282. Rainer Hampp Verlag.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084.*

Squicciarini, M. and Nachtigall, H. (2021). Demand for ai skills in jobs: Evidence from online job postings.

Vermeulen, W. and Amaros, F. G. (2024). How well do online job postings match national sources in european countries?: Benchmarking lightcast data against statistical and labour agency sources across regions, sectors and occupation.

Vrolijk, J., Mol, S. T., Weber, C., Tavakoli, M., Kismihók, G., and Pelucchi, M. (2022). Ontojob: Automated ontology learning from labor market data. In *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*, pages 195–200. IEEE.

Weichselbraun, A., Süsstrunk, N., Waldvogel, R., Glatzl, A., Braşoveanu, A. M., and Scharl, A. (2024). Anticipating job market demands—a deep learning approach to determining the future readiness of professional skills. *Future Internet*, 16(5):144.

Whelan, A., M. S. S. E. and Redmond, P. (2024). Skill requirements for emerging technologies in ireland. *ESRI Research Series 191.*

MAEVE: An Agnostic Dataset Generator Framework for Predicting Customer Behavior in Digital Marketing

William Ferreira da Silva Filho[1,2][a], Seyed Jamal Haddadi[1,2,3][b] and Julio Cesar dos Reis[1,2][c]

[1] *Hub de Inteligência Artificial e Arquiteturas Cognitivas (H.IAAC), Campinas, Brazil*

[2] *Artificial Intelligence Laboratory (Recod.ai), Campinas, Brazil*

[3] *Instituto de Computação, Universidade Estadual de Campinas (UNICAMP), Campinas, Brazil*

willpelicer@gmail.com, seyed@unicamp.br, jreis@ic.unicamp.br

Keywords: Customer Behavior Prediction, Digital Marketing, Event Logging, Dataset Generation.

Abstract: Data analysis plays a crucial role in assessing the effectiveness of business strategies. In Digital Marketing, analytical tools predominantly rely on traffic data and trend analysis, focusing on user behaviors and interactions. This study introduces a dataset generation framework to assist marketing professionals in conducting micro-level analyses of individual user responses to digital marketing strategies. The implemented proof of concept demonstrates that the framework can be integrated with enterprise software monitoring applications to ingest logs and, through appropriate configuration, generate comprehensive and valuable datasets. This research centers on the application of the framework for predicting customer behavior. The evaluation examines the extent to which the generated datasets are suitable for training various machine learning (ML) algorithms. The framework has shown promise in producing machine learning-ready datasets that accurately represent complex real-world scenarios.

1 INTRODUCTION

In the contemporary digital landscape, data is essential for business success and the validation of scientific theories. More specifically, large amounts of data can be refined into datasets, which, in turn, enable data-driven decision-making. What those datasets look like and how to create them depends on their source and platform (Renear et al., 2010).

Digital data sources include websites, mobile applications, and plugins. Data may also come from other sources, such as the Internet of Things, the financial market, or health care. Data can be applied across various domains, including agricultural monitoring, home surveillance, and the management of automotive or office environments. It may improve your user-targeted marketing or user experience. Data can enhance investment strategies and support the early detection of diseases. Data takes all kinds of shapes, coming from different systems in various industries. It is crucial to have access to a large amount of data to enable data-driven decision-making, analyze it to understand how data elements correlate to a question that needs answering, and plot it accordingly - creating a dataset.

Gathering data from different platforms infers different methodologies due to the differences in communication protocols, programming languages, data format, etc., making it challenging to create unique software to gather data from all of them. Data collection from a system requires the integration of specialized capabilities within the software or the development of auxiliary software dedicated solely to extracting system output.

This scenario drove the analytics market to build enterprise applications to monitor systems by capturing their data (for instance, Datadog (Datadog Documentation Authors, 2024) or Google Analytics). Their primary purpose is not to acquire data but to monitor and report specific system characteristics.

Many current analytics applications use macro-scale analyses because they provide metrics based on large-scale traffic data instead of user interaction. Monitoring applications such as Datadog enable micro-scale analysis by ingesting logs resulting from user interaction with graphical interfaces. Literature has shown platform-specific software to accumulate data (de Santana and Baranauskas, 2015) (Ma et al.,

[a] https://orcid.org/0009-0003-1763-1562

[b] https://orcid.org/0000-0001-6022-4143

[c] https://orcid.org/0000-0002-9545-2098

Filho, W., Haddadi, S. and Reis, J.
MAEVE: An Agnostic Dataset Generator Framework for Predicting Customer Behavior in Digital Marketing.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 433-440
ISBN: 978-989-758-716-0; ISSN: 2184-3228

2013) (Froehlich et al., 2007)(Rawassizadeh et al., 2013) (Pielot et al., 2014) (Ferreira et al., 2014). On a broader scale, enterprise applications accomplished much on data collection with platform agnosticism.

This article proposes MAEVE as a dataset generator framework. This study aims to answer the following research question: Can enterprise monitoring applications be leveraged to solve platform-agnosticism in dataset generation? Our study further investigates a novel ecosystem responsible for generating datasets based on API communication with those enterprise monitoring applications to generate datasets - assuming the enterprise application supports its platform. Such an ecosystem can help marketing strategies thrive with data-driven decision-making by providing a “plug-and-play” dataset generation framework for any software.

This research explores machine learning algorithms to experimentally analyze the framework’s results and the quality of the datasets as our metric of success. We demonstrate how the datasets generated by the framework are performed using different machine learning algorithms.

This study provides the following contributions:

- The conception and development of a novel framework “MAEVE” for micro-scale insights on user responses to digital marketing strategies.
- Integration with enterprise software monitoring applications as platform agnostic data sources to ingest logs and generate extensive and analyzable datasets.
- Demonstration of the dataset’s readiness for machine learning algorithms, reflecting real-world complexities and showcasing how the novel framework can enable data-driven decision-making.

The remainder of this article is organized as follows: Section 2 reviews the related works that laid the foundation for this research. Section 3 outlines the MAEVE framework proposal in detail, explaining its components and functionalities. Section 4 describes the experimental methodology used to evaluate MAEVE’s effectiveness, along with the results obtained. Section 5 discusses the key findings and highlights the open challenges in the field. Finally, Section 6 presents the concluding remarks and summarizes the contributions of this study.

2 RELATED WORK

Applied data science can be achieved mainly in two ways. The first is to use enterprise software directed to your needs, such as Google Analytics. Such tools provide you with data gathering solutions to improve business, such as access traffic to your website, peak access timelines, most used pages, etc. The second one is more specific and less friendly for people with no technology background, which is to build your data pipelines.

The objective of the MAEVE framework is to generate datasets useful to the user’s (the ”user” being the person who benefits from the framework’s dataset) specific needs in digital marketing. This allows users to configure the framework to generate datasets that facilitate predictions regarding client return rates, purchasing likelihood, and other key behaviors.

In practical terms, the user must identify and specify the location of the desired outcome within the system’s data (For instance, a log entry that records the precise moment a user interacts with the purchase button). Additionally, the user should define the interface interactions or characteristics that exhibit a correlation with that outcome. This approach enables the framework to identify relevant patterns and correlations in the data, thereby supporting predictive analytics.

Thus, MAEVE delivers an analytics solutions that for Digital Marketing professionals, empowering their decision-making with few configurations.

WELFIT (de Santana and Baranauskas, 2015) and Xiaoxiao Ma *et al.* (Ma et al., 2013) represent models of user-triggered event recorders. Aligning with the architectural principles outlined in both studies, logs are imported to establish independent modules. Moreover, the prevailing approach in mobile event logging, as seen in MyExperience (Froehlich et al., 2007), predominantly relies on operating system APIs to collect sensor and OS-specific events. However, such data does not align with the focus of this research, which prioritizes user interaction with application interfaces as the primary dataset for analysis.

As per software monitoring applications, a standard functionality exists in log management that allows the storage of logs from diverse platforms. Platforms like Datadog (Datadog Documentation Authors, 2024) showcase a promising solution, with distinct SDKs for various platforms converging into a unified log management system. This facilitates multi-platform event logging by being the platform itself, an event logger, and establishing a robust foundation for comprehensive monitoring purposes, particularly with Datadog’s (Datadog Documentation Authors, 2024) unique incorporation of Real User Monitoring (RUM) capabilities.

Moreover, our research underscores the need for an open-source ETL framework to generate digital marketing datasets across multiple platforms. While existing market ETL applications suffice for dataset generation, their platform-specific nature limits the research's objective (Informatica Power Center, 2024) (Talend Documentation Authors, 2024) (Microsoft, 2024).

Our proposed framework aims to fill this gap by being versatile, agnostic to data types, and exclusively utilizing NoSQL databases to align with its objectives. In summary, while WELFIT (de Santana and Baranauskas, 2015) and Xiaoxiao Ma *et al.* (Ma et al., 2013) highlight cutting-edge logging capabilities, the versatility, and comprehensiveness of enterprise monitoring applications like Datadog (Datadog Documentation Authors, 2024) position them as optimal choices for event logging in the proposed framework, which emphasizes multi-platform readiness and NoSQL compatibility.

The novelty in our proposed framework comes from creating an ecosystem that, joining the technologies presented above, can be used to create marketing-directed datasets on a platform-agnostic basis. The monitoring application brings state of the art in providing an event logger and a centralized, platform-agnostic log management system.

By creating MAEVE's modules based on API communications with monitoring applications, we abstract the implementation of that system, meaning that the event logger can be Datadog, Grafana - any application with an API for log ingestion. MAEVE's modules serve as the ETL that ingests logs, the products of which are the datasets. With this framework, with minimum knowledge of the data and minimum configuration, MAEVE allows a fast way of generating almost real-time datasets based on user interactivity with graphical interfaces.

3 MAEVE DATA GENERATOR FRAMEWORK

This research objective is to contribute to the context of digital marketing, by enabling the creation of datasets on a platform-agnostic basis, enabling data-driven decision making. The proposal is to create a dataset generation framework composed of three main modules: the log importer; the data normalizer; and the dataset generator. We named this framework as "*MAEVE*", which stands for Marketing Event Logging and Dataset Generation Framework. Figure 1 presents the relationship between those three modules. The following sections describes the chosen event logger and each component of the framework and its responsibilities.

3.1 Datadog: The Chosen Event Logger

For the implementation of the event logger abstraction, Datadog (Datadog Documentation Authors, 2024) was selected. Datadog is a robust software monitoring application designed to serve as a comprehensive platform for engineers to manage logs, and create alerts based on system performance, abnormal behavior, and more.

Several factors contributed to the decision to choose Datadog. Firstly, it offers a user-friendly experience with minimal setup requirements. Installation merely involves integrating its SDK into the system to be monitored and configuring a simple API and Application key. Once configured, Datadog seamlessly aggregates application logs.

Secondly, Datadog's log entries extend beyond simple text content. They encompass a wealth of contextual information such as the user's browsing activity, interacted elements, time zone, browser details, user session information, and more, providing invaluable insights.

Lastly, Datadog stands out from its competitors like Grafana and Google Analytics due to its extensive API. While many tools confine logs within their own ecosystems to promote platform lock-in, Datadog offers a well-documented API empowering users to access virtually any data sent to the platform.

It is important to notice that by choosing Datadog we leverage the state of the art from monitoring applications. Datadog is not conceptually an event logger. It's purpose is to monitor applications. However, to do so, it needs to import its logs, becoming, by extension, an event logger. We harvest that functionality from Datadog to create a platform-agnostic dataset generation framework, since Datadog provides SDK for several platforms, such as Android, iOS, Windows, Linux, etc. Thus, by connecting MAEVE with Datadog, we can generate datasets from a wide range of applications.

3.2 Importer

A pivotal aspect of this module is its abstraction. Leveraging the capabilities of an object-oriented language, we define interfaces and abstract classes that can be implemented to establish connections between the importer and any event logger. This design approach allows for flexibility in integration, as the event logger is not constrained to providing a specific API. Instead, it could be an event stream, files, a database, or any other data source.

The importer operates as a job-based application specifically tailored for API-based event loggers. This

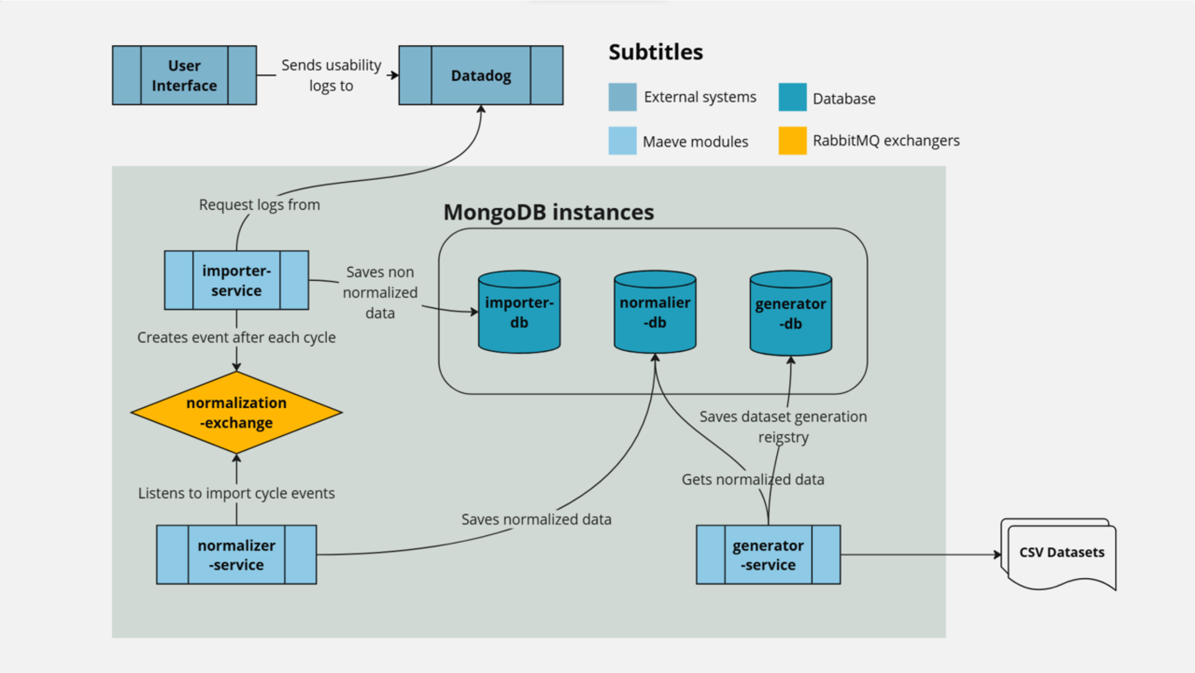


Figure 1: Component diagram of MAEVE.

design is essential due to the lack of real-time access to incoming logs. As a result, the importer employs its own CRON scheduled jobs. During each job execution, the importer queries Datadog (the event logger we have chosen for this experimentation) for all logs generated since the last job run up to the current timestamp. Each retrieved log from Datadog is then stored in its raw form within a document-oriented NoSQL database. If Datadog were to function as an event stream rather than an API, the need for scheduled jobs would be obviated, as we could subscribe to the stream and listen for incoming logs. However, given the versatility and applicability of job-based imports across various communication channels, this approach was chosen.c

Another crucial aspect of the importer is its role as the trigger for log normalization. With the logs securely within MAEVE's ecosystem, they become available for manipulation as needed. Additionally, the importer facilitates communication with other modules by employing an event-driven architecture implemented with RabbitMQ.

Upon successfully saving a log in the database, the importer dispatches a message to a RabbitMQ exchange containing the ID of the log. This notification signals to other modules that a new log is ready for transformation.

3.3 Normalizer

The normalizer module subscribes to the message exchange, to which the importer sends messages to. This is done to achieve an event-driven architecture (Cassandras, 2014), which is essential for the framework to be scalable. Upon receiving a message, the normalizer consumes it, initiating the normalization process for the log identified by the message's ID.

Normalization entails two configurable steps: a) determining relevant fields that will become features in the datasets and b) formatting field values. The first step can be configured within the module's settings, allowing users to define the desired structure for the log. The normalizer then removes unnecessary fields and retains only those designated for persistence.

The second configuration involves coding, employing a factory design pattern (Shvets, 2018) to implement a set of normalization rules. These rules are implemented as classes to clean and transform field values within the logs. For instance, rules could anonymize personal user data or standardize timestamps to a specific time zone. The factory design pattern ensures that all configured rules are applied to the log during normalization.

It is important to notice that this factory design pattern is crucial in MAEVE's architecture. Leveraging the abstraction capabilities provided by Java, this design pattern allows the normalizer to be the heart of MAEVE. This module allows developers to create essentially any rules to deal with the logs being ingested, making it a data engineering silver bullet.

After passing the two layers of data normalization, the normalized log is then saved in the normalizer's document NoSQL database instance. These logs represent the final form of the data and serve as the basis for dataset creation by the next module.

3.4 Generator

The generator module serves as the engine that leverages the storytelling capabilities inherent in normalized logs to address specific questions. In the context of this research, the question the generator is trying to answer through datasets is: will the user buy a product?

This module can read a normalized database, treat and organize the data based on its configuration. The configuration is a simple map of where the necessary fields can be found, and what field is the dataset meant to predict.

The generator's output is a CSV file in which each line refers to a normalized log, each column is a feature, and the final column is the binary response of what is being analyzed.

Utilizing abstraction and the factory design pattern, the generator empowers developers to create implementations of dataset generation in various formats. For this paper, CSV file implementation was chosen to facilitate experimentation. Bayesian prediction (Kruschke, 2014) with Python and the pandas framework (pandas Development Team, 2024), known to work well with CSV format, will be employed for testing the datasets.

3.5 External System and Usability Logs Generation

To evaluate the framework, it is necessary to find a suitable graphical interface system into which Datadog can be installed. To do so, keeping in mind the intention of generating marketing directed datasets, we used an open source marketplace UI.

MAEVE's inputs are logs, and to generate logs we need users interacting with the system's interface. For the generation of the logs, we mimic user interactivity using Cypress, a front-end integration testing framework, to create bots. Those bots are essentially automated tests that interact with the UI in a configured manner. The bots were configured to operate in three different behaviors: an assertive user to buy, a user that is not interested in the product and does not buy and a user that is frustrated by the UI and leaves the website.

4 EXPERIMENTAL EVALUATION

This experimental evaluation aims to apply five machine learning techniques to the created dataset to evaluate how the dataset generator framework is appropriate for predicting customer behavior in digital marketing.

4.1 Dataset

The dataset used in this experimentation comprises 15,000 rows and 25 features. The features can be seen on table 1. The features represent four aspects of a user session:

- **User Personal Data:** personal information on the user, such as gender, age, and for how long the user account has been active
- **Historical User Activity (Sessions):** features that show if the user was logged in during the session, the average amount of active sessions last month, etc.
- **Product Data:** Product rating, price, and important information that might lead to the purchase, such as is the product currently in the user's favorites list?
- **UI Interactivity Logs:** Most of the features are categorized as interactivity data, and they represent actions the user takes on the UI during the session, such as: did the user accesses the wallet, scrolls through the product, was there any frustration recognized from the usability pattern, how much time did the user spend on the product page?

The label distribution is divided into two distinct categories. The larger category, comprising 62.8% of the total, represents instances labeled as "Purchased." In contrast, the smaller category, accounting for 37.2% of the data, corresponds to entries labeled as "Not Purchased." This distribution is visually represented in a pie chart, highlighting the proportional difference between the two segments.

The next section details the machine learning techniques applied to this dataset.

4.2 Machine Learning Techniques

Since one of the research questions in this study is to predict whether a customer purchases a product, we used machine learning models to conduct this binary classification prediction. To this end, given the problem is a supervised learning problem, four classical methods, and a deep neural network model are chosen to solve this binary classification problem.

Table 1: The 25 features in the dataset generated by MAEVE.

Feature Name
session_id
logged
gender
age
home_page_access
product_accessed
is_product_favorite
product_view_time
home_page_rum_frustration_count
product_page_rum_count
item_in_cart
wallet_page_access
has_payment_method_registered
wallet_page_rum_frustration_count
sign_up_page_access
payment_page_rum_frustration_count
product_rating
product_price
tenure
num_sessions_last_month
total_spent_last_month
avg_time_on_product_pages_seconds
session_duration_minutes
product_page_scroll_depth
purchased

Support Vector Classifier (SVC). The Support Vector Machine (SVM) Classifier, or SVC, was incorporated due to its ability to identify the optimal hyperplane within a transformed feature space, thereby segregating classes by the widest margin possible (Cortes and Vapnik, 1995). SVM's utility in managing imbalanced datasets is underscored through its focus on maximizing margins while being minimally affected by the presence of majority classes. The implementation of SVC here involves both linear and Radial Basis Function (RBF) kernels.

KNeighbors (KNN). This non-parametric and lazy learning algorithm, classifies objects by aggregating the majority votes from their k closest neighbors within the feature space (Cover and Hart, 1967). To identify the optimal neighborhood size, the performance of the KNN algorithm was assessed using various k values (1, 3, 5, 7, and 9).

Gaussian Naive Bayes. Naive Bayes classifiers encompass a suite of algorithms for classification grounded in Bayes' Theorem. For continuous data, the Gaussian Naive Bayes approach posits that the feature values associated with each class follow a Gaussian distribution (Kamel et al., 2019).

Logistic Regression. This provides an approach for analyzing qualitative dependent variables that are categorical instead of continuous. It addresses the constraints of least squares regression in scenarios with binary or categorical outcomes by calculating the likelihood of particular events (De Menezes et al., 2017).

Deep Neural Network. Deep learning falls under machine learning techniques that utilize artificial neural networks to learn representations. A Fully Connected Feedforward Neural Network (FCNN) is utilized for binary classification tasks and is specially designed for such purposes. This network type is often known as a Dense Neural Network or, more generally, a Deep Neural Network (DNN) when it features multiple hidden layers.

4.3 Procedures

4.3.1 Data Labeling

To implement binary classification, the dataset needs to be labeled. For this reason, a criterion is defined to label the data which the mathematical formulation of the revised labeling criterion can be expressed as:

$$\text{Label} = \begin{cases} 1, & \text{if } (E \vee Q) \wedge (H \vee P) \\ 0, & \text{otherwise} \end{cases}$$

where:

$$\begin{aligned} E &= (V > 50) \vee (S > 90) \\ Q &= (D > 45) \vee (F < 5) \\ H &= (T > 100) \vee (N > 20) \\ P &= (R > 5) \vee (C < 600) \end{aligned}$$

Here, E, Q, H, and P represent conditions related to engagement, session quality, historical behavior, and product factors, respectively. The logical operators $\vee$ and $\wedge$ denote the logical OR and logical AND operations, respectively.

4.3.2 Data Preprocessing

Data Preprocessing encompasses loading the dataset, inspecting its structure, and splitting it into training (%70), validation (%20), and test sets (%10). Subsequently, categorical variables are encoded, while numerical features are standardized. This ensures uniform scaling across features. Finally, the dataset is prepared for model training, ensuring integrity and optimal utilization for subsequent optimization steps.

4.3.3 Model Creation and Fine-Tuning

In this phase, a neural network model with varying hyperparameters and other traditional machine

Table 2: Model Evaluation Metrics and Correct/Incorrect Classified Instances for Machine Learning Methods used.

Model	Recall	F1-score	Precision	Accuracy	Correctly/Incorrectly Classified Instances (%)	
KNN 1	0.70	0.70	0.70	0.70	69.50	30.50
KNN 3	0.73	0.73	0.73	0.73	73.43	26.57
KNN 5	0.76	0.75	0.75	0.75	75.50	25.50
KNN 7	0.76	0.75	0.76	0.76	76.13	23.87
KNN 9	0.77	0.75	0.76	0.76	76.53	23.47
GaussianNB	0.82	0.82	0.85	0.82	82.00	18.00
SVC_RBF	0.85	0.85	0.85	0.85	85.33	14.67
SVC_Linear	0.82	0.82	0.82	0.82	81.57	14.43
LR	0.80	0.80	0.80	0.80	80.20	19.80
DNN	0.88	0.88	0.88	0.88	87.78	12.22

learning models is dynamically generated. To optimize these models, Optuna (Akiba et al., 2019), an open-source automated hyperparameter optimization framework, provides a flexible and efficient platform. It automates the hyperparameter search process, leveraging advanced techniques such as Bayesian optimization to identify the optimal configurations.

4.4 Results

Table 2 presents the obtained results. Among the models analyzed, including KNN with different neighbors, GaussianNB, SVC with RBF and linear kernels, Logistic Regression (LR), and Deep Neural Networks (DNN), the DNN model achieved superior performance with the highest Recall, F1-score, Precision, and Accuracy at 0.88. It also exhibited the best classification performance, with 87.78% of instances correctly classified and an error rate of 12.22%. This indicates a significant advantage of deep learning techniques in handling complex classification tasks, highlighting their potential in predictive analytics and pattern recognition within diverse datasets.

5 DISCUSSION

This study addresses the challenge of designing and implementing an automated solution for generating high-quality datasets from user interaction logs across diverse platforms. Such datasets are essential for enabling precise, data-driven decision-making in digital marketing. The MAEVE dataset generator framework was developed as a flexible and scalable software tool, designed to transform raw, unstructured log data into structured datasets that accurately reflect real-world complexities. By capturing granular details of user interactions and behavior, MAEVE facilitates the extraction of actionable insights, enhancing the effectiveness of digital marketing strategies.

The framework presents several notable strengths. Firstly, fidelity, ensuring that the datasets generated preserve the integrity of the original data, maintaining the nuances and complexities necessary for robust analysis. Second, feature richness is achieved through the inclusion of a wide range of interaction features, thereby enhancing the efficacy of machine learning models in predicting outcomes. Third, customizability, allows users to adapt dataset generation processes to specific investigative needs or marketing contexts. Finally, scalability, ensures the framework's seamless integration with various machine learning methodologies, enabling efficient processing of large-scale datasets.

The experimental evaluation confirmed that the datasets produced by MAEVE are well-suited for training machine learning models, yielding high accuracy in predicting customer behavior. This underscores MAEVE's utility as an effective tool for data-driven research and digital marketing analytics. Its core attributes—fidelity, feature richness, customizability, and scalability—position the framework as an effective solution for generating realistic and machine learning-ready datasets applicable across diverse digital marketing environments.

6 CONCLUSION

Given the differences in capturing user interaction logs from different applications, platform-agnostic dataset generation is difficult to achieve. Moreover, given the platform differences, those data might look distinct and might not be normalized. In this scenario, predicting customer behavior based on graphical interface interactivity is costly for any new digital marketing endeavor. This study proposed a framework, MAEVE, that uses an abstraction for enterprise monitoring applications and created ETL modules for logs ingestion and normalization to, finally, generate digital marketing-directed datasets. Our framework enables (a) generate digital marketing-directed datasets based on user interactivity with graphical interfaces; and (b) to be platform agnostic, meaning that the same framework can be used to generate datasets for mobile, web, embedded applications, etc. Our solution contributes to the literature on predicting customer behavior while providing a technical approach that enables marketing experts and data scientists to have a quick start on their endeavors.

ACKNOWLEDGEMENTS

This work was supported by the Brazilian Ministry of Science, Technology and Innovations, with resources from Law nº 8,248, of October 23, 1991, within the scope of PPI-SOFTEX, coordinated by Softex and published Arquitetura Cognitiva (Phase 3), DOU 01245.003479/2024 -10. This work is also supported by the 'PIND/FAEPEX - "Programa de Incentivo a Novos Docentes da Unicamp" (#2560/23).

REFERENCES

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631.

Cassandras, C. G. (2014). The event-driven paradigm for control, communication and optimization. *Journal of Control and Decision*, 1(1):3–17.

Cortes, C. and Vapnik, V. (1995). *Support-vector networks*, volume 20. Springer.

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27.

Datadog Documentation Authors (2024). Datadog logs documentation. Online; accessed March 20, 2024.

De Menezes, F. S., Liska, G. R., Cirillo, M. A., and Vivanco, M. J. (2017). *Data classification with binary response through the Boosting algorithm and logistic regression*, volume 69. Elsevier.

de Santana, V. F. and Baranauskas, M. C. C. (2015). Welfit: A remote evaluation tool for identifying web usage patterns through client-side logging. *International Journal of Human-Computer Studies*, 76:40–49.

Ferreira, D., Goncalves, J., Kostakos, V., Barkhuus, L., and Dey, A. K. (2014). Contextual experience sampling of mobile application micro-usage. *MobileHCI 2014 - Proceedings of the 16th ACM International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 91–100. Cited By :99.

Froehlich, J., Chen, M. Y., Consolvo, S., Harrison, B., and Landay, J. A. (2007). Myexperience: A system for in situ tracing and capturing of user feedback on mobile phones. *Association for Computing Machinery*, page 57–70.

Informatica Power Center (2024). Informatica powercenter. Online; accessed March 20, 2024.

Kamel, H., Abdulah, D., and Al-Tuwaijari, J. M. (2019). *Cancer classification using gaussian naive bayes algorithm*. International Engineering Conference (IEC).

Kruschke, J. K. (2014). *Doing Bayesian Data Analysis: A Tutorial with R, JAGS, and Stan*. Academic Press; 2nd Revised ed, 2nd edition.

Ma, X., Yan, B., Chen, G., Zhang, C., Huang, K., Drury, J., and Wang, L. (2013). Design and implementation of a toolkit for usability testing of mobile apps. *Mobile Networks and Applications*, 18(1):81–97. Cited By :26.

Microsoft (2024). Sql server integration services (ssis). Online.

pandas Development Team (2024). pandas documentation. Online; accessed March 20, 2024.

Pielot, M., Church, K., and De Oliveira, R. (2014). An in-situ study of mobile phone notifications. *MobileHCI 2014 - Proceedings of the 16th ACM International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 233–242. Cited By :219.

Rawassizadeh, R., Tomitsch, M., Wac, K., and Tjoa, A. M. (2013). Ubiqlog: A generic mobile phone-based lifelog framework. *Personal and Ubiquitous Computing*, 17(4):621–637. Cited By :75.

Renear, A. H., Sacchi, S., and Wickett, K. M. (2010). Definitions of dataset in the scientific and technical literature. *Proceedings of the American Society for Information Science and Technology*, 47(1):1–4.

Shvets, A. (2018). *Dive Into Design Patterns*, volume 1. Refactoring.Guru.

Talend Documentation Authors (2024). Talend data integration. Online; accessed March 20, 2024.

Beyond Twitter: Exploring Alternative API Sources for Social Media Analytics

Alina Campan[a] and Noah Holtke
School of Computing and Analytics, Northern Kentucky University, Nunn Drive, Highland Heights, U.S.A.
{campana1, holtken1}@nku.edu

Keywords: Social Media Analytics, Federated Social Media Platforms, API-Scraping.

Abstract: Social media is a valuable source of data for applications in a multitude of fields: agriculture, banking, business intelligence, communication, disaster management, education, government, health, hospitality and tourism, journalism, management, marketing, etc. There are two main ways to collect social media data: web scraping (requires more complex custom programs, faces legal and ethical concerns) and API-scraping using services provided by the social media platform itself (clear protocols, clean data, follows platform established rules). However, API-based access to social media platforms has significantly changed in the last few years, with the mainstream platforms placing more restrictions and pricing researchers out. At the same time, new, federated social media platforms have emerged, many of which have a growing user base and could be valuable data sources for research. In this paper, we describe an experimental framework to API-scrape data from the federated Mastodon platform (specifically its flagship node, Mastodon.social), and the results of volume, sentiment, emotion, and topic analysis on two datasets we collected – as a proof of concept for the usefulness of sourcing data from the Mastodon platform.

1 INTRODUCTION

Social media and online social networks (OSNs) have been a primary means to spread and consume information for a while now, due to the low cost and high pervasiveness. Despite their negative aspects, such as the echo chamber effect, and their potential for the spread of misinformation and disinformation, the discourse on social media also has positive dimensions, as is reflective of real-world events and trends. This allows for the positive use of social media data in a multitude of application fields: agriculture, banking, business intelligence, communication, disaster management, disruptive technology, education, ethics, government, health, hospitality and tourism, journalism, management, marketing, understanding terrorism (Zachlod, 2022). Different analysis methods are being used, including sentiment analysis, topic discovery, word frequency analysis and content analysis (Zachlod, 2022); also, analysis methods are still being researched and developed that are capable of effectively handling massive amounts of social media data (Zachlod, 2022) with acceptable accuracy. Commercial tools for social media analysis are also available.

Despite the variety of application fields and analytic methods, the deployment of social media analysis frameworks follows similar "steps necessary to gain useful information or even knowledge out of social media"; these steps are discovery, tracking (or collection), preparation, and analysis (Stieglitz, 2018), (Zachlod, 2022). In the tracking step, data is collected from one (or more) social media platform(s), using the provided communication method (API, RSS, HTML scraping.) In a recent literature review, Zachlod reported that from 94 articles they reviewed, the social media platforms investigated in these research works were: Twitter (55 studies), Facebook (25 studies), Instagram (13 studies), YouTube (8 studies), TripAdvisor (8 studies), LinkedIn (4 studies), other - Foursquare, Google +, TikTok, WeChat, Sina Weibo (21 times) (Zachlod, 2022). Of all social media sites, Twitter used to be the most popular. Twitter was once the dominant social media platform among all others. This was due to its less stringent privacy controls compared to platforms like Facebook (as Twitter is a

[a] https://orcid.org/0000-0002-9296-3036

Campan, A. and Holtke, N.
Beyond Twitter: Exploring Alternative API Sources for Social Media Analytics.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 441-447
ISBN: 978-989-758-716-0; ISSN: 2184-3228

microblogging site designed for widespread dissemination of opinions, rather than communicating within a small group of friends), and in no small measure to its free API access for researchers. However, in recent years, there has been a shift towards a monetized model. Now, researchers must pay $100 for a subscription that permits sampling of only 10,000 messages per month. Given the uncertain future of Twitter's accessibility for academic research, an investigation of alternative social media sources and their potential for research is worth investigating. In this work, we are considering one of the new social media platforms, the federated Mastodon platform, and specifically its flagship node, Mastodon.social. We explored its API technology, scraped two datasets focused on two different topics for a short window of time, analyzed the daily volume, sentiment valence, and emotional content of the two datasets – as a proof of concept for the usefulness of sourcing data from the Mastodon platform.

2 RELATED WORK: MASTODON.SOCIAL

Mastodon is a social media service that has become popular as an alternative to X (formerly Twitter) since its inception in 2016. Users engage with the platform by posting short-form content and engaging in conversations in comment feeds. Communities consist of self-hosted servers, often owned and operated by users of the platform, which integrate fully with all other Mastodon servers in the network. Each of these "instances" maintains its own community standards, policies, and content moderation. Users are free to join whichever instance they choose, but their account retains the ability to browse all public content on the platform. This federated model of content hosting has contributed to the development of a diverse range of communities.

The acquisition of Twitter by Elon Musk has contributed to a significant increase in the size of Mastodon's user base, growing by as many as 700,000 accounts between October and December in 2022. Its active user base currently sits around 1 million active users, with historic highs around 2.5 million. Accounting for a constant influx of new publications, a high volume of decentralized instances, and the distinction between public and private content, it is difficult to estimate the total volume of unique content on Mastodon.

“Mastodon.social is one of the many independent Mastodon servers you can use to participate in the fediverse” (Mastodon.social, 2024). Mastodon.social is considered the flagship instance and sits currently (July 13, 2024) at 226K active users.

The figure below shows the evolution of numbers of Mastodon servers, users, and active; these statistics are available online at (Khun, 2024). The most recent statistics from (Mastodon statistics, 2024) show the numbers of servers at 9168, users at 8,741,802, and active users at 854,905, on July 13, 2024.

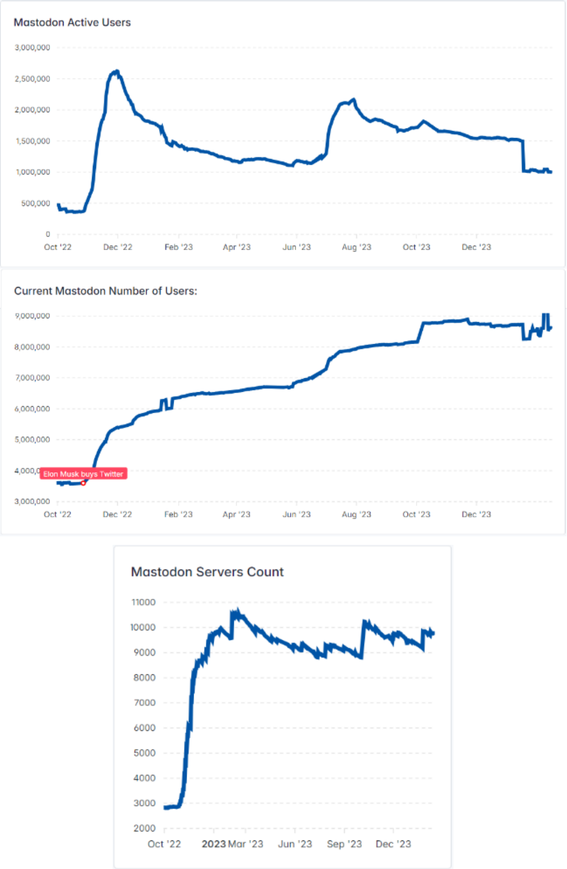


Figure 1: Number of Mastodon servers, users, and active users. Images captured from interactive graphics at https://mastodon-analytics.com/ (Khun, 2024).

The Guardian article (Nicholas, 2023) provides a comprehensive timeline of how the Mastodon user base evolved in relation to key events involving Twitter. Nevertheless, the challenging process of migrating communities has so far prevented Mastodon from gaining momentum and achieving widespread adoption as a mainstream social media platform. However, even with a significantly smaller user base, with more niche communities compared with the broader audience of Twitter, it is still worth investigating Mastodon as an alternative source of data for social media analysis; that is, given its considerable advantages, such as higher message character limit, federated architecture, chronological message feed (Lamaj, 2023), and free API access.

An alternative approach to overcome the API limitations newly imposed by social media platforms is to instead collect data by web scraping. However, this approach requires special web tools and add-ons such as BeautifulSoup and Selenium, and the data collection must carefully address legal and ethical concerns (Harrell, 2024).

Social media analysis comprises a large variety of analysis methods, but data is usually sourced from mainstream social media platforms (Zachlod, 2022). Until now, little work has been done on tracking and analyzying data from alternative social media platforms, such as Mastodon. In (David, 2023), the rtoot package is presented, that can be used to collect statuses (a.k.a. toots) from Mastodon and perform some analytics (such as comparing the length of toots from iOS and Web.) In this work, we perform a more thorough examination and investigate if there is value in conducting an analysis of data sourced from Mastodon.social: can tasks such as sentiment, emotion, and topic analysis reveal meaningful trends that are reflective of real-world events?

3 METHODOLOGY

In Figure 2, we show the steps we took to collect Mastodon data and analyze it. The steps are framed in the social media analysis framework presented in (Stieglitz, 2018) (Zachlod, 2022). Our methodological approach follows the steps in (Zachlod, 2022) and (Stieglitz, 2018), therefore, by adequately adjusting the tracking/collection step, the analytical process can be adapted to function with an alternative social media source. We explain each step in detail in the following sections.

3.1 Mastodon API

Mastodon provides access to its data via REST API. We used the Mastodon.py Python wrapper for the Mastodon API to interact with the Mastodon social network. The session.timeline() function was used to collect all messages (called statuses) marked public and whose content string contained one of a set of keywords; Similarly, the session.timeline_hashtag() function was used to collect those statuses marked public and matching one of a set of hashtags. While the account holding the access token for this data collection was hosted on the Mastodon.social instance, we could still access public data originating from any instance on the federated network. All unique statuses found matching any of the hashtags or keywords were collected and stored in a mongoDB collection for analysis. We chose to focus on two distinct topics: the 2024 US election and competing social media platforms. Each topic was tracked for 7-14 days, and we collected 20,064 social media related statuses, and 6,904 US election related statuses. Table 1 shows the keywords and hashtags that we used for tracking matching statuses for the two selected topics:

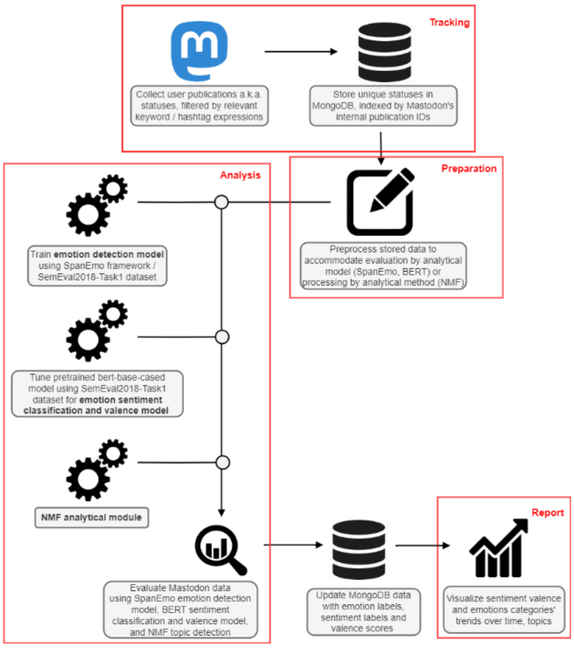

Tracking
Collect user publications a.k.a. statuses, filtered by relevant keyword / hashtag expressions
Store unique statuses in MongoDB, indexed by Mastodon's internal publication IDs
Preparation
Preprocess stored data to accommodate evaluation by analytical model (SpanEmo, BERT) or processing by analytical method (NMF)
Analysis
Train emotion detection model using SpanEmo framework / SemEval2018-Task1 dataset
Tune pretrained bert-base-cased model using SemEval2018-Task1 dataset for emotion sentiment classification and valence model
NMF analytical module
Evaluate Mastodon data using SpanEmo emotion detection model, BERT sentiment classification and valence model, and NMF topic detection
Update MongoDB data with emotion labels, sentiment labels and valence scores
Report
Visualize sentiment valence and emotions categories' trends over time, topics

Figure 2: Experimental framework for Mastodon.social data collection and analysis.

Table 1: Keywords and hashtags used for data tracking on Mastodon.social.

Topic	Hashtag list	Keyword list
Social Media	BlueSky, JackDorsey, Facebook, MarkZuckerberg, Meta, Threads, Twitter, ElonMusk, Musk, TwitterMigration, TwitterExodus, X, Xodus, Truth, TruthSocial	Jack Dorsey, Mark Zuckerberg, Zukerberg, Elon, Musk, Elon Musk, BlueSky, Facebook, Meta, Threads, Twitter, X, Truth Social
US Election	POTUS, President, Biden, Democrats, Election2024, Trump, Republicans, USelection, Vote, VoteBlue, VoteBlue2024, Voting	

3.2 Analytical Methods

We conducted several types of analysis: sentiment class and valence prediction, emotion analysis, and topic discovery.

Sentiment classification is a type of analysis where each message is predicted to belong to one of several predefined classes (positive, neutral, negative), based on its content. Similarly, ***sentiment valence*** prediction associates to each message a numerical score from a range (such as [-1,1]), where the lower the score, the more negative the message is, and the higher the score, the more positive the message is; scores around 0 indicated a neutral or mixed emotional state in the text. Both types of tasks can be approached with a variety of methods (such as VADER (Hutto, 2014), linear regression etc.); more recently, methods based on LLMs have been used for this purpose. We utilized for both tasks a pretrained BERT model with three sentiment classes (negative, positive, and neutral) (Devlin, 2019) (Rathi 2020) that we tuned on a dataset from the SemEval2018-Task1 (Mohammad, 2018). Specifically, we used the tweet set combined from the 2018-Valence-oc-En-train.txt and 2018-Valence-oc-En-dev.txt files, where messages with "Intensity Class" equal to -3, -2, or -1 were assigned to the "negative" class, messages with "Intensity Class" equal to 1, 2, or 3 were assigned to the "positive" class, and the "neutral" class consisted of all messages with "Intensity Class" equal to 0.

The tuned BERT model was then used to predict sentiment class labels and valences for the statuses in our US election and social media Mastodon datasets.

Emotion prediction is tasked with determining which emotions from a given set are present in a message. We followed the approach from SpanEmo (Alhuzali, 2021) (Alhuzali, 2021a) and trained a SpanEmo model on the data from 2018-E-c-En-train.txt and with validation data the 2018-E-c-En-dev.txt (Mohammad, 2018). The SpanEmo model obtained was used to predict the expression of anger, anticipation, disgust, fear, joy, love, optimism, hopeless [sic], sadness, surprise, and trust emotions in US election and social media Mastodon datasets.

Topic analysis attempts to identify topics or themes in a collection of texts. We used non-negative matrix factorization for identifying topics in our two data collections (Greene, 2017).

All of these analytical methods have been tested for accuracy with good results in other works (such as (Alhuzali, 2021)), and we will not include metrics to reflect their validity in this paper.

4 EXPERIMENTAL RESULTS

Figures 3 and 4 show the sentiment classes and the emotion classes respectively, over time, for the periods during which we collected the respective datasets. What is shown is the daily number of unique statuses as reflected in each sentiment class or each emotion category. The sentiment classes are disjoint, i.e. each message belongs exclusively to one sentiment class. The emotion classes overlap, i.e. each status may display several emotions.

As seen in Figures 3 and 4, there are several peaks and valleys in the sentiment and emotion graphs for the two datasets. For example, anger and disgust emotions peaked in the Social Media dataset on March 5, 2024. That is reflected in the following messages shown in Table 2, which indicate users' reactions to an ongoing service outage at Meta that impacted Facebook, Instagram, and Threads, among various other microservices.

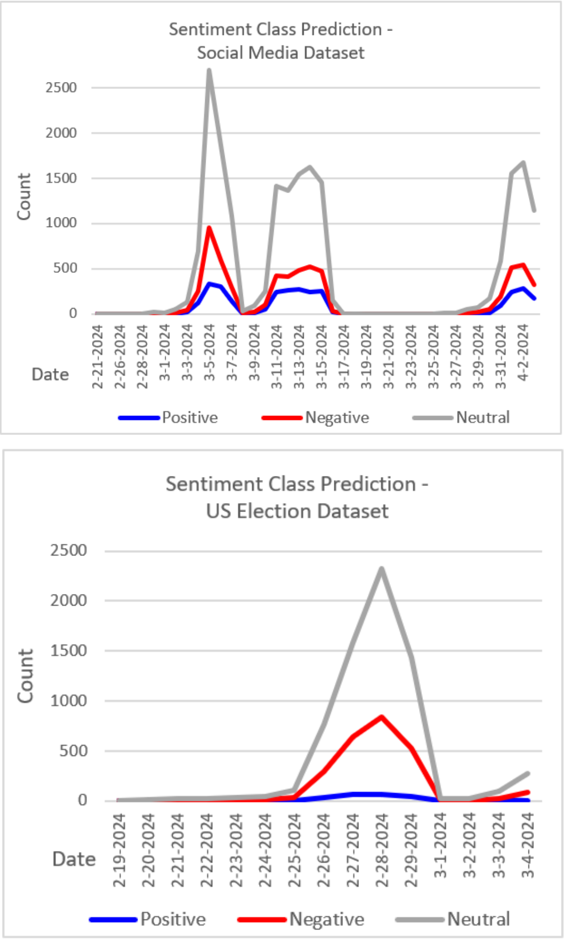


Figure 3: Sentiment classes, daily counts.

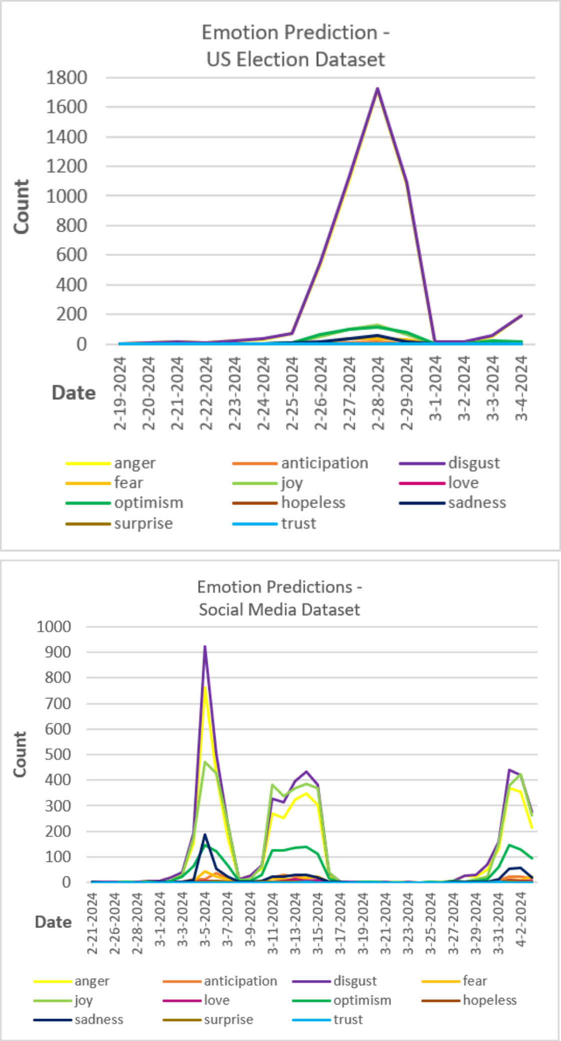


Figure 4: Emotion classes, daily counts.

Table 2: Messages during 2024-03-05 peak.

Date	Message Content	Sentiment (predicted)	Emotions (predicted)
24-03-05 T15:30:56	“Bloody timing of #Facebook going down just after I'd replied to someone in a private message that I've not spoken to for ages instantly making me think I'd fallen victim to some * hack.”	Negative	Anger, disgust
24-03-05 T15:33:03	“MAJOR outage at Meta at the moment. Got booted out of my Facebook account, can't login. Instagram seems to be similarly affected. #Facebook #Instagram #Meta #Outage”	Negative	Anger, disgust, sadness
24-03-05 T15:45:25	“Did Elon buy Facebook?”	Neutral	(null)

We also looked at how sentiment and emotion classes overlapped – which, to our knowledge, has not been investigated before. By verifying that each sentiment class maps into *expected* emotion classes also proves the validity of the independent methods applied for sentiment detection (BERT) and emotion detection (SpanEmo.) For example, anger, disgust, and fear are reasonably associated with negative sentiments; whereas joy, love, and optimism are associated with positive sentiments. Tables 3 and 4 illustrate how the different emotion classes overlap with the negative, positive, and neutral (largely irrelevant and ignored) classes. We highlighted the significant majority sentiment class for each emotion, and we can see that each emotion has highest overlap with the expected class between positive and negative classes:

Table 3: Sentiment and emotion classes overlap in the Social Media dataset.

BERT label	Negative	Neutral	Positive
anger	**2348**	2045	50
anticipation	6	133	**132**
disgust	**2735**	2632	72
fear	**116**	99	7
joy	57	2494	**2053**
love	2	23	**60**
optimism	37	771	**784**
hopeless	**4**	0	0
sadness	**433**	134	7
surprise	**10**	**12**	**6**

Table 4: Sentiment and emotion classes overlap in the US Election dataset.

BERT label	Negative	Neutral	Positive
anger	**2189**	2708	23
anticipation	2	30	**7**
disgust	**2236**	2736	20
fear	**95**	35	0
joy	11	253	**148**
love	0	3	**3**
optimism	7	285	**124**
hopeless	**1**	0	0
sadness	**95**	46	0
surprise	**3**	12	0

Figure 5 shows the daily average sentiment value for the observed datasets, in the observed time windows. The values are negative for all days in the US Elections dataset, and mostly negative or neutral (around 0) for the Social Media dataset. Again, peaks and valleys are noticeable, but not all are significant – that is because some of these points represent a very small number of messages; for example, the most negative point in the Social Media dataset is recorded for March 23rd, when there were only 4 statuses collected on the topic, 1 neutral, and 3 negative. On March 5th, when there was a peak of positive and negative counts, the overall average valence shown is -0.11, since the valences of the statuses in two sets, the positive and negative, largely cancel each other out. So, the volume of messages that contribute to the average should be taken into consideration when the average sentiment valence is interpreted.

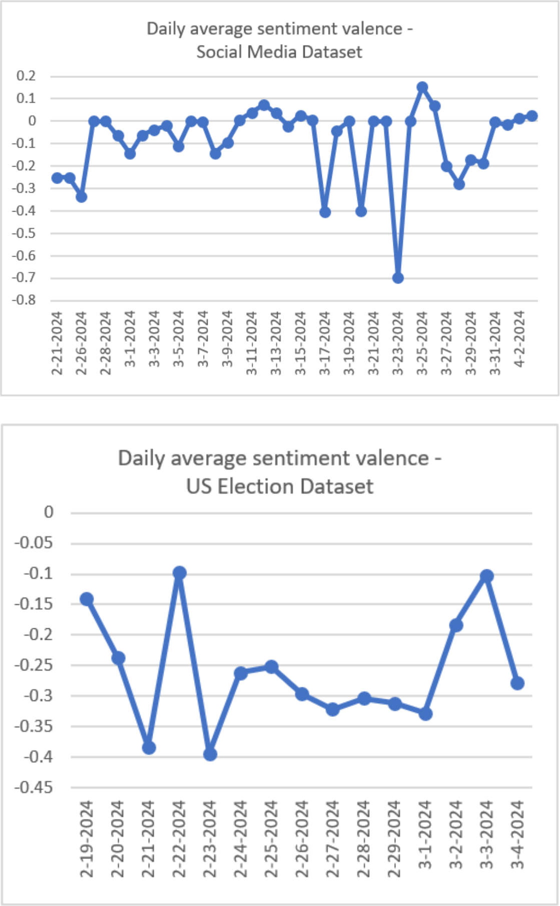


Figure 5: Sentiment valence, daily averages.

Finally, we performed topic analysis using non-negative matrix factorization (NMF) (Greene, 2017), (NMF documentation for scikit-learn, 2024) on a subset of 1755 statuses from the Social Media dataset for March 5, 2024 (the day with the maximum volume of messages in the observed interval), and which had the language specified as English (although some of these were actually in a different language). Figure 6 shows a t-SNE projection representing the words similarities obtained during topic analysis for this dataset:

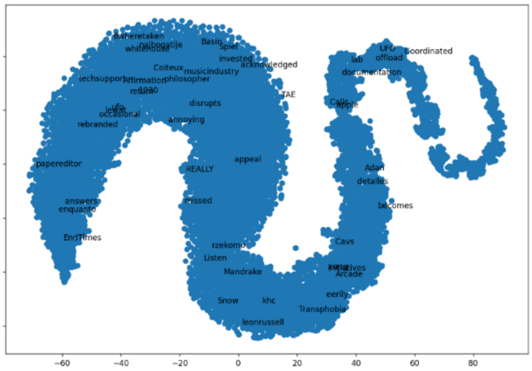


Figure 6: Words in Word2Vec model, t-SNE projection.

The representative terms determined for the NMF model; four topics are shown below. The number of topics was selected based on the coherence and separation of the topic terms:

```
Topic 1: https, com, news, score, id,
item, ycombinator, url, title,
discussion
Topic 2: value, form, labour, marx,
exchange, like, one, production,
bailey, commodity
Topic 3: facebook, instagram, meta,
outage, com, threads, www, https, 2024,
users
Topic 4: bluesky, network, also, nbsp,
fediverse, people, data, app, website,
protocol
```

Topic 3 reflects the Facebook outage that occurred on March 5, 2024.

5 CONCLUSIONS AND FUTURE WORK

Our experimental results show that there is value in analyzing data extracted from new and alternative social media platforms such as Mastodon. Analytical tasks such as emotion, sentiment, topic analysis can still reveal trends and identify reflections of real-world events.

In the future, we want to observe and compare various social media platforms for data completeness, quality, volume. We want to investigate how data collected from Mastodon, Threads, BlueSky, and other federated (or soon-to-be federated) platforms compares with that from the mainstream social networks in terms of discourse, sentiment, topics etc. – are they similar or very different? We also plan to research the difference in the data value depending on the collection method (RSS, API, HTML scraping.)

REFERENCES

Zachlod, C., Samuel, O., Ochsner, A., Werthmüller, S. (2022). Analytics of social media data – State of characteristics and application. In *Journal of Business Research, Vol. 144, May 2022, pp. 1064-1076*.

Stieglitz, S., Mirbabaie, M., Ross, B., Neuberger, C. (2018). Social media analytics – Challenges in topic discovery, data collection, and data preparation. In *International Journal of Information Management, Vol. 39, April 2018, pp.156-168*.

Mastodon statistics (2024), online at https://api.joinmastodon.org/statistics, visited July 2024.

Khun, E. (2024) - @erickhun@mastodon.social, Mastodon growth dashboard, online at https://mastodon-analytics.com/, visited July 2024.

Nicholas, J. (2023). Elon Musk drove more than a million people to Mastodon – but many aren’t sticking around, online at https://www.theguardian.com/news/datablog/2023/jan/08/elon-musk-drove-more-than-a-million-people-to-mastodon-but-many-arent-sticking-around, Jan 7, 2023.

Mastodon.social (2024), online at https://mastodon.social/about, July 2024.

Lamaj, D. (2023), Twitter Vs. Mastodon: Which is a Better Alternative? (Pros and Cons), online at https://publer.io/blog/twitter-vs-mastodon/, July 2023.

Harrell, N.B., Cruickshank, I., Master, A. (2024). Overcoming Social Media API Restrictions: Building an Effective Web Scraper, In *Proceedings of the ICWSM Workshops*, June 2024.

David Schoch, D., and Chan, C.-H. (2023). Software presentation: Rtoot: Collecting and Analyzing Mastodon Data. In *Sage Journal of Mobile Media & Communication, Volume 11, Issue 3, 575-578, https://doi.org/10.1177/20501579231176678*

Hutto, C. and Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media. 8 (1)*.

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, June 2019.

Rathi, P. (2020). Sentiment Analysis using BERT, code repository, online at https://www.kaggle.com/code/prakharrathi25/sentiment-analysis-using-bert.

Mohammad, S., Bravo-Marquez, F., Salameh, M. and Kiritchenko, S. (2018). SemEval-2018 Task 1: Affect in Tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation, pages 1–17, New Orleans, Louisiana*, June 2018.

Alhuzali, H. and Ananiadou, S. (2021). SpanEmo: Casting Multi-label Emotion Classification as Span-prediction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, April 2021.

Alhuzali, H., (2021a). SpanEmo, code repository, online at https://github.com/hasanhuz/SpanEmo.

Greene, D. (2017). Topic modelling with Scikit-learn. Presented at PyData Ireland, September 2017, github repository online at: https://github.com/derekgreene/topic-model-tutorial/

NMF documentation for scikit-learn (2024), online at https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html, visited 2024.

Optimizing Federated Learning for Intrusion Detection in IoT Networks

Abderahmane Hamdouchi[a] and Ali Idri[b]
Mohammed VI Polytechnic University, Benguerir, Morocco
{Abderahmane.hamdouchi, ali.idri}@um6p.ma

Keywords: Intrusion Detection System, Federated Learning, Deep Learning, Netflow, IoT, Cybersecurity.

Abstract: The Internet of Things (IoT) involves billions of interconnected devices, making IoT networks vulnerable to cyber threats. To enhance security, deep learning (DL) techniques are increasingly used in intrusion detection systems (IDS). However, centralized DL-based IDSs raise privacy concerns, prompting interest in Federated Learning (FL). This research evaluates FL configurations using dense neural networks (DNN) and convolutional neural networks (CNN) with two optimizers, stochastic gradient descent (SGD) and Adam, across 20% and 60% feature thresholds. Two cost-sensitive learning techniques were applied: undersampling with binary cross-entropy and weighted classes using weighted binary cross-entropy. Using the NF-ToN-IoT-v2 dataset, 16 FL configurations were analyzed. Results indicate that SGD, combined with CNN and the Undersampling technique applied to the top 7 features, outperformed other configurations.

1 INTRODUCTION

In the current digital landscape, the volume of data generated and stored has surged dramatically, driven by the decreasing cost of storage and increasing tendency to record every digital interaction. This data proliferation is further exacerbated by the growing adoption of Internet of Things (IoT) devices, including smart home technologies, the expansion of smart cities, and advancements associated with Industry 4.0. For big data companies, insights derived from IoT devices are invaluable, rendering data a critical asset that requires robust protection (Atharvan et al., 2022).

Intrusion detection systems (IDS) are essential in securing IoT environments. The integration of machine learning (ML) into an IDS enhances threat detection capabilities, but ML models often encounter challenges when dealing with imbalanced data, leading to complications in model design and an increased risk of overfitting (de Zarzà et al., 2023; Thakkar & Lohiya, 2023). Several strategies have been investigated to address these challenges: (1) cost-sensitive learning, which includes techniques such as cost-sensitive resampling (oversampling and undersampling (Luengo et al., 2011)) to adjust data distribution; (2) cost-sensitive algorithms, where ML algorithms are modified to incorporate a cost matrix, although this approach is time-consuming (Lomax & Vadera, 2013); (3) cost-sensitive ensembles, which combine predictions from traditional ML models while accounting for misclassification costs (Krawczyk et al., 2014; Tao et al., 2019); and (4) cost-sensitive learning in deep learning (DL), where model training involves adjusting weights using standard loss functions such as weighted binary cross entropy (WBCE) (Ho & Wookey, 2020), specifically developed to address the challenges posed by imbalanced data (Dina et al., 2023; Kerkhof et al., 2022).

However, early IDS systems were hindered by their lack of adaptability and slow response times to emerging threats, leaving them exposed for prolonged periods (Murphy, 2018). To address these limitations, more advanced IDS systems have begun to employ basic ML models to autonomously learn and identify new threats. Although the incorporation of ML has improved the accuracy of attack detection, most ML-based IDS systems remain centralized. This centralization involves a single organization collecting and processing data from multiple devices to train its ML models, raising significant privacy

[a] https://orcid.org/0009-0002-6623-8276
[b] https://orcid.org/0000-0002-4586-4158

Hamdouchi, A. and Idri, A.
Optimizing Federated Learning for Intrusion Detection in IoT Networks.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 448-456
ISBN: 978-989-758-716-0; ISSN: 2184-3228

concerns. This is particularly relevant in IoT environments such as smart wearables and healthcare devices, where sensitive and large volumes of data are at stake. Consequently, there is a growing demand for decentralized approaches to data management (McMahan et al., 2017).

To address the privacy concerns inherent in centralized ML approaches, federated learning (FL) (McMahan et al., 2017) was introduced in 2016. FL allows multiple devices, often referred to as clients or parties, to collaboratively train a model without sharing their data directly. Instead, these devices send model updates to a central entity known as an aggregator or coordinator, where the updates are combined to refine a global model. This approach is designed to enhance privacy by ensuring that the data remains local to the devices, thereby reducing the risk of exposure. FL achieves this by transmitting only the gradients instead of the raw data itself. The local training occurs on each device using its own dataset, and only the computed model parameters are sent to the central server. As a result, sensitive information never leaves the device, significantly reducing the risk of interception or unauthorized access during transmission.

Recent efforts have focused on developing FL-based IDS for IoT environments (Hei et al., 2020; Nguyen et al., 2019; Thu Huong et al., 2020). Despite this progress, many proposed approaches do not adequately address the challenges of imbalanced data, leading to models that are prone to overfitting. Additionally, these methods often fail to evaluate the effectiveness of different FL optimizers and DL base learners, or consider the impact of varying feature sets. Furthermore, the review (Agrawal et al., 2021) highlighted the challenges of implementing FL in IoT settings, but failed to provide concrete recommendations for enhancing IDS with FL or critically assessing the proposed solutions. This gap in detailed analysis poses challenges for cybersecurity experts in identifying the key issues associated with integrating FL into an IDS for IoT.

To address the existing research gap, this study investigates 16 FL configurations for IDS in IoT contexts, specifically employing deep learning (DL) through dense neural networks (DNN) and convolutional neural networks (CNN). These configurations (16 = 2 optimizers for the FL server × 2 DL architectures × 2 cost-sensitive configurations × 2 feature thresholds) explore the impact of different cost-sensitive learning approaches (resampling based on undersampling and weighted classes based on WBCE) along with the selection of FL optimizers and the number of features. The study utilized the NF-ToN-IoT-v2 dataset to evaluate the effectiveness of FL with various cost-sensitive setups, feature numbers, and FL optimizers. We conduct 16 experiments: eight with the dataset's original distribution using WBCE, and eight with undersampled data to balance attack and non-attack instances among participants using binary cross entropy (BCE). The study assesses the results using two FL optimizers: stochastic gradient descent (SGD) (Amari, 1993) and Adam (Zhang, 2019), with two feature thresholds (20% and 60%). The performance of the FL models was evaluated in binary classification tasks over 100 optimization rounds using four performance criteria: accuracy, AUC, precision, and recall. The Scott–Knott (SK) statistical test (Scott & Knott, 1974) and the Borda count (BC) voting system (Saari, 2001) were employed to compare and rank the models.

This study addresses the following research questions (RQs):

- **(RQ1).** What is the optimal choice between SGD and Adam optimizers in the context of FL for attack detection?
- **(RQ2).** What is the best FL configuration for detecting attacks across various settings?

The key contributions of this study are as follows:

1. Determining the best optimizer for FL in the context of intrusion detection using the NF-ToN-IoT-v2 dataset.
2. Identifying optimal FL setup for different configurations.
3. Developing a generalized FL model for IDS applicable to various IoT datasets.

The remainder of this paper is organized as follows. Section 2 provides a review of related literature. Section 3 presents the data used in this study. Section 4 outlines the research methodology. Section 5 presents the results and discussion of the experiments. Finally, Section 6 presents the conclusions and suggests directions for future research.

2 RELATED WORK

Several significant studies have explored anomaly detection across various domains, with a particular emphasis on IoT, utilizing different FL methodologies. This section provides an overview of key research efforts that have employed FL for intrusion detection within IoT environments.

Friha et al. (Friha et al., 2022) proposed an FL-based IDS (FELIDS) aimed at securing agricultural IoT infrastructure by ensuring data privacy through localized learning. To defend against attacks on Agricultural IoT systems, FELIDS leverages three DL classifiers: DNN, CNN, and recurrent neural networks (RNN). The effectiveness of the system was evaluated using three datasets: CSE-CIC-IDS2018, MQTTset, and InSDN. The findings indicate that FELIDS outperforms traditional centralized machine learning (non-FL) methods by offering enhanced data privacy for IoT devices and achieving an accuracy of 99.71% with the CNN classifier on the InSDN dataset, 89.56% with the RNN classifier on the MQTTset dataset, and 94.15% with the RNN classifier on the CSE-CIC-IDS2018 dataset. Mothukuri et al. (Mothukuri et al., 2022) proposed a novel approach that leverages FL to train gated recurrent units (GRUs) models while ensuring that data remains on local IoT devices. Only the learned weights of the model are shared with the central server of the FL. In addition, the method incorporates an ensemble technique to aggregate updates from multiple sources, thereby improving the accuracy of the global ML model. The results demonstrated that this approach not only outperforms traditional centralized ML methods in safeguarding data privacy but also achieves an overall average accuracy of 90.255% in attack detection. Idrissi et al. (Idrissi et al., 2023) introduced Fed-ANIDS, a Network IDS that combines ML-based anomaly detection (AD) with FL to address privacy concerns inherent in centralized models. The system detects intrusions by calculating an intrusion score based on the reconstruction error of normal traffic using various AD models, including simple autoencoders, variational autoencoders, and adversarial autoencoders. The method was evaluated using three datasets: USTC-TFC2016, CIC-IDS2017, and CSE-CIC-IDS2018. The results indicated that autoencoder-based models outperformed other generative adversarial network-based models, and that the FedProx aggregation framework was more effective than FedAVG. The proposed method achieved peak accuracies of 99.95%, 93.54%, and 94.48% for the USTC-TFC2016, CIC-IDS2017, and CSE-CIC-IDS2018 datasets, respectively.

3 EXPERIMENTAL DESIGN

This section describes the datasets, performance metrics, and methodology employed in the empirical evaluations conducted in this study.

3.1 Dataset Description

In this study, 43 NetFlow features conforming to version 9 standard were extracted and labeled from the ToN-IoT (Alsaedi et al., 2020) dataset using the nProbe tool by Ntop. The resulting dataset, designated as NF-ToN-IoT-V2 (Sarhan et al., 2021), comprises 16,940,469 instances labeled as either attack or no-attack, reflecting a significant class imbalance, with 36% labeled as no-attack and 64% as attack. This data was chosen because it consists of real IoT traffic with simulated attacks. Furthermore, the “Pcap” files are labeled and available as open source, providing a foundation for developing a generalized model applicable to all NetFlow V9 data types.

3.2 Performance Measures

To evaluate the effectiveness of the proposed binary classification models, we employed accuracy, recall, precision, and area under the receiver operating characteristic curve (AUC) as evaluation metrics (Naidu et al., 2023). These metrics were selected owing to their widespread use and acceptance in the field.

3.3 Statistical Test and Borda Count

- **Scott Knott (SK)** is a clustering algorithm commonly used to compare multiple groups in analysis of variance studies. It addresses the issue of overlapping groups by starting with all observed mean effects grouped together and iteratively dividing these groups into smaller subgroups, ensuring that no two subgroups share any common members (Scott & Knott, 1974).
- **Borda Count (BC)** is a voting method in which voters rank candidates according to their preferences. Each candidate receives points based on their rank, with the lower ranks earning fewer points. The points are then aggregated, and the candidate with the highest total is declared as the winner. In this study, the Borda count method was used to identify the top-performing model, treating all performance measures equally (Saari, 2001).

3.4 Methodology

Figure 1 illustrates the methodology employed to assess and compare the effects of different FL optimizers, DL architectures, cost-sensitive learning setups, and feature thresholds on the detection capabilities of FL-based IDS. We evaluated the performance of two FL optimizers, SGD and Adam, with two different feature thresholds (20% and 60%) and two cost-sensitive learning setups (undersampling and WBCE) over 100 rounds using SK and BC voting systems. The experimental procedure included the following steps:

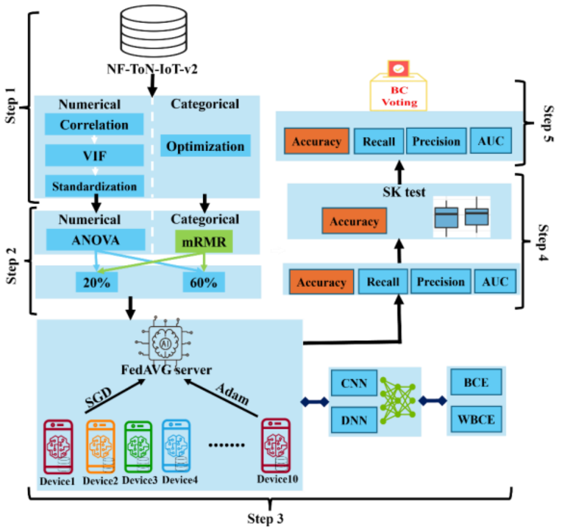


Figure 1: Experimental process.

- **Step 1.** Prepare the raw data by removing missing values, duplicate rows, and unnecessary attributes. This process includes reclassifying categorical features and standardizing numerical features.
- **Step 2.** involves the application of two FS techniques: mRMR for categorical features and ANOVA for numerical features. These methods were selected based on a comparative analysis of FS and ML techniques on IDS datasets (Amiri et al., 2011; Shakeela et al., 2021; Tao et al., 2019; Zouhri et al., 2023). Additionally, two feature thresholds (20% and 100%) were implemented, as recommended in previous studies (Dhaliwal et al., 2018; Kurniabudi et al., 2020; Nakashima et al., 2018). This process resulted in the creation of two dataset variations.
- **Step 3.** Set up a simulated IoT network by creating virtual instances using TFF based on the DNN and CNN architectures. We utilized 10 devices, each referred to as *Device$_i$*, and configured two optimizers for the central FedAVG server instance. This instance facilitates the exchange of DL model parameters between the mobile IoT devices and the central FL server. Cost-sensitive learning was implemented using two approaches: one using raw data with WBCE, respecting the original dataset distribution, and the other using undersampling to balance the dataset with equal numbers of attacks and BCE. Each local dataset *i* was assigned to the corresponding virtual *Device$_i$*.
- **Step 4.** Construct and evaluate the performance of the 16 FL configurations (16 = 2 optimizers for the FL server × 2 DL architectures × 2 cost-sensitive configurations × 2 feature thresholds) in terms of accuracy, recall, precision, and AUC over 100 rounds. Additionally, the SK test and BC system were used to rank the FL configurations for each cost-sensitive learning setup, feature threshold, and FL optimizer.
- **Step 5.** Compare the performances of Adam and SGD optimizers for each cost-sensitive learning setup and feature threshold using the NF-ToN-IoT-v2 dataset. Ultimately, identify the optimal FL configuration for cyber-detection within the NetFlow IoT dataset framework.

3.5 Abbreviation

To enhance readability and simplify model names, this study adopts the following specific naming conventions:

DLArchitecture_Optimizer_CostSensitiveSetup_FeatureThreshold

The abbreviations for DL architectures are DN for DNN and CN for CNN. The FL optimizers are abbreviated as S for SGD and A for Adam. The cost-sensitive setups are abbreviated as U for undersampling and W for weighted classes. For instance, DNSW20 represents a configuration utilizing the DNN architecture with the SGD optimizer, WBCE with weighted classes, and 20% of the features.

4 RESULTS AND DISCUSSION

This section analyzes the outcomes of using the FL technique with the DNN and CNN architectures. The evaluation includes two optimizers (Adam and SGD), two feature thresholds (20% and 60%), and two cost-sensitive setups (undersampling and WBCE) over 100 rounds on the NF-ToN-IoT-v2 dataset for binary classification. The results of the empirical study are discussed in relation to the RQs introduced in section 1.

4.1 Optimal Choice Between SGD and Adam Optimizers in FL for Attack Detection (RQ1)

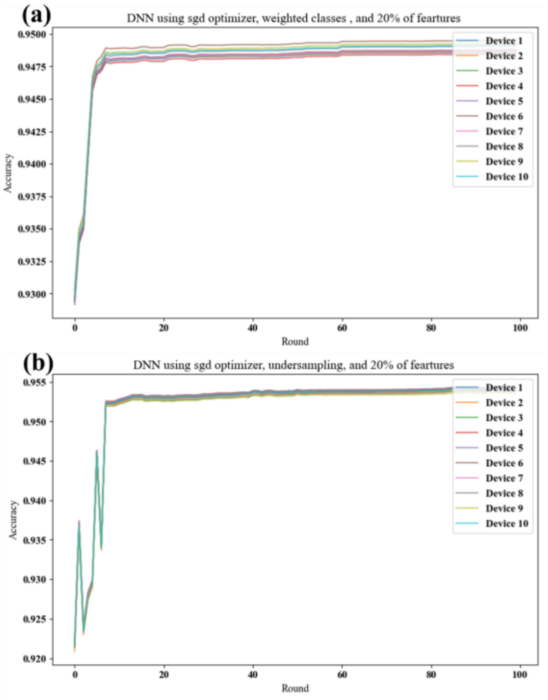


Figure 2: Accuracy progression across rounds for 10 devices using SGD with 20% of features: (a) DNN with weighted classes and (b) DNN with undersampling.

This subsection investigates the impact of SGD and Adam optimizers on the performance of FL configurations, with an emphasis on identifying the optimizer that enhances the accuracy of an FL-based IDS in IoT contexts. We analyze the average model accuracy across different feature thresholds for each cost-sensitive learning setup using DNN and CNN architectures, utilizing the SGD and Adam optimizers over 100 rounds to determine the optimal FL optimizer. For instance, when evaluating the accuracy of cost-sensitive setups (undersampling and weighted classes) with SGD and Adam using DNN and 20% of features across 10 devices: (1) Figures 2.a and 2.b display the accuracy values of DNN using the SGD optimizer with weighted classes and undersampling, respectively, allowing us to assess the models' accuracy for each device over 100 rounds; and (2) we calculate the average accuracy values for the 10 devices for different FL configurations in each round, as illustrated in Figure 3. For example, as shown in Figure 3. a, the average accuracy of the FL architecture across 10 devices, deploying a DNN with weighted classes and utilizing 20% of features with SGD as the FL optimizer, is referred to as DNSW20, whereas the average accuracy of the FL architecture across 10 devices, deploying a DNN with undersampling and using 20% of features with SGD as the FL optimizer, is indicated as DNSU20.

Figure 3.a shows the average accuracy values obtained using the DNN with 20% of the features. We observe the following:

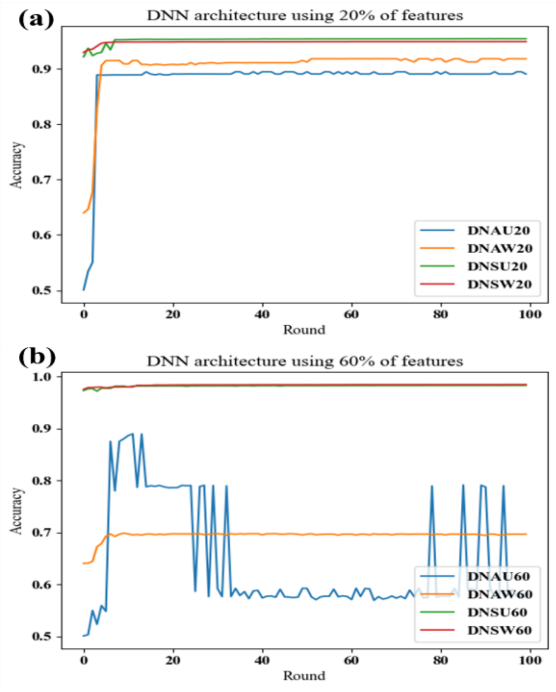


Figure 3: Average accuracy of FL based on DNN architecture using: (a) 20% of features and (b) 60% of features.

- For DNSU20, variability is present in the first 5 rounds, followed by stabilization, achieving a high accuracy of approximately 95%.
- For DNSW20, variability is present in the first 3 rounds, followed by stabilization, resulting in a high accuracy of approximately 94%.
- For DNAW20, an increase is observed in the first 3 rounds, followed by stabilization with very slight variations, leading to a consistent accuracy of approximately 91%.
- For DNAU20, an increase is observed in the first 5 rounds, followed by stabilization with very slight variations, resulting in a consistent accuracy of approximately 89%.

Figure 3.b shows the average accuracy values obtained using the DNN with 60% of the features. We observe the following:

- For DNSW60, stability is maintained throughout all rounds, achieving a high accuracy of approximately 98%.
- For DNSU60, stability is maintained in all rounds except for a slight variation in the second round, leading to a high accuracy of approximately 98%.
- For DNAW60, an increase is observed in the first 6 rounds, followed by stabilization, resulting in a consistent accuracy of approximately 91%.
- For DNAU60, the highest variation is observed across all rounds, with an initial increase in the first 5 rounds, some stability between rounds 5 and 15, followed by a decrease at round 36, and high variation at round 80, resulting in a maximum accuracy of 87%.

When using DNN as the base learner, the SGD optimizer proved to be the most effective FL optimizer, securing the top two positions for both the 20% and 60% feature thresholds. Specifically, DNSU20 for 20% of features and DNSW60 for 60% of features achieved the highest accuracies of 95% and 98%, respectively. In contrast, the Adam optimizer ranked lowest, with DNAU20 for 20% of features and DNAU60 for 60% of features, recording the lowest accuracies of 89% and 87%, respectively.

Figure 4.a illustrates the average accuracy values obtained using CNN with 20% of features. We observe the following:

- For CNSU20, a slight increase is observed during the first 5 rounds, followed by stabilization over the next 10 rounds. Variability occurs in the subsequent 3 rounds, after which stabilization occurs in the remaining rounds, achieving a high accuracy of approximately 95%.
- For CNSW20, a slight increase is observed during the first 5 rounds, followed by stabilization, ultimately leading to a high accuracy of approximately 94%.
- For CNAW20, a significant increase is observed during the first 3 rounds, followed by stabilization, resulting in a high accuracy of approximately 91%.
- For CNAU20, stabilization is observed throughout all rounds, resulting in a consistent accuracy of approximately 50%.

Figure 4.b shows the average accuracy values obtained using the CNN with 60% of the features. We observe the following:

- For CNSW60, a slight increase is observed during the first 5 rounds, followed by stabilization, ultimately achieving a high accuracy of approximately 98%.
- For CNSU60, a significant increase is observed during the first 5 rounds, followed by stabilization, ultimately resulting in a high accuracy of approximately 98%.
- For CNAW60, a significant increase is observed during the first 10 rounds, followed by stabilization over the next 10 rounds. Significant variability is present in the remaining rounds, with accuracies ranging between 70% and 92%.
- For CNAU60, stabilization is observed throughout all rounds, resulting in a consistent accuracy of approximately 50%.

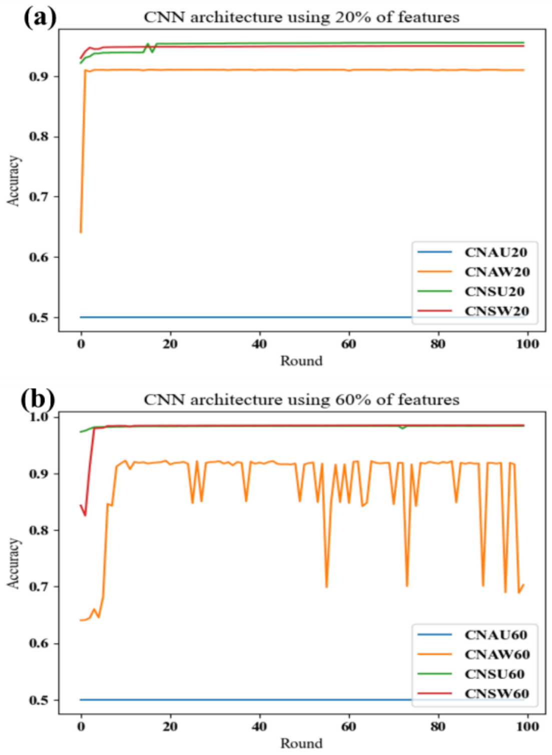


Figure 4: Average accuracy of FL based on CNN architecture using: (a) 20% of features and (b) 60% of features.

When using the CNN as the base learner, the SGD optimizer proved to be the most effective FL optimizer, securing the top two positions for both the 20% and 60% feature thresholds. Specifically, CNSU20 for 20% of features and CNSW60 for 60% of features achieved the highest accuracies of 95% and 98%, respectively. Conversely, the Adam optimizer ranked lowest, with CNAU20 stabilizing at an accuracy of 50% for 20% of the features and

CNAU60 showing variability with accuracy ranging between 70% and 92% for 60% of the features.

In summary, SGD outperformed Adam as an FL optimizer across DL architectures used as the base learners. Additionally, when utilizing SGD, employing weighted classes, particularly WBCE, as a cost-sensitive learning method yielded better performance with 60% of the features, while undersampling performed more effectively with 20% of the features across different DL architectures. On the other hand, when using Adam, the weighted classes (WBCE) consistently achieved better performance than the undersampling technique.

4.2 Optimal FL Configuration for Attack Detection Across Varied Settings (RQ2)

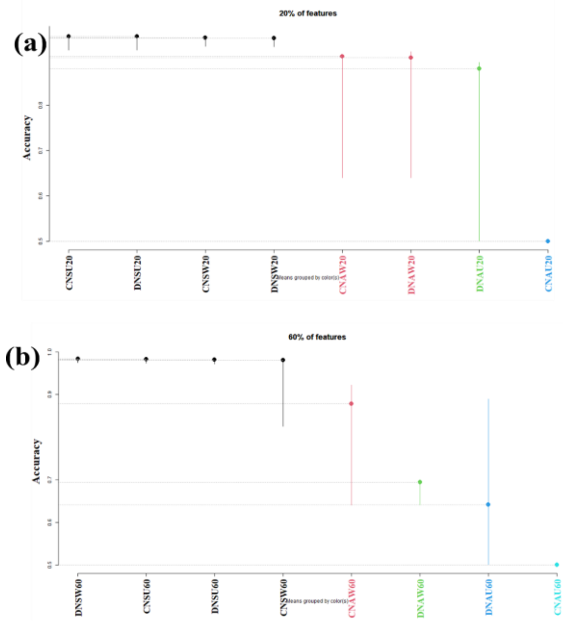


Figure 5: SK test results of FL configurations using (a) 20% of features and (b) 60% of features.

In this section, we compare various FL configurations, including FL optimizers, DL base learners, and cost-sensitive setups, for each feature threshold. The SK test was employed to focus on accuracy, grouping models, and identifying the most effective SK clusters, as illustrated in Figure 5. Additionally, the BC method was used to prioritize the models within the top SK clusters based on metrics such as accuracy, AUC, recall, and precision, as shown in Table 2. The SK test results are displayed in a graph, where the x-axis categorizes the FL classifier variants by cluster, arranging the best clusters from left to right, and the y-axis shows the accuracy scores. The central dots on each vertical line represent the mean accuracy, with the lines illustrating the outcomes of 100 rounds for each FL classifier. This analysis involved calculating the average accuracy for each round i across the 10 devices, denoted as $Average_i$, using Equation (1). For example, DNSW20 represents the average accuracy of the DNN architecture with the SGD optimizer, WBCE with weighted classes, and 20% of the features across 10 over 100 rounds $(Average_1, \dots, Average_{100})$.

$$Average_i = \sum_{j}^{\#Device} \frac{Accuracy_j}{\#Devices} \quad (1)$$

From Figure 5.a, we observe the following:

- For the 20% feature threshold, the SK distribution results in four distinct clusters. The first cluster comprises all models utilizing the SGD optimizer, including CNSU20, DNSU20, CNSW20, and DNSW20. The second cluster consists of CNAW20 and DNAW20. The third cluster includes only DNAU20, while the fourth cluster contains only CNAU20.
- For the 60% feature threshold, the SK distribution forms five distinct clusters. The first cluster includes all models using the SGD optimizer, specifically DNSW60, CNSU60, DNSU60, and CNSW60. The second, third, fourth, and fifth clusters contain CNAW60, DNAW60, DNAU60, and CNAU60, respectively.

Table 1: BC ranking of the FL variants belong to the best SK cluster.

TS	#F	Model	Accuracy	AUC	Recall	Precision	BC
20%	**7**	**CNSU20**	**95.19%**	**0.9874**	**92.41%**	**97.85%**	**9**
		CNSW20	94.89%	0.9872	97.33%	94.81%	8
		DNSU20	95.18%	0.9859	92.61%	97.62%	7
		DNSW20	94.78%	0.9859	97.26%	94.73%	6
60%	**20**	**DNSW60**	**98.34%**	**0.9984**	**99.03%**	**98.38%**	**11**
		DNSU60	98.17%	0.9984	98.52%	97.84%	7
		CNSU60	98.31%	0.9987	98.23%	98.39%	7
		CNSW60	98.07%	0.9943	98.48%	98.49%	5

TS: Feature threshold #F: Number of features

The findings indicate that the optimal FL configuration is achieved using the SGD optimizer and varies based on the feature threshold. Specifically, (1) with 20% of the features, the CNN base learner combined with the SGD optimizer outperforms the DNN, securing the top two positions according to the BC voting system. Conversely, (2) with 60% of the features, the DNN occupies the top two positions. Specifically, the best model for the 20% feature threshold is CNSU20, which achieved the highest BC score of 9 with an accuracy of 95.19%, an AUC of 0.9874, a recall of 92.41%, and a precision of 97.85% using 7 features. For the 60% feature threshold, DNSW60 achieved the best BC score of 11 with an accuracy of 98.34%, an AUC of 0.9984, a recall of 99.03%, and a precision of 98.38% using 20 features. Although CNSU20 and DNSW60 demonstrated comparable performance across various metrics, CNSU20 was selected as the preferred model due to its effectiveness with a minimal number of features (7 features).

5 CONCLUSION AND FURTHER WORKS

The research evaluated and compared 16 FL configurations for the binary classification of network intrusions, employing DNN and CNN as base learning models. The study explored the performance of two FL optimizers, SGD and Adam, in combination with two feature thresholds (20% and 60%) and two cost-sensitive learning approaches (Undersampling with BCE and weighted classes with WBCE) using the NF-ToN-IoT-v2 dataset. Evaluation metrics included accuracy, AUC, recall, and precision, further supported by the SK statistical test and the BC ranking system. The results demonstrated that SGD is a more reliable optimizer for attack detection in FL frameworks. The most effective model configuration was achieved using SGD as the FL optimizer, combined with CNN as the base learner and the Undersampling technique over the top 7 features.

The findings underscore the significance of employing FL in the development of decentralized IDSs specifically tailored for IoT networks to enhance attack detection. Future research should extend empirical evaluations to further validate or refine these results, potentially by utilizing a variety of datasets to assess the robustness and adaptability of FL-based IDS across diverse IoT environments. Additionally, investigating alternative models within FL frameworks could offer valuable insights into optimizing both performance and efficiency. Furthermore, deploying these models on embedded devices using TinyML and FL methodologies represents a promising direction for continued exploration.

REFERENCES

Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Alazab, M., Bhattacharya, S., Maddikunta, P. K. R. & Gadekallu, T. R. (2021). Federated Learning for Intrusion Detection System: Concepts, Challenges and Future Directions. Computer Communications, 195, 346–361. https://doi.org/10.1016/j.comcom.2022.09.012

Alsaedi, A., Moustafa, N., Tari, Z., Mahmood, A. & Adna N Anwar. (2020). TON-IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems. IEEE Access, 8, 165130–165150. https://doi.org/10.1109/ACCESS.2020.3022862

Amari, S. ichi. (1993). Backpropagation and stochastic gradient descent method. Neurocomputing, 5(4–5), 185–196. https://doi.org/10.1016/0925-2312(93)90006-O

Amiri, F., Rezaei Yousefi, M., Lucas, C., Shakery, A. & Yazdani, N. (2011). Mutual information-based feature selection for intrusion detection systems. Journal of Network and Computer Applications, 34(4), 1184–1199. https://doi.org/10.1016/J.JNCA.2011.01.002

Atharvan, G., Koolikkara Madom Krishnamoorthy, S., Dua, A. & Gupta, S. (2022). A way forward towards a technology-driven development of industry 4.0 using big data analytics in 5G-enabled IIoT. International Journal of Communication Systems, 35(1), e5014. https://doi.org/10.1002/DAC.5014

de Zarzà, I., de Curtò, J. & Calafate, C. T. (2023). Optimizing Neural Networks for Imbalanced Data. Electronics 2023, Vol. 12, Page 2674, 12(12), 2674. https://doi.org/10.3390/ELECTRONICS12122674

Dhaliwal, S. S., Nahid, A. Al & Abbas, R. (2018). Effective Intrusion Detection System Using XGBoost. Information 2018, Vol. 9, Page 149, 9(7), 149. https://doi.org/10.3390/INFO9070149

Dina, A. S., Siddique, A. B. & Manivannan, D. (2023). A deep learning approach for intrusion detection in Internet of Things using focal loss function. Internet of Things, 22, 100699. https://doi.org/10.1016/J.IOT.2023.100699

Friha, O., Ferrag, M. A., Shu, L., Maglaras, L., Choo, K. K. R. & Nafaa, M. (2022). FELIDS: Federated learning-based intrusion detection system for agricultural Internet of Things. Journal of Parallel and Distributed Computing, 165, 17–31. https://doi.org/10.1016/J.JPDC.2022.03.003

Hei, X., Yin, X., Wang, Y., Ren, J. & Zhu, L. (2020). A trusted feature aggregator federated learning for distributed malicious attack detection. Computers & Security, 99, 102033. https://doi.org/10.1016/J.COSE.2020.102033

Ho, Y. & Wookey, S. (2020). The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. IEEE Access, 8, 4806–4813. https://doi.org/10.1109/ACCESS.2019.2962617

Idrissi, M. J., Alami, H., El Mahdaouy, A., El Mekki, A., Oualil, S., Yartaoui, Z. & Berrada, I. (2023). Fed-ANIDS: Federated learning for anomaly-based network intrusion detection systems. Expert Systems with

Applications, 234, 121000. https://doi.org/10.1016/J.ESWA.2023.121000

Kerkhof, M., Wu, L., Perin, G. & Picek, S. (2022). Focus is Key to Success: A Focal Loss Function for Deep Learning-Based Side-Channel Analysis. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 13211 LNCS, 29–48. https://doi.org/10.1007/978-3-030-99766-3_2/COVER

Krawczyk, B., Woźniak, M. & Schaefer, G. (2014). Cost-sensitive decision tree ensembles for effective imbalanced classification. Applied Soft Computing, 14(PART C), 554–562. https://doi.org/10.1016/J.ASOC.2013.08.014

Kurniabudi, Stiawan, D., Darmawijoyo, Bin Idris, M. Y. Bin, Bamhdi, A. M. & Budiarto, R. (2020). CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection. IEEE Access, 8, 132911–132921. https://doi.org/10.1109/ACCESS.2020.3009843

Lomax, S. & Vadera, S. (2013). A survey of cost-sensitive decision tree induction algorithms. ACM Computing Surveys (CSUR), 45(2). https://doi.org/10.1145/2431211.2431215

Luengo, J., Fernández, A., García, S. & Herrera, F. (2011). Addressing data complexity for imbalanced data sets: Analysis of SMOTE-based oversampling and evolutionary undersampling. Soft Computing, 15(10), 1909–1936. https://doi.org/10.1007/S00500-010-0625-8/TABLES/16

McMahan, B., Moore, E., Ramage, D., Hampson, S. & Arcas, B. A. y. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data (pp. 1273–1282). PMLR. https://proceedings.mlr.press/v54/mcmahan17a.html

Mothukuri, V., Khare, P., Parizi, R. M., Pouriyeh, S., Dehghantanha, A. & Srivastava, G. (2022). Federated-Learning-Based Anomaly Detection for IoT Security Attacks. IEEE Internet of Things Journal, 9(4), 2545–2554. https://doi.org/10.1109/JIOT.2021.3077803

Murphy, J. F. A. (2018). The General Data Protection Regulation (GDPR). Irish Medical Journal, 111(5), 747. https://doi.org/10.1007/978-3-319-57959-7

Naidu, G., Zuva, T. & Sibanda, E. M. (2023). A Review of Evaluation Metrics in Machine Learning Algorithms. Lecture Notes in Networks and Systems, 724 LNNS, 15–25. https://doi.org/10.1007/978-3-031-35314-7_2/FIGURES/3

Nakashima, M., Kim, Y., Kim, J., Kim, J. & Sim, A. (2018). Automated Feature Selection for Anomaly Detection in. Network Traffic Data, 1(1), 27. https://doi.org/10.1145/1122445.1122456

Nguyen, T. D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N. & Sadeghi, A. R. (2019). DÏoT: A federated self-learning anomaly detection system for IoT. Proceedings - International Conference on Distributed Computing Systems, 2019-July, 756–767. https://doi.org/10.1109/ICDCS.2019.00080

Saari, D. G. (2001). Decisions and Elections. Decisions and Elections. https://doi.org/10.1017/CBO9780511606076

Sarhan, M., Layeghy, S., Moustafa, N. & Portmann, M. (2021). NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST, 371 LNICST, 117–135. https://doi.org/10.1007/978-3-030-72802-1_9/COVER

Scott, A. J. & Knott, M. (1974). A Cluster Analysis Method for Grouping Means in the Analysis of Variance. Biometrics, 30(3), 507. https://doi.org/10.2307/2529204

Shakeela, S., Sai Shankar, N., Mohan Reddy, P., Kavya Tulasi, T. & Mahesh Koneru, M. (2021). Optimal Ensemble Learning Based on Distinctive Feature Selection by Univariate ANOVA-F Statistics for IDS. https://doi.org/10.24425/ijet.2021.135975

Tao, X., Li, Q., Guo, W., Ren, C., Li, C., Liu, R. & Zou, J. (2019). Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. Information Sciences, 487, 31–56. https://doi.org/10.1016/J.INS.2019.02.062

Thakkar, A. & Lohiya, R. (2023). Attack Classification of Imbalanced Intrusion Data for IoT Network Using Ensemble-Learning-Based Deep Neural Network. IEEE Internet of Things Journal, 10(13), 11888–11895. https://doi.org/10.1109/JIOT.2023.3244810

Thu Huong, T., Phuong Bac, T., Minh Long, D., Doan Thang, B., Duc Luong, T., Thanh Binh, N. & Kim Phuc, T. (2020). LocKedge: Low-Complexity Cyberattack Detection in IoT Edge Computing. https://arxiv.org/abs/2011.14194v1

Zhang, Z. (2019). Improved Adam Optimizer for Deep Neural Networks. 2018 IEEE/ACM 26th International Symposium on Quality of Service, IWQoS 2018. https://doi.org/10.1109/IWQOS.2018.8624183

Zouhri, H., Idri, A. & Ratnani, A. (2023). Evaluating the impact of filter-based feature selection in intrusion detection systems. International Journal of Information Security, 1–27. https://doi.org/10.1007/S10207-023-00767-Y/TABLES/17

An Improved Meta-Knowledge Prompt Engineering Approach for Generating Research Questions in Scientific Literature

Meng Wang[1], Zhixiong Zhang[1,2], Hanyu Li[1,2] and Guangyin Zhang[2]
[1]*National Science Library, Chinese Academy of Science, Beijing, China*
[2]*University of Chinese Academy of Science, Beijing, China*
{wangmeng2022, zhangzx, lihy, zhangguangyin}@mail.las.ac.cn

Keywords: Research Question Generation, Prompt Engineering, Knowledge Extraction, LLMs, Knowledge-Rich Regions.

Abstract: Research questions are crucial for the development of science, which are an important driving force for scientific evolution and progress. This study analyses the key meta knowledge required for generating research questions in scientific literature, including research objective and research method. To extract meta-knowledge, we obtained feature words of meta-knowledge from knowledge-enriched regions and embedded them into the DeBERTa (Decoding-enhanced BERT with disentangled attention) for training. Compared to existing models, our proposed approach demonstrates superior performance across all metrics, achieving improvements in F1 score of +9% over BERT (88% vs. 97%), +3% over BERT-CNN (94% vs. 97%), and +2% over DeBERTa (95% vs. 97%) for identifying meta-knowledge. And, we construct the prompts integrate meta-knowledge to fine tune LLMs. Compared to the baseline model, the LLMs fine-tuned using meta-knowledge prompt engineering achieves an average 88.6% F1 score in the research question generation task, with improvements of 8.4%. Overall, our approach can be applied to the research question generation in different domains. Additionally, by updating or replacing the meta-knowledge, the model can also serve as a theoretical foundation and model basis for the generation of different types of sentences.

1 INTRODUCTION

Research questions play a crucial role in revealing the specific content of scientific and technological literature and grasping the research theme of an article., which serve as both the logical starting point and the guiding core of scientific research (Kuhn, 1962). Scientific literature, as an essential medium for recording scientific knowledge, is essentially a record and description of the process of proposing and solving research questions. Research question sentences are a crucial component of the knowledge content in scientific literature. By identifying research question sentences in scientific literature, we can explore the knowledge content contained within. It can be said that grasping the research question sentences of an article is an important prerequisite for understanding the content of a piece of scientific literature. Therefore, it will be of great significance to automatically identifying or generating research questions in scientific literature.

However, there are two limitations to current researches about identifying or generating research questions. Firstly, most current studies are mainly based on training on general datasets, ignoring the meta knowledge required for specific domains or tasks. Secondly, even if domain data is used for training LLMs, they have not filtered and refined the meta knowledge in scientific literature, and still mix a lot of redundant information. Therefore, we attempt to propose a research question generation method based on meta-knowledge prompt engineering. To extract key meta knowledge required for generating research questions from scientific literature, a sentence classification model based on feature word vectors is proposed. Then, research question generation prompts that integrate meta-knowledge will be used to fine-tune LLMs, which will provide more accurate and targeted input, thereby improving the quality and accuracy of the generated results. The architecture of the proposed method in this paper is shown in Figure 1.

The main contributions of this paper are as follows:

(1) To improving the quality and accuracy of the generated results, the prompts that integrate meta-

Wang, M., Zhang, Z., Li, H. and Zhang, G.
An Improved Meta-Knowledge Prompt Engineering Approach for Generating Research Questions in Scientific Literature.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 457-464
ISBN: 978-989-758-716-0; ISSN: 2184-3228

knowledge are constructed and used to fine tune LLMs.

(2) To extract key meta knowledge required for generating research questions from scientific literature, an improved DeBERTa model considering the feature word vectors is proposed.

(3) To improve the efficiency of meta-knowledge extraction, sections and paragraphs containing meta knowledge are located in scientific literature.

(4) The constructed prompt dataset that integrates meta knowledge is used to fine-tune LLMs.

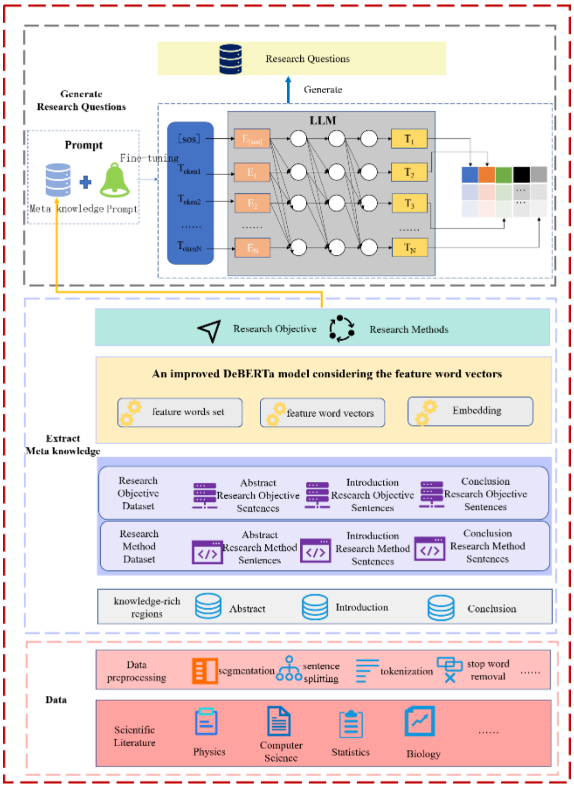


Figure 1: The architecture of the meta-knowledge prompt engineering approach for generating research questions in scientific literature.

The rest of this paper is organized as follows. The existing research of the meta-knowledge extraction and prompt engineering is presented in Section 2. Section 3 discusses an improved DeBERTa model, which considers the feature word vectors and knowledge-rich regions to extract key meta knowledge from scientific literature. The prompts that integrate meta-knowledge are constructed and used to fine tune LLMs in Section 4. Finally, Section 5 ends this study with conclusions and future work.

2 LITERATURE REVIEW

2.1 Meta-Knowledge Extraction

Meta-Knowledge extraction, also known as information extraction, refers to the task of automatically extracting structured information from unstructured or semi-structured text (Sarawagi, 2008). It aims to identify and extract relevant entities, relations, and events from text data, converting them into a structured format that can be easily processed and analyzed by downstream applications (Martinez-Rodriguez et al., 2018). Knowledge extraction plays a vital role in various natural language processing (NLP) applications, such as question answering, information retrieval, and knowledge graph construction (Chowdhary & Chowdhary, 2020).

In recent years, two mainstream approaches have emerged in the field of knowledge extraction: methods based on pre-trained models and methods based on LLMs. Methods based on pre-trained models utilize language models pre-trained on large-scale unlabeled text data, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and DeBERTa (He et al., 2020), and fine-tune them for specific knowledge extraction tasks. Chen et al. further explored the potential of DeBERTa for knowledge extraction by proposing a novel framework called DeBERTa-KE. This framework leverages the power of DeBERTa to jointly extract entities and relations from text, enabling end-to-end knowledge extraction (Chen et al., 2021).

With the growth of computational resources and the expansion of training data, large language models such as GPT (OpenAI, 2023), LLaMA (Touvron et al., 2023), and ChatGLM (Zeng et al., 2023) have demonstrated remarkable capabilities in the field of natural language processing. Researchers have begun to explore the use of these large language models for knowledge extraction tasks. The Meta AI team open-sourced the LLaMA model, which has 65 billion parameters. However, as the key information of sentences, feature words directly reflect the main content and deep meaning of the sentence and play an important role in improving the accuracy of research question sentence identification. Therefore, it is necessary to consider feature words in knowledge extraction (Touvron et al., 2023).

2.2 Prompt Engineering

Prompt engineering, also known as prompt design or prompt optimization, refers to the process of designing and optimizing prompts to effectively elicit desired behaviors or outputs from language models (Liu et al., 2023). It involves carefully crafting input prompts that guide the language model to generate high-quality, relevant, and coherent text. The quality of the generated text heavily depends on the effectiveness of the input prompts (Reynolds & McDonell, 2021). Well-designed prompts can significantly improve the coherence, relevance, and accuracy of the generated text, while poorly designed prompts can lead to nonsensical, irrelevant, or even harmful outputs.

We summarized the researches of meta-knowledge extraction and prompt engineer, and the results showed that there are two primary deficiencies in the current research: (1) many studies do not consider feature words in identifying research question sentences; (2) some researchers only use prompts to fine tune LLMs, which ignoring the meta knowledge required for specific domains or tasks. Therefore, this study analyzes the meta-knowledge required for generating research questions and manually summarizes the feature words of them. Moreover, the feature words are then embedded into the extraction model to improve the accuracy of the meta-knowledge extraction. The extracted meta-knowledge is integrated into prompt engineering to train LLMs, thereby enhancing the quality of the generated research questions.

3 EXTRACTING META KNOWLEDGE

3.1 Analysis of Meta-Knowledge Required for Research Question Generation

As the starting point and core of scientific research, research questions determine the direction, content, and objectives of a study. Generally, research questions can be divided into two main categories: theoretical questions and methodological questions (Alvesson & Sandberg, 2013). Theoretical questions focus on exploring the essence, laws, and mechanisms of things, aiming to establish or develop scientific theories. In scientific literature, these questions are usually reflected in the research objective section, where researchers explicitly state the theoretical issues they intend to explore.

Methodological questions arise from the challenges encountered in the technical methods during the research process, aiming to explore effective solutions. In scientific literature, methodological questions are usually reflected in the sentences about research method, where researchers focus on introducing the specific technical solutions and implementation steps adopted to solve the problems.

Thus, the key meta-knowledge required for generating research questions from scientific literature in this paper are the sentences of research objective and method, respectively.

3.2 Feature Word Vector Construction

3.2.1 Feature Word Sets

This paper employs manual annotation and iteration-based semi-automatic annotation methods to construct a dataset of research objective sentences and method sentences, obtaining a total of 20,000 high-quality corpus entries. From a linguistic perspective, feature words and characteristic sentence patterns in these two types of sentences are analyzed to construct a basic feature word set.

By combining the grammatical positions and contextual information of feature words, this paper obtains a total of 40 feature words. Some of the feature words and their contexts are shown in Table 1.

Table 1: Some feature words and the contexts.

feature words	contexts
analyze	…… were analyzed
	In order to analyze ……
	This paper analyzes ……
propose	…… was proposed in this study.
	In this paper …… is proposed
	This paper proposes ……
study	……was studied in this paper
	…… was studied

3.2.2 Feature Word Vector

Based on the analysis of part-of-speech tags and syntactic structure types of feature words, this paper calculates the frequency of feature words appearing in predicate positions and further expands the basic feature word set. A total of 40 feature words for knowledge elements are obtained, with a total frequency of 22,400 (notably, a sentence may contain multiple predicates). The proportion of each feature word represents its weight. Table 2 shows the frequency and weight distribution of some feature words.

Table 2: The frequency and weight distribution of some feature words.

feature words	frequency	weight
propose	5619	0.2508
explore	2520	0.1125
analyze	1955	0.0873
study	1702	0.0760
investigate	1549	0.0692
……	……	……
Total	22400	1

3.3 Embedding Feature Word Vector

This paper considers embedding the weight information of feature word vectors directly in the Classifier output stage within the DeBERTa model. The specific working mechanism of embedding feature word weight information is as follows:

Assume that for each input sentence, the DeBERTa model generates a hidden state vector $\mathcal{H} =< h_0, h_1, h_2, \dots, h_L >$, where the dimension is L. In the DeBERTa model, the dimension of the hidden state vector is generally 768. The weight vectors of feature words constructed in this paper is $\mathrm{F} =< f_0, f_1, f_2, \dots, f_{feature\_dim} >$, where f_n is the weight of the n^{th} feature vector. However, when the input sentence does not match any feature word, $f_n = 0$. $feature\_dim$ is the dimension of the feature vector. The hidden state vector of the RoBERTa model and the feature vector weight are concatenated, and this operation is implemented in the forward method of the Roberta Classifier, i.e.:

$$\mathcal{H}' = concat(h, f) \tag{1}$$

The dimension of the concatenated vector $\mathcal{H}'$ is L + feature_dim.

The linear layer of the classifier processes the concatenated vector $\mathcal{H}'$, and the formula is as follows:

$$\text{logits} = \mathrm{W} \cdot \mathcal{H}' + b \tag{2}$$

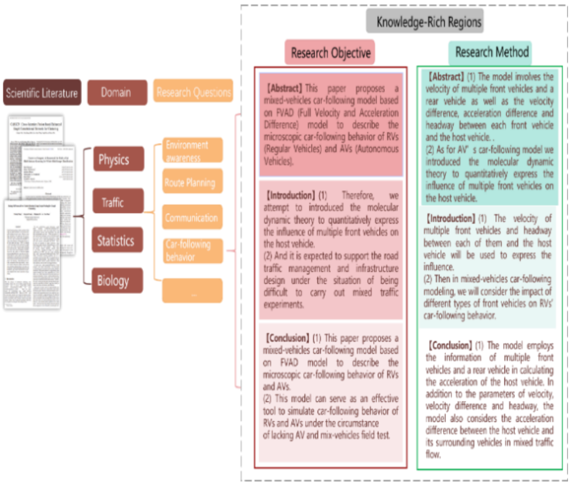
where W is the weight matrix with a dimension of L + feature_dim. b is the bias vector. logits is the raw score output by the classifier, which is finally passed to the softmax function to obtain the predicted probability distribution:

$$P = \text{softmax}(\mathrm{W} \cdot \mathcal{H}' + b) \tag{3}$$

3.4 Extracting Meta Knowledge

3.4.1 Locating Knowledge-Rich Regions

In scientific literature, knowledge is not evenly distributed but exhibits certain concentrations and regularities (Fortunato et al., 2018). Therefore, under the constraints of this writing logic, research objective sentences and research method sentences tend to be concentrated in the specific sections or paragraphs mentioned above. The knowledge-rich regions of research objective sentences and method sentences are shown in Figure 2.



【Abstract】 This paper proposes a mixed-vehicles car-following model based on FVAD (Full Velocity and Acceleration Difference) model to describe the microscopic car-following behavior of RVs (Regular Vehicles) and AVs (Autonomous Vehicles).

【Introduction】 (1) Therefore, we attempt to introduced the molecular dynamic theory to quantitatively express the influence of multiple front vehicles on the host vehicle.
(2) And it is expected to support the road traffic management and infrastructure design under the situation of being difficult to carry out mixed traffic experiments.

【Conclusion】 (1) This paper proposes a mixed-vehicles car-following model based on FVAD model to describe the microscopic car-following behavior of RVs and AVs.
(2) This model can serve as an effective tool to simulate car-following behavior of RVs and AVs under the circumstance of lacking AV and mix-vehicles field test.

【Abstract】 (1) The model involves the velocity of multiple front vehicles and a rear vehicle as well as the velocity difference, acceleration difference and headway between each front vehicle and the host vehicle. .
(2) As for AV' s car-following model we introduced the molecular dynamic theory to quantitatively express the influence of multiple front vehicles on the host vehicle.

【Introduction】 (1) The velocity of multiple front vehicles and headway between each of them and the host vehicle will be used to express the influence.
(2) Then in mixed-vehicles car-following modeling, we will consider the impact of different types of front vehicles on RVs' car-following behavior.

【Conclusion】 (1) The model employs the information of multiple front vehicles and a rear vehicle in calculating the acceleration of the host vehicle. In addition to the parameters of velocity, velocity difference and headway, the model also considers the acceleration difference between the host vehicle and its surrounding vehicles in mixed traffic flow.

Figure 2: The knowledge-rich regions of research objective and method.

3.4.2 Experiment

This paper selects the full text of scientific literature and extracts the abstract, introduction, and conclusion sections by locating fine-grained knowledge-rich regions. The training corpus is divided into training, validation, and test sets according to the ratio of 8:1:1, ensuring the consistency of positive and negative sample distributions across the datasets. The dataset format is shown in Table 3.

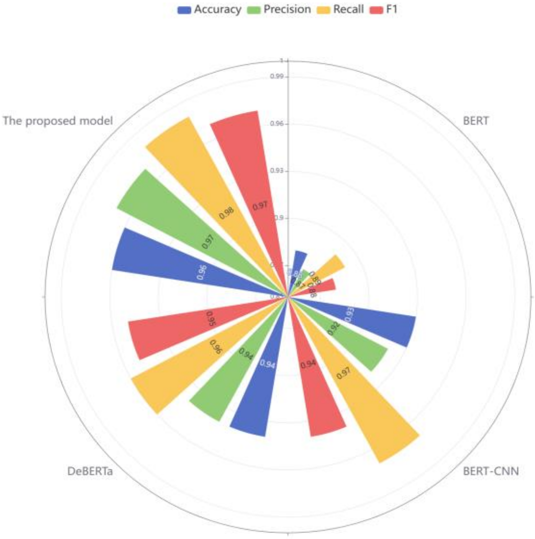


Figure 3: The extraction results of different models.

This paper selects BERT, BERT-CNN (Safaya et al., 2020), and DeBERTa as baseline models. According to Ref. (Li et al., 2023) and (Mei et al., 2023), the hyperparameter settings are shown in Table 4, and the extraction results are presented in Figure 3. The experimental results demonstrate that compared to the other three types of baseline models, the DeBERTa model based on feature word vectors proposed in this paper achieves the best meta-knowledge extraction performance, with an F1 score of 0.97.

Table 3: The dataset format.

Label	Sentence
0	Developing sharing economy of forestry has become an option to promote forestry development and solve the problems emerging from forestry economy.
1	In order to reveal the properties of polar metabolome in inflammatory cells, we selected LPS-induced RAW264.7 inflammatory cell models as the carrier for the research of metabolic fingerprint analysis.
2	As for AV's car-following model we introduced the molecular dynamic theory to quantitatively express the influence of multiple front vehicles on the host vehicle.

Table 4: The Hyperparameters of different models.

Hyperparameters	BERT	BERT-CNN	DeBERTa
lr	2e-5	2e-5	1e-5
b	16	16	64
e	30	30	10

4 GENERATING RESEARCH QUESTIONS BASED ON META-KNOWLEDGE PROMPT ENGINEERING

4.1 Constructing Meta-Knowledge Prompt Engineering

Considering the two key meta-knowledge elements of research questions: research objective sentences and research method sentences, we integrate the above-mentioned meta knowledge to manually construct the prompts, aiming to provide more accurate knowledge input for LLMs and improve the quality of research question generation. The format of the research question prompt is as follows: Given the title: "Title", the research objective: "Research Objective Sentence", the research methods: "Research Method Sentence", can we distill a concise question summarizing the research issue addressed in this article? Please use appropriate question words! Question: "Summarized Research Question". This prompt includes three knowledge elements: paper title, research objective sentence, and research method sentence, which are integrated into a complete research question generation task description, and finally provides a manually summarized research question.

This paper manually constructs 2,000 research question generation prompts, and some examples are as follows: Given the title: "Interpolating between Images with Diffusion Models", the research objective: "One little-explored frontier of image generation a........", the research methods: "We apply interpolation in the latent space \.......", can we distill a concise question summarizing the research issue addressed in this article? Please use appropriate question words! Question: How can we enable interpolation between two images using diffusion models, a capability missing from current image generation pipelines?

4.2 Fine-Tuning LLMs for Research Questions

To improve the quality of research question generation, this paper fine-tunes LLMs using the constructed prompt dataset that integrates meta knowledge. The fine-tuning dataset consists of three parts: task description, input, and output. The task description clearly states the objective of the generation task, what kind of task the model needs to complete, and what specific requirements and constraints exist, providing clear guidance for the subsequent input and output.

Based on the constructed fine-tuning dataset for research questions that integrates fine-grained knowledge, we adopt the LoRA (Low-Rank Adaptation) fine-tuning approach to fine-tune the large model (Su et al., 2021). The hyperparameter settings are as follows: batch_size: int = 10, micro_batch_size: int = 2, num_epochs: int = 2, learning_rate: float = 1e-5, lora_r: int = 8, lora_alpha: int = 16, lora_dropout: float = 0.05. The core idea of LoRA is to introduce a set of low-rank projection matrices at each layer of the large model and optimize these matrices to adapt the original model.

Specifically, for the i^{th} layer of the model, LoRA defines two projection matrices A_i and B_i with dimensions $(d,\ r)$and $(r,\ d)$, respectively, where d is the hidden layer dimension of the model, $r$is the projection dimension, and $r \ll d$.

During forward propagation, LoRA adds a correction term based on the projection matrices to the original layer computation result. Suppose the original forward computation of the i-th layer can be represented as:

$$h_i = f_i(x_i) \tag{4}$$

where x_i is the input of the i^{th} layer, and f_i is the forward computation function of the i^{th} layer (such as self-attention, feed-forward network, etc.). In LoRA, the modified forward computation formula is:

$$h_i' = f_i(x_i) + A_i B_i f_i(x_i) = f_i(x_i) + \Delta_i \tag{5}$$

where $\Delta_i = A_i B_i f_i(x_i)$ represents the correction term introduced by LoRA. This correction term can be seen as adding a low-rank perturbation to the original layer output $f_i(x_i)$.

The optimization objective of LoRA is to minimize the loss function of the modified model on the new task:

$$\mathcal{L}\left(\theta, \{A_i, B_i\}_{i=1}^{L}\right) = \sum_{(x, y)\in\mathcal{D}} \ell\left(f_{\theta, \{A_i, B_i\}}(x), y\right) \tag{6}$$

where θ represents the fixed parameters of the original model, $\{A_i, B_i\}_{i=1}^{L}$ represents all the projection matrices introduced by LoRA, $\mathcal{D}$ is the training dataset of the new task, and ℓ is the task-related loss function (such as cross-entropy loss). During the optimization process, we only update $\{A_i, B_i\}_{i=1}^{L}$, while keeping θ unchanged. Therefore, the training overhead of LoRA is much smaller than that of traditional full-parameter fine-tuning. At the same time, since the rank *r* of the projection matrices is much smaller than the dimension *d* of the original model, the additional parameters introduced by LoRA are also much smaller than the original model. The fine-tuning experimental results of different models are shown in Table 5.

4.3 Experimental Results and Analysis

To verify the effectiveness of the research question generation method that integrates meta-knowledge extraction, this paper selects Mistral-7B (Devillers et al., 2023), Baichuan2-7B (Wu et al., 2023), Chatglm3-13B (Zeng et al., 2022), Internlm-7B (Cai et al., 2024), and Llama3-8B (Touvron et al., 2023) as benchmark models. We compare the quality of the generated research questions with and without fine-tuning, and use the Sentence-Bert model (Reimers, 2019) to calculate the similarity between the generated research questions and standard answers under both conditions to evaluate the quality of the generated research questions. The comparison of generation results from different LLMs is shown in Table 6.

This paper sets the similarity threshold c = 0.8 as the accuracy threshold for generating research questions. Specifically, when c ≥ 0.8, the generated research question is correct; otherwise, the generated

Table 5: The results of the fine-tuning LLMs.

LLMs	BLEU	brevity_penalty	length_ratio	rouge1	rouge2	rougeL
Mistral-7B	0.167	0.939	0.941	0.506	0.255	0.449
Baichuan2-7B	0.121	0.980	0.980	0.459	0.215	0.405
Chatglm3-13B	0.133	0.982	0.982	0.463	0.213	0.411
Internlm-7B	0.161	0.948	0.950	0.493	0.243	0.429
Llama3-8B	0.154	0.932	0.934	0.491	0.239	0.430

Table 6: Comparison of generation results from different LLMs.

Instruction	Given the title， the research question and the research methods, can we distill a concise question summarizing the research issue addressed in this article? Please use appropriate question words! the title: 'DiffMatch: Diffusion Model for Dense Matching', the research question:'To address this, we propose DiffMatch, a novel conditional diffusion-based framework designed to explicitly model both the data and prior terms.', the research methods: 'Unlike previous approaches, this is accomplished by leveraging a conditional denoising diffusion model.'
Mistral	How can conditional diffusion models enable dense matching?
Baichuan2	How can diffusion models enable dense matching of images?
Chatglm3	How can conditional diffusion modeling enable dense matching?
Internlm	How effectively can conditional denoising diffusion models model both data and prior terms for dense matching?
Llama3	How accurately matches dense data a conditional denoising diffusion model?

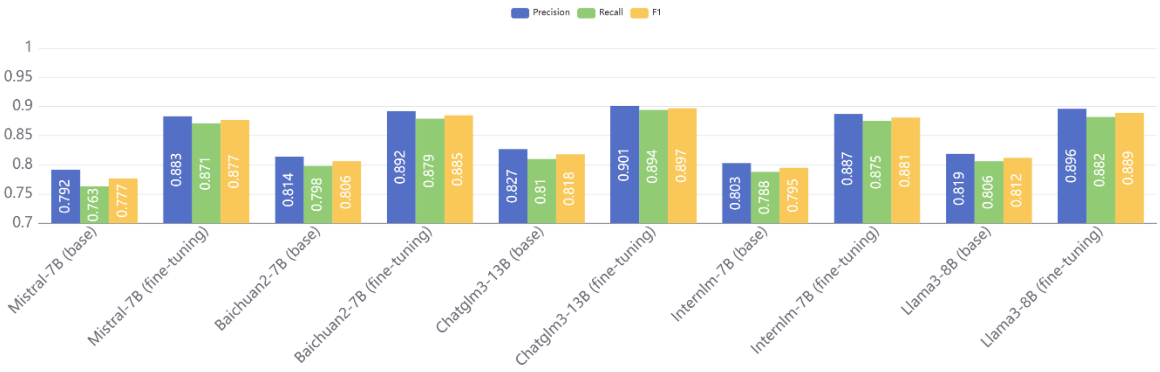


Figure 4: The experimental results of the benchmark LLMs and fine-tuned LLMs.

research question is incorrect. The experimental results of the benchmark models and fine-tuned models are shown in Figure 4. The experimental results demonstrate that the research question generation method based on meta-knowledge prompts provides more accurate and rich knowledge element inputs, reduces the difficulty of the recognition task, and improves the quality of research question generation.

5 CONCLUSIONS

This paper proposes a research question generation method based on meta-knowledge prompt engineering. To extract key meta knowledge required for generating research questions from scientific literature, a sentence classification model based on feature word vectors is proposed. Then, research question generation prompts that integrate meta-knowledge are used to fine-tune LLMs, which provide more accurate and targeted input, thereby improving the quality and accuracy of the generated results. The key contributions of this study are summarized as follows:

(1) In meta-knowledge extraction, we construct feature word sets for research objective sentences and research method sentences, and considers the feature word vector based on syntactic structure features. Utilizing the feature word vectors and the constructed. By concatenating the feature word vectors with the model's output, the model is trained, which helps model to capture and enhance the semantic expression and contextual information of feature words. Experimental results show that the DeBERTa model based on feature word vectors proposed in this paper achieves the best meta-knowledge extraction performance, with an F1 score of 0.97; compared to the original DeBERTa, the precision and recall are improved by 2.6% and 1.7%, respectively.

(2) Based on the key meta-knowledge: research objective sentences and research method sentences, research question prompts that integrate meta- knowledge are manually constructed, and LLMs are fine-tuned. Experimental results indicate that, the proposed method that integrates meta-knowledge extraction effectively improves the quality of generation, with an average F1 score of 88.6% after fine-tuning, an increase of 8.4%; from an individual model analysis, the fine-tuned Chatglm3-13B achieves the highest F1 score of 89.7%.

(3) This method can be applied to the generation task of research question sentences in different domains. In addition, by updating or replacing the meta-knowledge, it can generate different types of sentences, thereby providing a theoretical basis or model foundation for other downstream tasks.

Notably, this paper only optimizes the task of generating research question sentences for scientific literature. In future research, we plan to enhance the generation of other types of sentences. In addition, with the development of MultiModal LLMs, to improve the performance of text generation, combining multimodal data (such as images, tables, etc.) with prompt engineering is also one of the hot issues.

ACKNOWLEDGEMENTS

This study was funded by the National Key R&D Program of China(2022YFF0711900).

REFERENCES

Kuhn, T. S. (1962). *The Structure of Scientific Revolutions.* University of Chicago Press. https://doi.org/10.7208/chicago/9780226458106.001.0001

Sarawagi, S. (2008). Information extraction. Foundations and Trends® in Databases, 1(3), 261-377.

Martinez-Rodriguez, J. L., Lopez-Arevalo, I., & Rios-Alvarado, A. B. (2018). Openie-based approach for knowledge graph construction from text. *Expert Systems with Applications*, *113*, 339-355.

Chowdhary, K., & Chowdhary, K. R. (2020). Natural language processing. *Fundamentals of artificial intelligence*, 603-649.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 4171-4186.* https://doi.org/10.18653/v1/N19-1423

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692.*https://arxiv.org/abs/1907.11692

He, P., Liu, X., Gao, J., & Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Chen, Y., Zhang, Y., Hu, C., & Huang, Y. (2021). Jointly extracting explicit and implicit relational triples with reasoning pattern enhanced binary pointer network. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 5694-5703).

OpenAI. (2023). GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*.

Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., ... & Tang, J. (2023). GLM-130B: An Open Bilingual Pre-trained Model. *arXiv preprint arXiv:2210.02414*.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys, 55*(9), 1-35.

Reynolds, L., & McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended abstracts of the 2021 CHI conference on human factors in computing systems* (pp. 1-7).

Alvesson, M., & Sandberg, J. (2013). Constructing research questions: Doing interesting research.

Fortunato, S., Bergstrom, C. T., Börner, K., Evans, J. A., Helbing, D., Milojević, S., ... & Barabási, A. L. (2018). Science of science. *Science*, *359*(6379), eaao0185.

Safaya, A., Abdullatif, M., & Yuret, D. (2020). Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media. *arXiv preprint arXiv:2007.13184*.

Li, X., Zhang, Z., Liu, Y., Cheng, J., Tian, X., Wang, S., Su, X., Wang, R., & Zhang, T. (2023). A study on the method of identifying research question sentences in scientific articles. *Library and Information Service*, 67(09), 132-140. https://doi.org/10.13266/j.issn.0252-3116.2023.09.014

Mei, X., Wu, X., Huang, Z., Wang, Q., & Wang, J. (2023). A multi-scale semantic collaborative patent text classification model based on RoBERTa. *Computer Engineering & Science*, 45(05), 903-910.

Su, J., Lu, Y., Pan, S., Wen, B., & Liu, Y. (2021). RoFormer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864.*

Devillers, M., Saulnier, P., Scialom, T., Martinet, J., Matussière, S., Parcollet, T., ... & Staiano, J. (2023). Mistral: A Strong, Efficient, and Controllable Multi-task Language Model. *arXiv preprint* arXiv:2304.08582. https://arxiv.org/abs/2304.08582

Wu, J., Li, D., Li, S., Fu, T., Chen, K., Wang, C., ... & Zhang, Z. (2023). Baichuan 2: Open Large-scale Language Models. *arXiv preprint* arXiv:2304.09070. https://arxiv.org/abs/2304.09070

Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., ... & Tang, J. (2022). Glm-130b: An open bilingual pre-trained model.*arXiv preprint arXiv:2210.02414*.

Cai, Z., Cao, M., Chen, H., Chen, K., Chen, K., Chen, X., ... & Lin, D. (2024). Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Reimers, N. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv preprint arXiv:1908.10084*.

A Smart Hybrid Enhanced Recommendation and Personalization Algorithm Using Machine Learning

Aswin Kumar Nalluri and Yan Zhang[a]
School of Computer Science and Engineering, California State University San Bernardino, 5500 University Parkway, San Bernardino, CA, 92407, U.S.A.
008087502@coyote.csusb.edu, Yan.Zhang@csusb.edu

Keywords: Personalized Movie Recommendation, Hybrid Filtering, Content-Based Filtering, Term Frequency-Inverse Document Frequency, Collaborative Filtering, Alternating Least Squares.

Abstract: In today's era of streaming services, the effectiveness and precision of recommendation systems are pivotal in enhancing user satisfaction. Traditional recommendation systems often grapple with challenges such as data sparsity in user-item interactions, the need for parallel processing, and increased computational demands due to matrix densification, all of which hinder the overall efficiency and scalability of recommendation systems. To address these issues, we proposed the Smart Hybrid Enhanced Recommendation and Personalization Algorithm (SHERPA), a cutting-edge machine learning approach designed to revolutionize movie recommendations. SHERPA combines Term Frequency-Inverse Document Frequency (TF-IDF) for content-based filtering and Alternating Least Squares (ALS) with weighted regularization for collaborative filtering, offering a sophisticated method for delivering personalized suggestions. We evaluated the proposed SHERPA algorithm using a dataset of over 50 million ratings from 480,000 Netflix users, covering 17,000 movie titles. The performance of SHERPA was meticulously compared to traditional hybrid models, demonstrating a 70% improvement in prediction accuracy based on Root Mean Square Error (RMSE) metrics during the training, testing, and validation phases. These findings underscore SHERPA's ability to discern and cater to users' nuanced preferences, marking a significant advancement in personalized recommendation systems.

1 INTRODUCTION

In recent years, personalized recommendation systems have gained significant popularity due to the growing prevalence of online shopping platforms, social networks, and streaming services. Consider the last time you tried to choose a movie on a streaming site — it wasn't easy, was it? The challenge lies in the limitations of the engines behind those "Recommended for You" lists. These systems often rely on what you've already watched (collaborative filtering) (Ni et al., 2021) or suggest content based on genres you seem to prefer (content-based filtering) (Permana and Wibowo, 2023)(Philip et al., 2014). However, they frequently end up showing you more of the same, making it difficult to discover something new and exciting. This highlights the need for a smarter approach which truly understands your current mood by blending various advanced techniques from the world of machine learning, introducing you to content you'll genuinely enjoy.

In the competitive landscape of streaming platforms, the key to success hinges on engaging and delighting audiences. A crucial element in achieving this is providing movie recommendations that captivate viewers, almost like a touch of magic. Getting these recommendations right can increase user retention and encourage word-of-mouth promotion, which is vital in the ongoing streaming wars. It's not just about suggesting what an algorithm thinks you should watch; it's about understanding what viewers really want to see next, turning casual viewers into devoted fans eager to discover their next favorite movies.

This project introduces the Smart Hybrid Enhanced Recommendation and Personalization Algorithm (SHERPA) with the goal of revolutionizng movie recommendation processes. SHERPA combines collaborative filtering, content-based filtering, and advanced machine learning techniques to deliver tailored, accurate, and personalized content recommendations. Our goal is to simplify the movie discovery process by aligning recommendations with your preferences, not just based on what you've already

[a] https://orcid.org/0000-0002-5474-4019

Nalluri, A. and Zhang, Y.
A Smart Hybrid Enhanced Recommendation and Personalization Algorithm Using Machine Learning.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 465-472
ISBN: 978-989-758-716-0; ISSN: 2184-3228

seen. The focus is on creating a journey of content exploration that resonates with you, because, ultimately, every movie night should be about discovering something that truly hits the spot. SHERPA aims to eliminate the need for endless scrolling, ensuring that finding your next favorite movie is just a click away.

2 RELATED WORK

Traditional machine learning approaches in recommendation systems primarily focus on collaborative filtering and content-based filtering strategies (Ni et al., 2021). Collaborative filtering predicts user preferences by analyzing interactions and drawing insights from user behavior (Son and Kim, 2017). While this technique is widely used for its simplicity and effectiveness, it often faces challenges, particularly with new users (the cold start problem) and sparsity in user-item interactions (Wu et al., 2018)(Rahul et al., 2021).

Content-based filtering, on the other hand, suggests items based on their features and user preferences, emphasizing item metadata (Permana and Wibowo, 2023). However, this method may lead to a lack of diversity in recommendations, as it tends to suggest items similar to those the user has already interacted with (Philip et al., 2014).

Recent advancements in recommendation systems have made significant progress in overcoming these limitations. Techniques such as Singular Value Decomposition (SVD) have been employed to analyze user-item interactions and predict ratings by uncovering latent factors (Rahul et al., 2021). Additionally, new algorithms like Alternating Least Squares (ALS) with Weighted Regularization have enhanced collaborative filtering by prioritizing known interactions and incorporating regularization to prevent overfitting (SurvyanaWahyudi et al., 2017).

By combining these approaches, hybrid models that integrate elements of both content-based and collaborative filtering have been developed (Burke, 2002). These hybrid systems provide more comprehensive recommendations by considering both user behavior and content characteristics (Parthasarathy and Sathiya Devi, 2023). Zhou, et al. proposed an collaborative filtering algorithm Alternating-Least-Squares with Weighted-λ-regularization (ALS-WR), which is implemented on a parallel Matlab platform. They claimed that the performance of ALS-WR (in terms of root mean squared error (RMSE)) monotonically improves with both the number of features and the number of ALS iterations (Zhou et al., 2008). Chiny, et al. implemented a recommendation System based on TF-IDF and Cosine Similarity (Chiny et al., 2022). Hybrid systems not only improve the precision of recommendations but also offers a deeper understanding of user preferences and content relevance, paving the way for a new era in recommendation systems (Parthasarathy and Sathiya Devi, 2023).

3 DATASET AND PREPROCESSING

3.1 Dataset

The project involves two main datasets: the Movie Titles dataset and the Movie Ratings dataset, which are included in the Netflix Prize dataset posted on Kaggle (Netflix, 2006).

The Movie Titles dataset contains the information of 17,770 movies, with each movie represented as a tuple in the form: <Movie ID, Release Year, Movie Title, Director, Cast, Genre, Overview>. The original Movie Titles dataset contains Movie ID, Release Year, and Movie Title information of movies. We get extra information about these movies such as Director, Cast, Genre, Overview of Movie, from IMDB, an online database of information related to films, television series, etc.

This dataset provides a comprehensive overview of movies released from 1890 to 2005, with titles in English. The following is an examples of movie entries:

- Example: <1, 2003, Dinosaur Planet, Christian Slater, Scott Sampson, Animation, A four-episode animated series charting the adventures of four dinosaurs each on a different continent in the prehistoric world.>. This tuple shows that the movie ID is 1, the release year of this movie is 2003, the movie title is Dinosaur Planet, the director is Christian Slater, the cast is Scott Sampson, and the genre is Animation.

The Movie Ratings dataset comprises over 50 million ratings from 480,189 Netflix users, covering 17,770 movie titles, collected between October 1896 and December 2005. Each rating entry or instance contains User ID, Movie ID, Date of Rating, and Rating. Movie IDs are sequentially numbered from 1 to 17770. User IDs range from 1 to 2,649,429, with some numbers missing, representing a total of 480,189 users. Date of Ratings are consistently formatted as YYYY-MM-DD across all files. Ratings are on a five-star scale, ranging from 1 to 5 to show user opinion, where 5 represents the highest rating. To ensure customer privacy, unique customer IDs have been anonymized. The 50 million movie ratings dataset is splitted into three datasets. The training dataset contains total of $35,721,947$ ratings, the test dataset contains total of $7,654,704$ ratings, and the validation dataset contains total of $7,654,704$ ratings. The following is an examples of movie rating instances:

- Example: 1, 401047, 4, 2005-06-03
  This example shows that the user with ID 401047 rated the movie with ID 1 as 4 stars on June 3, 2005.

3.2 Data Preprocessing

During the data preprocessing stage, we structured unprocessed data to align with the machine learning model's format for effective learning. This involved parsing data from a file, extracting movie IDs, customer IDs, and ratings, and structuring them into a list. We converted this list into a pandas DataFrame for easier manipulation and handled format issues by skipping lines that didn't match the expected format. Additionally, we cleaned the data by replacing any NaN values with empty strings, preparing it for further analysis.

4 METHODOLOGIES

Recommendation systems use filtering algorithms to provide recommendations to users. These algorithms are classified or categorized majorly into collaborative-based filtering, content- based filtering, and hybrid algorithms. The proposed Smart Hybrid Enhanced Recommendation and Personaliza- tion Algorithm (SHERPA) integrates Term Frequency-Inverse Document Frequency (TF-IDF) for content-based filtering and Alternating Least Squares (ALS) with weighted regularization for collaborative filtering, offering a sophisticated method for delivering personalized suggestions.

4.1 Term Frequency-Inverse Document Frequency (TF-IDF)

Term Frequency-Inverse Document Frequency (TF-IDF) is a statistical measure used to evaluate how important a word in a document within a collection of texts known as a corpus (Rajaraman and Ullman, 2011). It is often used in text mining and information retrieval to weight and evaluate words differently based on their importance to a document relative to a collection. Words that are frequent in one document but less common across others receive a TF-IDF value suggesting they could be crucial, for comprehending the content of that document (Chiny et al., 2022).

Term Frequency (TF) is the number of times a term appears in a document relative to the total word count of that document. TF is calculated using Equation 1 as follows (Rajaraman and Ullman, 2011):

$$tf(t,\, d) \;=\; \frac{N_{t,d}}{N_d}, \tag{1}$$

where $N_{t,d}$ represents the number of times that term t occurs in document d, and N_d represents the total number of terms in the document d.

Inverse Document Frequency (IDF) measures the rarity of a term across all documents. IDF is calculated using Equation 2 as follows (Rajaraman and Ullman, 2011):

$$idf(t,\, D) \;=\; log\frac{N}{|d \,\in\, D : t \,\in\, d|}, \tag{2}$$

where N is the total number of documents in the collection in the corpus $N = |D|$; $|d \,\in\, D : t \,\in\, d|$ is the number of documents where the term t appears.

By combining Equation 1 and Equation 2, The TF-IDF score for term t in document d is calculated as follows:

$$tfidf(t,d,D) = tf(t,d) \times\ idf(t,D) \tag{3}$$

Words with high TF-IDF scores in a document are used more in that document and less in others, making them key indicators of what the document is about.

4.2 Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a matrix decomposition method that allows you to approximate a matrix as a product of 3 matrices (Kadhim et al., 2017). This process allows us to uncover connections in the data. For example, when we have information about how users rated items such as movies, but not every user rates every item, SVD comes in to complete the missing information (Widiyaningtyas et al., 2022). The SVD of an $m \times n$ complex matrix M is a factorization of the form

$$M = U \times \sum V^T, \tag{4}$$

where M is the original user item rating matrix, U is the matrix where each row represents a user in terms of latent factors, Σ is a diagonal matrix with singular values that indicate the importance of each latent factor, V^T is the transpose of a matrix where each column represents an item in terms of latent factors.

4.3 Alternating Least Squares (ALS)

Alternating Least Squares (ALS) is a technique that handles sparse data by optimizing matrix factorization process by breaking it down into two smaller or more manageable subproblems (Takács and Tikk, 2012). Unlike Singular Value Decomposition (SVD), which considers all entries in the user-item interaction matrix (including unknown or missing values), ALS focuses only on the known ratings and it scales well for large datasets and integrates regularization directly to prevent overfitting, making it ideal for collaborative filtering (Pilászy et al., 2010).

ALS with Weighted-λ-Regularization is an enhancement to the standard ALS approach. It introduces a regularization term to the optimization process, which helps to avoid overfitting a common problem where a model performs well on the training data but poorly on unseen data. The goal of ALS with Weighted-λ-Regularization is to find user and item feature matrices that predict how users would rate items, even new or previously unrated ones (Zhou et al., 2008).

The effectiveness of this method is measured by a loss function that captures two things (Zhou et al., 2008):

- How well the model predicts the known ratings.
- How complex the model is (the size of the user and item feature matrices).

The loss function is represented mathematically as:

$$f(U,M) = \sum_{(i,j)\in I} (r_{ij} - u_i^T m_j)^2 + \lambda \left(\sum_i n_{u_i} \|u_i\|^2 + \sum_j n_{m_j} \|m_j\|^2 \right), \tag{5}$$

where r_{ij} is the actual rating of item j by user i, u_i is the feature vector representing user i, m_j is the feature vector representing item j, I is the set of all (user, item) pairs for which the rating is known, λ is the regularization weight that controls the trade-off between fitting the training data well and keeping the model simple to avoid overfitting, n_{u_i} is the number of items rated by user i, which weighs the user's feature vector, n_{m_j} is the number of users who have rated item j, which weighs the item's feature vector.

Loss function with efficient weighted regularization controls the complexity of the model and prevents overfitting by penalizing large values of the user and item feature vectors.

ALS with Weighted-λ-Regularization is highly suitable for large-scale datasets because of its ability to efficiently handle sparse user-item matrices by focusing on observed interactions, reducing memory requirements, and allowing for parallel computation.

4.4 Content-Based Filtering

Content-Based Filtering is a method used by recommendation systems to suggest items to users based on the characteristics of the items themselves rather than on the user's interaction with other users (Van Meteren and Van Someren, 2000). This method uses item features (like overview, genre, director, cast in movies) to recommend items similar to what the user has liked and positively rated in the past (Philip et al., 2014).

Several algorithms are commonly used in content-based recommendation systems. TF-IDF is chosen over traditional techniques because it provides a more sophisticated way to evaluate the importance of words (or terms) in the content (Van Meteren and Van Someren, 2000). Unlike simple frequency counts, TF-IDF accounts for the rarity of terms across all documents, thus giving higher weight to terms that are unique to a particular item (Permana and Wibowo, 2023). This is crucial in differentiating items with similar but not identical content, as common terms do not overly influence the similarity score.

4.5 Collaborative Based Filtering

Collaborative filtering functions, as a recommendation system algorithm, forecasts a user's preferences by considering the preferences of users (Hameed et al., 2012). It operates on the premise that if users A and B share viewpoints on an item, it is probable that A will align with B's perspective on another item that A has not yet encountered (Wu et al., 2018) (Konstan and Riedl, 2012). By analyzing user item interactions like ratings or viewing history, the algorithm detects patterns and resemblances among users or items (Ni et al., 2021) (Goyani and Chaurasiya, 2020). This approach enables tailored recommendations by tapping into the preferences of the user community, making it widely adopted in suggesting movies, music, and various products. Figure 1 illustrates the mechanisms of collaborative and content-based filtering techniques. Collaborative filtering recommends items by identifying patterns among similar users, while content-based filtering suggests items based on their similarity to content previously liked by the user.

4.6 Hybrid Filtering

A Hybrid filtering algorithm enhances recommendation systems by merging collaborative and content-

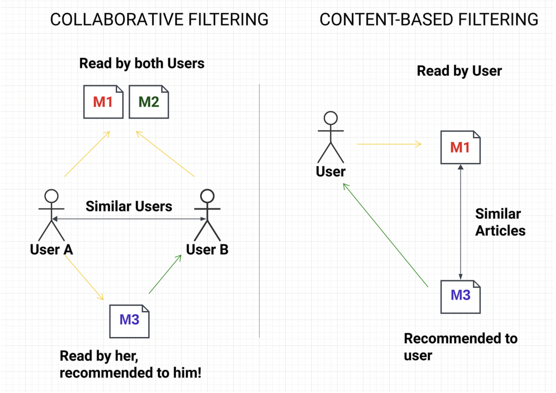


Figure 1: Comparison of collaborative and content-based filterings.

based filtering strategies leveraging the strengths of each to compensate for their shortcomings (Goyani and Chaurasiya, 2020)(Sharma et al., 2022). This strategy integrates the Singular Value Decomposition (SVD) technique, which forecasts user preferences based on patterns, in user item interactions with TF-IDF which examines item content to gauge its significance (Burke, 2002)(Thorat et al., 2015). By merging the personalized forecasts of SVD and the content specificity of TF-IDF, the hybrid model provides varied and thorough recommendations effectively tackling issues, like the cold start dilemma and enhancing recommendation accuracy (Parthasarathy and Sathiya Devi, 2023).

4.7 SHERPA

The proposed Smart Hybrid Enhanced Recommendation and Personalization Algorithm (SHERPA) is a recommendation system that intelligently combines the strengths of two methods: Alternating Least Squares (ALS) with Weighted Regularization for collaborative filtering, and Term Frequency-Inverse Document Frequency (TF-IDF) for content-based filtering, as shown in Figure 2.

By utilizing ALS with Weighted-λ-Regularization, SHERPA focuses on implicit data like known ratings and handles sparse data by optimizing matrix factorization process with loss function to avoid overfitting problem by computing independently user and item matrices across multiple processors or nodes in a cluster. At the same time, the incorporation of TF-IDF allows SHERPA works on explicit data by assigning weights ('Overview' - 45%, 'Genre' - 25%,'Director' - 15%, 'Cast' - 15%) to movie attributes based on their importance in a document. This weighting scheme helps identify the most distinctive and relevant terms for each document and transforms text-based movie attributes into numerical vectors. This vectorization allows the system to quantify and compare movie characteristics mathematically.

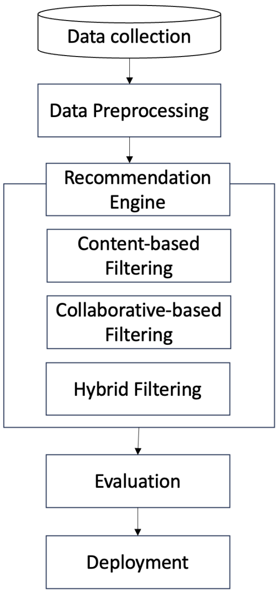


Figure 2: SHERPA Recommendation System Architecture.

This dual strategy working on both implicit and explicit data enables SHERPA to effectively handles large datasets, supports scalability and parallelization. it addresses the limitations of traditional methods to deliver more relevant recommendation and enhancing user satisfaction.

5 EXPERIMENT AND EVALUATION

To demonstrate the capabilities of the proposed SHERPA algorithm, we implemented a series of experiments. In the experimental setup, a dual-core processor and at least 2 GB of RAM are essential for general system operation. For the computationally intensive tasks of training and test, a GPU with a minimum of 2 GB of VRAM is necessary. Examples of suitable GPUs include the NVIDIA GTX 1050 or higher-end models.

5.1 Evaluation Metric

Root Mean Square Error (RMSE) is a standard way to measure the error of a model in predicting quantitative data (Hyndman and Koehler, 2006). It's particularly useful in recommender systems to evaluate the difference between predicted and actual ratings. RMSE provides a way to quantify the magnitude of prediction errors, taking the square root of the average squared differences between the prediction and the actual observation. The formula of RMSE is:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - a_i)^2}, \quad (6)$$

where p_i represents the predicted value for the ith instance, a_i is the actual value for the ith instance, N is the total number of instances.

A lower RMSE value indicates a better fit of the model to the data. It's especially effective in highlighting the impact of large errors, given that it squares the differences before averaging. However, it should be noted that RMSE can be sensitive to outliers and might not be well-suited if the error distribution is not uniform.

In the context of our paper, RMSE will serve as a key indicator of the accuracy of our recommendation system's predictions, allowing us to fine-tune the algorithm for optimal performance.

5.2 Evaluation Scenarios

We have designed two distinct scenarios to evaluate the performance of the SHERPA algorithm. One is designed for the existing users and the other is for new users. These scenarios are constructed to evaluate the system's responsiveness to each user's unique needs whether they're browsing casually or conducting specific searches based on their past interactions.

5.2.1 For Existing Users

For existing users, we designed two different scenarios to evaluate the proposed algorithm. One is to recommend movies to existing users who log in but do not conduct any search; the other is to recommend movies to existing users who log in and search a key word.

Existing User Log in and Without Search. When an existing user logs in without conducting any searching, the system uses their interactions to recommend movies. Since the user is simply browsing, collaborative filtering is used. This involves the algorithm analyzing the activities of users, with interests and suggesting movies that those users have enjoyed.

The following are the recommendation results from Hybrid and SHERPA approaches, the top 10 movies for existing user id = 401047 and without search keyword:

HYBRID Results:

1. Unknown Pleasures
2. The Swindle
3. Saint Sinner
4. Lone Wolf and Cub: Baby Cart in Peril
5. Die Hard 2: Die Harder
6. Seems Like Old Times
7. Kati Patang
8. Korn: Deuce
9. Hocus Pocus
10. The Usual Suspects

SHERPA Results:

1. Mel Gibson's Passion of the Christ
2. The Best of Friends: Vol. 4
3. Stargate SG 1: Season 7
4. The Winds of War
5. Stargate SG 1: Season 8
6. Friends: Season 6
7. Alias: Season 3
8. 24: Season 1
9. CSI: Season 3
10. Shania Twain: Up Close and Personal

Existing User Log in and Search with Keyword. When an existing user logs in and searches for a term like ”The Company”, the system transitions to the recommendation method. It combines the user's data with the search query to suggest options that cater not only to popular choices or similar users but also to results directly related to the search term.

The following are the recommendation results from Hybrid and SHERPA approaches, the top 10 movies for existing user id = 401047 and with search keyword ”The Company”:

HYBRID Results:

1. Center Stage
2. Ballet Favorites
3. Expo: Magic of the White City
4. A Raisin in the Sun
5. Robin and the 7 Hoods
6. Unknown Pleasures
7. Out of Sync
8. Orchestra Rehearsal
9. Category 6: Day of Destruction
10. The Usual Suspects

SHERPA Results:

1. Center Stage
2. Ballet Favorites
3. Expo: Magic of the White City
4. A Raisin in the Sun
5. Robin and the 7 Hoods
6. Swan Lake: Tchaikovsky (Matthew Bourne)
7. Out of Sync
8. Orchestra Rehearsal
9. Category 6: Day of Destruction
10. What Have I Done to Deserve This?

5.2.2 For New Users

New User Log in and Search with Keyword: When a new user looks up a term like ”The Company” without any viewing history, the algorithm uses content-based filtering. This approach analyzes factors such as genre, storyline, and actors of the movie to suggest movies with similar content to ”The Company”. The aim is to provide tailored recommendations based solely on the search query.

The following are the recommendation results from Hybrid and SHERPA approaches, the top 10 movies for new user and with search keyword ”The Company”:

HYBRID Results:

1. Center Stage
2. Ballet Favorites
3. Expo: Magic of the White City
4. A Raisin in the Sun
5. Robin and the 7 Hoods
6. Unknown Pleasures
7. Out of Sync
8. Orchestra Rehearsal
9. Category 6: Day of Destruction
10. The Usual Suspects

SHERPA Results:

1. Center Stage
2. Ballet Favorites
3. Expo: Magic of the White City
4. A Raisin in the Sun
5. Robin and the 7 Hoods
6. Swan Lake: Tchaikovsky (Matthew Bourne)
7. Out of Sync
8. Orchestra Rehearsal
9. Category 6: Day of Destruction
10. What Have I Done to Deserve This?

5.3 Results

In this Section, we compare the SHERPA algorithm’s performance against traditional hybrid systems using Root Mean Square Error (RMSE) metric across the training, test, and validation datasets as detailed below:

Table 1: The comparison of Hybrid and SHERPA algorithms.

Models	Training	Test	Validation
Hybrid	2.8289	2.9487	2.9492
SHERPA	0.8606	0.9039	0.9041
Improvement	69.6%	69.4%	69.3%

The comparison of the Hybrid and SHERPA algorithms across training, test, and validation datasets reveals significant differences in their performance. In the training dataset, the Hybrid model shows an RMSE of 2.8289, indicating some challenges in understanding user preferences, while SHERPA impressively reduces this to 0.8606, marking a substantial 69.6% improvement. Moving to the test dataset, Hybrid exhibits an RMSE of 2.9487, suggesting occasional inaccuracies, whereas SHERPA achieves a more reliable RMSE of 0.9039, a 69.4% enhancement. In the validation dataset, Hybrid scores 2.9492 in RMSE, highlighting room for improvement, whereas SHERPA excels with an RMSE of 0.9041, showcasing consistent and reliable performance.

SHERPA’s success is attributed to its advanced matrix factorization technique, weighted-λ-regularization, parallelization for scalability, computational efficiency, hybrid filtering approach, and continuous learning, which collectively result in a 70% improvement over traditional Hybrid algorithms. SHERPA’s balanced approach ensures both technical superiority and a more personalized recommendation experience for users.

6 CONCLUSION

This paper introduced Smart Hybrid Enhanced Recommendation and Personalization Algorithm (SHERPA), an advanced machine learning algorithm created to enhance and personalize the movie recommendation process. By combining content-based filtering using TF-IDF and collaborative filtering through ALS with Weighted Regularization, SHERPA has shown an improvement in recommendation accuracy and user satisfaction.

Through analysis using metrics like RMSE, SHERPAs performance compared to traditional hybrid models was highlighted. Notably SHERPA achieved a decrease in prediction errors with enhancements of around 70% across training, testing and validation datasets when compared to its predecessor. This emphasizes the algorithms improved capability to comprehend and forecast user preferences providing relevant content suggestions. Moreover, SHERPA’s innovative methodology tackles issues seen in existing recommendation systems such as overfitting and addressing the cold start problem. This ensures a scalable solution that caters to user interactions. Its proficiency in managing datasets and customizing content based on user behaviors as well as item traits sets a new standard in recommendation system technology.

In summary, the SHERPA algorithm signifies a progression in recommendation systems. The users content discovery experience is enhanced by SHERPA, which also paves the way for advancements in machine learning and artificial intelligence research and development. In the changing world personalized recommendation systems like SHERPA play a crucial role in driving future innovations.

ACKNOWLEDGMENTS

This project is based upon work supported by the U.S. National Science Foundation under Grant No. 2142503.

REFERENCES

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-adapted Interaction*, 12:331–370.

Chiny, M., Chihab, M., Bencharef, O., and Chihab, Y. (2022). Netflix recommendation system based on tf-idf and cosine similarity algorithms. In *Proceedings of the 2nd International Conference on Big Data, Modelling and Machine Learning*, pages 15–20.

Goyani, M. and Chaurasiya, N. (2020). A review of movie recommendation system: Limitations, survey and challenges. *Electronic Letters on Computer Vision and Image Analysis*, 19(3):0018–37.

Hameed, M. A., Al Jadaan, O., and Ramachandram, S. (2012). Collaborative filtering based recommendation system: A survey. *International Journal on Computer Science and Engineering*, 4(5):859.

Hyndman, R. J. and Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688.

Kadhim, A. I., Cheah, Y.-N., Hieder, I. A., and Ali, R. A. (2017). Improving tf-idf with singular value decomposition (svd) for feature extraction on twitter. In *3rd international engineering conference on developments in civil and computer engineering applications*.

Konstan, J. A. and Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User Modeling and User-adapted Interaction*, 22:101–123.

Netflix (2006). Netflix prize data on kaggle.com. Accessed: 2024-09-06.

Ni, J., Cai, Y., Tang, G., and Xie, Y. (2021). Collaborative filtering recommendation algorithm based on tf-idf and user characteristics. *Applied Sciences*, 11(20):9554.

Parthasarathy, G. and Sathiya Devi, S. (2023). Hybrid recommendation system based on collaborative and content-based filtering. *Cybernetics and Systems*, 54(4):432–453.

Permana, A. H. J. P. J. and Wibowo, A. T. (2023). Movie recommendation system based on synopsis using content-based filtering with tf-idf and cosine similarity. *International Journal on Information and Communication Technology*, 9(2):1–14.

Philip, S., Shola, P., and Ovye, A. (2014). Application of content-based approach in research paper recommendation system for a digital library. *International Journal of Advanced Computer Science and Applications*, 5(10).

Pilászy, I., Zibriczky, D., and Tikk, D. (2010). Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of the 4th ACM conference on Recommender systems*, pages 71–78.

Rahul, M., Kumar, V., and Yadav, V. (2021). Movie recommender system using single value decomposition and k-means clustering. In *IOP Conference Series Materials Science and Engineering*, volume 1022. IOP Publishing.

Rajaraman, A. and Ullman, J. D. (2011). *Mining of massive datasets*. Autoedicion.

Sharma, S., Rana, V., and Malhotra, M. (2022). Automatic recommendation system based on hybrid filtering algorithm. *Education and Information Technologies*, 27(2):1523–1538.

Son, J. and Kim, S. B. (2017). Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems with Applications*, 89:404–412.

SurvyanaWahyudi, I., Affandi, A., and Hariadi, M. (2017). Recommender engine using cosine similarity based on alternating least square-weight regularization. In *International Conference on Quality in Research (QiR): International Symposium on Electrical and Computer Engineering*, pages 256–261. IEEE.

Takács, G. and Tikk, D. (2012). Alternating least squares for personalized ranking. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 83–90.

Thorat, P. B., Goudar, R. M., and Barve, S. (2015). Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, 110(4):31–36.

Van Meteren, R. and Van Someren, M. (2000). Using content-based filtering for recommendation. In *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop*, volume 30, pages 47–56. Barcelona.

Widiyaningtyas, T., Ardiansyah, M. I., and Adji, T. B. (2022). Recommendation algorithm using svd and weight point rank (svd-wpr). *Big Data and Cognitive Computing*, 6(4):121.

Wu, C. S. M., Garg, D., and Bhandary, U. (2018). Movie recommendation system using collaborative filtering. In *International Conference on Software Engineering and Service Science (ICSESS)*, pages 11–15. IEEE.

Zhou, Y. H., Wilkinson, D., Schreiber, R., and Pan, R. (2008). Large-scale parallel collaborative filtering for the netflix prize. In *The 4th International Conference on Algorithmic Aspects in Information and Management*, pages 337–348. Springer.

An End-to-End Generative System for Smart Travel Assistant

Miraç Tuğcu[1], Begüm Çıtamak Erdinç[1], Tolga Çekiç[1], Seher Can Akay[1], Derya Uysal[1], Onur Deniz[1]
and Erkut Erdem[2]

[1]*Natural Language Processing Department, Yapı Kredi Teknoloji, Istanbul, Turkey*
[2]*Department of Computer Engineering, Hacettepe University, Ankara, Turkey*
{*mirac.tugcu, begum.citamakerdinc, tolga.cekic, seher.akay, derya.uysal, onur.deniz*}*@ykteknoloji.com.tr,*
erkut@cs.hacettepe.edu.tr

Keywords: Generative AI, Voice Assistant, Text-to-Speech, Speech-to-Text, Chatbot, Language Models, Deep Learning, Natural Language Processing.

Abstract: Planning a travel with a customer assistant is a multi-stage process that involves information collecting, and usage of search and reservation services. In this paper, we present an end-to-end system of a voice-enabled virtual assistant specifically designed for travel planning in Turkish. This system involves fine-tuned state-of-the-art models of Speech-to-text (STT) and Text-to-speech (TTS) models for increased success in the tourism domain for Turkish language as well as improvements to chatbot experience that can handle complex, multifaceted conversations that are required for planning a travel thoroughly. We detail the architecture of our voice-based chatbot, focusing on integrating STT and TTS engines with a Natural Language Understanding (NLU) module tailored for travel domain queries. Furthermore, we present a comparative evaluation of speech modules, considering factors such as parameter size and accuracy. Our findings demonstrate the feasibility of voice-based interfaces for streamlining travel planning and booking processes in Turkish language which lacks high-quality corpora of speech and text pairs.

1 INTRODUCTION

When planning their trips, users encounter a range of options and constraints. Traditionally, the planning process relies mostly on online search engines and user interactions via an interface which can be cumbersome. Voice assistants offer a more flexible and accessible way for users to express their needs. Improving human-computer interaction is possible by developing such an interface with a virtual assistant to offer a natural and intuitive way to provide information. A voice-enabled assistant has the potential to significantly improve this experience by allowing users to verbally convey their needs, and receive both textual and spoken confirmations about booking details.

The main objective of such an assistant system is understanding the requests of user and perform an action related to a travel topic. Therefore, intent classification and slot-filling, which are two crucial NLU components, are used to decide which travel related function to perform e.g. searching for tours, booking a hotel, cancelling reservations and so on. In (Dündar et al., 2020), a robust intent classifier for Turkish language is proposed with a similar objective for the banking domain. However, a slot-filling module is also needed to perform an action related to intention based on the preferences of a user. To meet this need, a named entity recognition (NER) model can be integrated into the chatbot. A recent work (Stepanov and Shtopko, 2024) demonstrates a specialized transformer model that outperforms ChatGPT and fine-tuned LLMs in zero-shot cross-domain NER benchmarks for various languages except Turkish. Users might specify their preferences in a more natural manner where contextual relation and domain knowledge are required. With this purpose, slot-filling can be even more successful in a few-shot setting with LLMs (Brown et al., 2020) instead of zero-shot.

To understand the user's intention, we utilized a BERT (Bidirectional Encoder Representations from Transformers) classifier (Devlin et al., 2019), which has been specifically fine-tuned for Turkish (Schweter, 2020). BERT is well-suited for understanding context and nuance in a language due to its deep bidirectional architecture. This allows the model to consider the full context of a word by looking at words that come before and after it. This is partic-

Tuğcu, M., Erdinç, B., Çekiç, T., Akay, S., Uysal, D., Deniz, O. and Erdem, E.
An End-to-End Generative System for Smart Travel Assistant.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 473-479
ISBN: 978-989-758-716-0; ISSN: 2184-3228

ularly beneficial for agglutinative languages such as Turkish.

There are notable challenges to developing a voice-enabled travel assistant in Turkish, due to the lack of natural voice generation and Automatic Speech Recognition (ASR) models which are also robust to noise and low-quality voice sources. This is mostly because of the limited availability of high-quality parallel data for training robust speech recognition and synthesis models in Turkish. There are multi-lingual TTS models successful in generating natural speech or speech recognition e.g. XTTS (Casanova et al., 2024) and Whisper (Radford et al., 2023), however, the models either have licences not available for commercial use or demand high computational resources. The latency of a response generated by a smart assistant directly affects the user experience. Therefore, a mono-lingual and single speaker but a robust, small architecture satisfies the high-throughput need such as MMS (Pratap et al., 2024) and FastSpeech2 (Ren et al., 2020). For automatic speech recognition task, there are successful models introduced in recent years such as Wav2Vec 2.0 (Baevski et al., 2020) and Whisper (Radford et al., 2022) with multi-lingual foundation models available. However, this foundation models has only rudimentary capabilities in some languages such as Turkish and they require further fine tuning to perform well enough for active usage.

In this study, a chatbot with NLU modules such as intent classification and slot filling in the travel domain for searching, booking and purchasing purposes of hotels and tours is developed. We approached the slot-filling problem with a hybrid approach by using few-shot prompting technique with an LLM where context matters the most for user messages. We further trained robust and lightweight STT and TTS models for Turkish language in the tourism domain to develop a voice interface for the chatbot which completes the virtual assistant experience.

2 SYSTEM ARCHITECTURE

Visual representation of our developed system is illustrated at Figure 1. The user is able communicate with the assistant through speech modules or written chat. Conversation flow manager is a multi module system that understands the intention of the user, leverages generative slot-filling and pattern matching to perform an action with given information through travel services. Through the function calling component located in the conversation flow manager, intent classification, slot filling, and pattern matching components elucidated in Section 2.1 enable the semantic interpretation of transcribed sentences. The generative slot-filling and pattern matching components address different needs by employing distinct methodologies. Generative slot filling leverages generative models to identify entities that cannot be easily expressed through predefined rules, whereas the pattern matching component uses regular expressions and fuzzy match scores based on predefined dictionaries to detect entities with fixed formats, such as hotel names, cities, districts, and dates. By leveraging the information extracted from these sentences within travel planning services, the system facilitates user interaction, ensuring the effective execution of functions such as system utilization and information retrieval from services. Moreover, with the function calling component, we enabled dynamic modification of endpoints and service variables directly from the interface we designed. This approach allowed for the seamless integration of new services with intents and facilitated the rapid adaptation to changing service requirements without the need for additional coding.

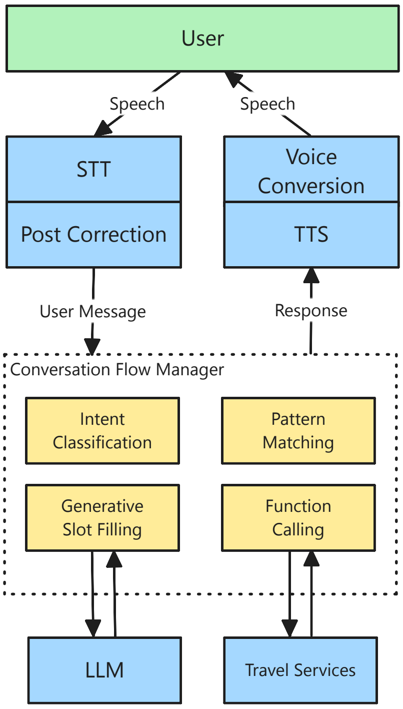


Figure 1: Overall virtual assistant architecture.

2.1 NLU Module

The Natural Language Understanding (NLU) component of our chatbot has two major components: intent classification and entity recognition. For intent classification, we utilized the BERT model based on the methodology outlined in (Dündar et al., 2020). We employed BERTurk model (Schweter, 2020), which is a model trained in Turkish corpora. This BERT based classifier met our demands and surpassed few shot training with LLMs, hence it is finetuned to be used as an intent classifier in travelling domain for this work as well.

On the other hand, we have observed that the entity collection for tourism can be challenging. The words we consider as entities can vary significantly in terms of subject matter and type. It may be necessary to perform entity extraction for a diverse range of entities, such as spa, sport, aquapark, nature, outdoor pool, child/baby-friendly, pet-friendly, honeymoon, and seafront. A user may wish to specify multiple features of the desired hotel within a single sentence to obtain results based on those criteria. Since it is more appropriate to consider these words as features rather than distinct entities, they were tagged as feature1, feature2 and so forth, before being transmitted to the relevant services. Additionally, the sentences constructed by users do not adhere to specific rhetorical patterns. Due to these problems, we decided that the use of large language models is more appropriate for this problem because of their capability of understanding complex patterns with fewer training examples.

To achieve this, we utilize ChatGPT from OpenAI to extract entities from sentences with our generative slot-filling component. By engineering a dynamic prompt, we were able to receive a JSON-formatted output that parsed the specified types and numbers of entities from the given sentences. Additionally, through the modification capabilities provided within the application, we enabled the addition or removal of new entities without the need for further development.

Moreover, working with large language models inherently posed the risk of receiving outputs in irregular formats. To mitigate this, we provided JSON examples within the prompts and implemented checks to ensure the outputs adhered to JSON rules. Additionally, since user prompts were directly fed into the large language model for entity extraction, this opened the possibility for the system's outputs to be manipulated. To address this, we refined the prompts to prevent users from altering the system prompt and obtaining distorted results.

Additionally, as mentioned in Section 2, we ensured that entities, which could be defined by rules, were identified for travel services by comparing them against regular expressions and words in our custom dictionaries, using calculated fuzzy match scores. The system we developed is depicted in Figure 2.

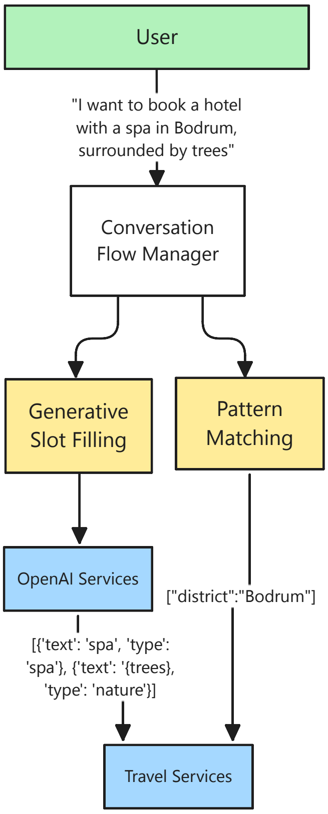

User
"I want to book a hotel with a spa in Bodrum, surrounded by trees"
Conversation Flow Manager
Generative Slot Filling
Pattern Matching
OpenAI Services
["district":"Bodrum"]
[{'text': 'spa', 'type': 'spa'}, {'text': '{trees}, 'type': 'nature'}]
Travel Services

Figure 2: Extracting entities from user prompts.

2.2 Text-to-Speech Module

The presented generation pipeline, as shown in Figure 3, contains a phoneme encoder, the LightSpeech TTS model, a vocoder and a voice conversion model. The text is converted to phoneme sequence by using an open-source Turkish grapheme-to-phoneme model and dictionary (McAuliffe et al., 2017). Speech synthesis with low latency is necessary for a seamless user experience which is why we use LightSpeech (Luo et al., 2021) and Parallel WaveGAN (PWG) (Yamamoto et al., 2020) vocoder that proved its efficiency. LightSpeech model is based on FastSpeech 2 but its architecture is designed more lightweight and more efficient via Neural Architecture Search. The audio quality is on par with FastSpeech2 while having a remarkable inference speed up. The generated mel-spectrograms are transformed into audio waveform by using a Parallel WaveGAN vocoder that is pre-trained on LibriTTS (Zen et al., 2019) which is capable of high-fidelity speech generation for Turkish. Finally, the OpenVoice (Qin et al., 2023) model is utilized for zero-shot cross-lingual Voice Cloning to specifically convert the speaker of the generated audio waveform. This pipeline allows alternative models to be used in

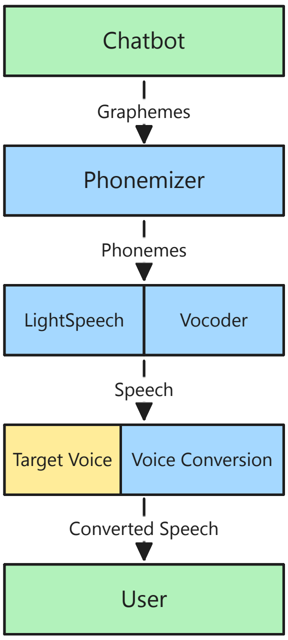


Figure 3: Text-to-speech pipeline for generation and voice cloning.

any part. For example, a vocoder model or a different voice cloning model can be easily implemented to replace respective parts.

2.3 Speech-to-Text Module

For the speech-to-text module Wav2Vec 2.0 speech recognition model is used and in order to correct potential errors in transcripts a post-correction method using an N-gram language model is used as shown in Figure 4. Wav2Vec 2.0 is a transformer based model that can be trained with raw audio data without any need for preprocessing (Baevski et al., 2020). Using raw audio data helps both with managing training data and with inference in the software pipeline as it does not introduce another layer that increases complexity. Using a multi-lingual foundation model with this architecture we have fine-tuned the model Turkish data and tourism related data. We have also implemented another layer for post-correction using N-gram based language model KenLM (Heafield, 2011). KenLM is a fast language modelling tool that can be used to create N-gram language models efficiently and also can be adapted to work with the Wav2Vec 2.0 model. Post-correction layer is used not only because it helps with correcting transcription errors that may arise due to similar sounding words and external noises; but also, recent studies have shown that using N-gram language model based post-correction can improve performance in low resource languages (Avram et al., 2023) and it can help with better adaptation on specific domains (Ma et al., 2023). We have created 5-gram language models to use in our experiments from general domain and tourism domain texts.

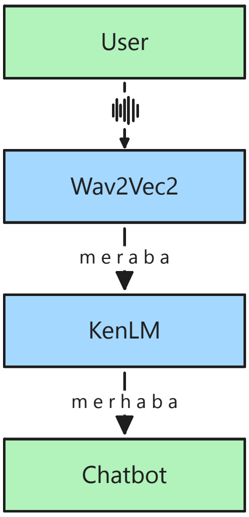


Figure 4: Speech-to-text pipeline that contains sequence modelling and post-correction models. As a side note, `merhaba` means `hello` in Turkish.

The other model we experimented on is W2v-BERT which combines the language model post-correction aspect into the trained model (Chung et al., 2021). This model uses a BERT encoder model as a language model instead of an N-gram language model. The advantage of using BERT is that it keeps a larger contextual information and also semantic knowledge of the words as well but compared to using an N-gram model it is a more resource demanding approach.

3 EXPERIMENTS & RESULTS

3.1 Text-to-Speech Experiments

Experimental Setup. We evaluate the LightSpeech model trained on a dataset that contains 5,131 audio samples with approximately 6 hours of novel reading without their text pairs. The average duration of the audio samples is 4.1 seconds. The transcriptions of audio samples are generated using our STT method. The errors in synthetic data consist mostly of similar-sounding words. Therefore, the effects of these errors are very little. Moreover, the errors in the synthetic transcriptions are expected to be minimal due to audio quality. The speech dataset and its phonemes are generated and aligned with the Montreal Forced Aligner public tool (McAuliffe et al., 2017) following (Ren et al., 2020). The audio waveforms are transformed into mel-spectrograms following (Luo et al., 2021), differently, we set the frame size and hop size to 300 and 1200 concerning the sampling rate of 24000. We train the model for 100k steps on a single NVIDIA V100 GPU. Models in our TTS pipeline other than the LightSpeech model are utilized as pre-trained models with their public weights.

Evaluation Methodology. There is no straightforward approach to evaluate speech generation. Most of the speech features like timbre or prosody may vary in the generated speech of a text compared to the ground truth utterance and it is an even harder challenge for multi-speaker datasets. Therefore, it is meaningful to evaluate a system by the aspect or the feature needed. We decided to evaluate intelligibility and pronunciation by transcribing the generated speech with ASR. We specifically choose a well-known and capable multi-lingual model Whisper (Radford et al., 2023), and 743 audio-text pairs from the Turkish subset of multi-lingual ASR benchmark known as FLEURS (Conneau et al., 2023) which is considered as an out-of-domain evaluation with respect to our training domain. The subset is approximately 2.6 hours long and the average duration of samples is 12.6 seconds. We generate speech of the texts from the dataset to create synthetic audio and original text pairs for each TTS model. ASR models transcribe TTS outputs into text hypotheses, allowing us to calculate the Word Error Rate (WER) and Character Error Rate (CER) by comparing them to the original transcript. For measuring the error rates, we apply the normalization of Whisper on references and hypotheses. In most cases, another preferable evaluation method is to evaluate naturalness and audio fidelity by Mean Opinion Score (MOS) metric but it's not an automatic evaluation strategy and the reliance on human raters presents a challenge. However, we decided to use a subset of 100 utterances generated for each model from the ASR benchmark we mentioned. We compare our results with public models successful in Turkish speech synthesis: 1) pre-trained Turkish MMS TTS model (Pratap et al., 2024) which is an end-to-end model with VITS (Kim et al., 2021) architecture, and 2) multi-lingual XTTS (Casanova et al., 2024) model with zero-shot voice-cloning feature that has a novel architecture based on Tortoise (Betker, 2023) and a HiFi-GAN vocoder (Kong et al., 2020) with 26M parameters. Parameter sizes of models are shown in Table 1.

Results. In our experiments, the LightSpeech model (our setup) is able to generate utterances that preserve the text content better than the MMS TTS model, as shown in Table 2. LightSpeech performs less well than XTTS which has equal or better accuracy on ASR evaluation than the original utterances in the FLEURS dataset. However, our setup is as accurate as XTTS on the CER metric evaluation with Whisper-large-v2. Also, there is a slight difference with XTTS on the CER metric evaluation with Whisper-medium. However, the MOS score of LightSpeech is less natural than MMS and far from XTTS on naturalness as shown in Table 3. This is mostly due to the size of the training data and its recording quality. Also, our observations show that our model is less natural on long input sequences of the FLEURS benchmark because it is trained on a dataset with short sequences and is not able to generalize long sequences in terms of naturalness. Note that MMS and XTTS models have nearly 12x and 150x more parameters than LightSpeech + PWG respectively, as shown in Table 1. Therefore, the results show that the model is robust in comprehensibility but needs improvement on naturalness, considering the constraints imposed by its size and limited training data.

Table 1: Text-to-speech models that is used in experiments and their parameter size.

Model	#Params
LightSpeech	1.8M
LightSpeech + PWG	3.1M
MMS	36.3M
XTTS	466.9M

Table 2: Evaluation results of Turkish speech synthesis by using Whisper models and FLEURS benchmark dataset. Original denotes the results of ASR models from Whisper paper (Radford et al., 2023).

Model	WER(↓)	CER(↓)
Whisper-medium		
LightSpeech	13.0	2.9
MMS	18.4	4.4
XTTS	10.1	2.5
Original	10.1	-
Whisper-large-v2		
LightSpeech	10.8	**2.5**
MMS	15.3	3.7
XTTS	8.3	**2.5**
Original	8.4	-

Table 3: MOS scores from a human study with regard to naturalness on a subset of Turkish FLEURS dataset.

Model	MOS(↑)
LightSpeech	2.98 ±0.081
MMS	3.34 ±0.082
XTTS	4.43 ±0.055

Table 4: Performance Comparison of Speech Recognition Models.

General Test Set		
Model	WER(%)	CER(%)
Wav2Vec 2.0	14.038	4.070
W2v-BERT	16.636	4.252
Wav2Vec 2.0 + kenLM	**8.106**	**1.669**
Tourism Test Set		
Model	WER(%)	CER(%)
W2v-BERT	13.112	2.229
Wav2Vec 2.0+kenLM	**8.888**	**1.974**

3.2 Speech-to-Text Experiments

Experimental Setup. The evaluation was done on a 6 hours dataset that is obtained from the public Turkish Common Voice dataset in the general domain and 1 hours dataset from tourism domain. For evaluation of Speech-to-text tasks, generally used metrics are word error rate (WER) and character error (CER). These metrics measure the error rates of transcriptions compared to the actual transcriptions of audio files. The lower error rates mean the model is more successful.

Results. Our experiments have demonstrated that using Wav2Vec 2.0 model together with kenLM post-correction outperforms using it without the language model and W2v-BERT model. The results are shown in shown in table 4. It is unsurprising for the N-gram language model post-correction to surpass the performance of the base model as the previous studies have shown similar results. The low scores from W2v-BERT model may be due to the multi-lingual foundation model's BERT model not being too successful in Turkish language.

4 CONCLUSION & FUTURE WORK

In this paper, we introduced the pipeline for a voice assistant in Turkish, that is capable of helping users in the tourism domain. This assistant leverages an intuitive voice interface by enabling users to seamlessly request information, access travel services, and complete their entire travel planning experience through spoken interactions. For slot-filling task of the assistant, a hybrid approach that combines regular expressions with few-shot LLM prompting is utilized. Additionally, lightweight and robust models for our NLU and speech modules are implemented to ensure a conversation at a natural pace. Our findings have demonstrated that the speech-to-text and text-to-speech models we trained achieved high intelligibility in spite of the scarcity of Turkish speech resources.

For future work, to improve the performance of text-to-speech models we intend to increase the quality and the quantity of our training data by speech enhancement and denoising techniques. We also aim to implement a zero-shot prosody cloning feature to the TTS pipeline to control the emotion emphasized in synthesized speech. For speech recognition, an additional post-correction model will be used to correct transcriptions of foreign words that can often be encountered in the tourism domain. For the NLU component, which constitutes the chatbot's understanding functions, we aim to leverage generative methods further to provide the user with more diverse and varied responses.

ACKNOWLEDGEMENTS

This work was done as a part of Smart Travel Digital Ecosystem project which was supported by TUBITAK (no: 9220043) and Celtic-NEXT (no: CE-2020-2-3) for the international industrial R&D projects program.

REFERENCES

Avram, A.-M., Smădu, R.-A., Păiş, V., Cercel, D.-C., Ion, R., and Tufiş, D. (2023). Towards improving the performance of pre-trained speech models for low-resource languages through lateral inhibition.

Baevski, A., Zhou, H., Mohamed, A., and Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations.

Betker, J. (2023). Better speech synthesis through scaling. *arXiv preprint arXiv:2305.07243*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. *CoRR*, abs/2005.14165.

Casanova, E., Davis, K., Gölge, E., Göknar, G., Gulea, I., Hart, L., Aljafari, A., Meyer, J., Morais, R., Olayemi, S., et al. (2024). Xtts: a massively multilingual zero-shot text-to-speech model. *arXiv preprint arXiv:2406.04904*.

Chung, Y.-A., Zhang, Y., Han, W., Chiu, C.-C., Qin, J., Pang, R., and Wu, Y. (2021). W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training.

Conneau, A., Ma, M., Khanuja, S., Zhang, Y., Axelrod, V., Dalmia, S., Riesa, J., Rivera, C., and Bapna, A. (2023). Fleurs: Few-shot learning evaluation of universal representations of speech. In *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805. IEEE.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dündar, E. B., Kiliç, O. F., Cekiç, T., Manav, Y., and Deniz, O. (2020). Large scale intent detection in turkish short sentences with contextual word embeddings. In *KDIR*, pages 187–192.

Heafield, K. (2011). KenLM: Faster and smaller language model queries. In Callison-Burch, C., Koehn, P., Monz, C., and Zaidan, O. F., editors, *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland. Association for Computational Linguistics.

Kim, J., Kong, J., and Son, J. (2021). Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In *International Conference on Machine Learning*, pages 5530–5540. PMLR.

Kong, J., Kim, J., and Bae, J. (2020). Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in neural information processing systems*, 33:17022–17033.

Luo, R., Tan, X., Wang, R., Qin, T., Li, J., Zhao, S., Chen, E., and Liu, T.-Y. (2021). Lightspeech: Lightweight and fast text to speech with neural architecture search. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5699–5703. IEEE.

Ma, R., Wu, X., Qiu, J., Qin, Y., Xu, H., Wu, P., and Ma, Z. (2023). Internal language model estimation based adaptive language model fusion for domain adaptation.

McAuliffe, M., Socolof, M., Mihuc, S., Wagner, M., and Sonderegger, M. (2017). Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pages 498–502.

Pratap, V., Tjandra, A., Shi, B., Tomasello, P., Babu, A., Kundu, S., Elkahky, A., Ni, Z., Vyas, A., Fazel-Zarandi, M., et al. (2024). Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52.

Qin, Z., Zhao, W., Yu, X., and Sun, X. (2023). Openvoice: Versatile instant voice cloning. *arXiv preprint arXiv:2312.01479*.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2022). Robust speech recognition via large-scale weak supervision.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., and Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Ren, Y., Hu, C., Tan, X., Qin, T., Zhao, S., Zhao, Z., and Liu, T.-Y. (2020). Fastspeech 2: Fast and high-quality end-to-end text to speech. *arXiv preprint arXiv:2006.04558*.

Schweter, S. (2020). Berturk - bert models for turkish.

Stepanov, I. and Shtopko, M. (2024). Gliner multi-task: Generalist lightweight model for various information extraction tasks. *arXiv preprint arXiv:2406.12925*.

Yamamoto, R., Song, E., and Kim, J.-M. (2020). Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6199–6203. IEEE.

Zen, H., Dang, V., Clark, R., Zhang, Y., Weiss, R. J., Jia, Y., Chen, Z., and Wu, Y. (2019). Libritts: A corpus derived from librispeech for text-to-speech. *arXiv preprint arXiv:1904.02882*.

Comparative Performance Analysis of Active Learning Strategies for the Entity Recognition Task

Philipp Kohl[1][a], Yoka Krämer[1][b], Claudia Fohry[2] and Bodo Kraft[1]
[1] *FH Aachen, University of Applied Sciences, 52428 Jülich, Germany*
[2] *University of Kassel, 34121 Kassel, Germany*
{p.kohl, y.kraemer, kraft}@fh-aachen.de, fohry@uni-kassel.de

Keywords: Active Learning, Selective Sampling, Named Entity Recognition, Span Labeling, Annotation Effort.

Abstract: Supervised learning requires a lot of annotated data, which makes the annotation process time-consuming and expensive. Active Learning (AL) offers a promising solution by reducing the number of labeled data needed while maintaining model performance. This work focuses on the application of supervised learning and AL for (named) entity recognition, which is a subdiscipline of Natural Language Processing (NLP). Despite the potential of AL in this area, there is still a limited understanding of the performance of different approaches. We address this gap by conducting a comparative performance analysis with diverse, carefully selected corpora and AL strategies. Thereby, we establish a standardized evaluation setting to ensure reproducibility and consistency across experiments. With our analysis, we discover scenarios where AL provides performance improvements and others where its benefits are limited. In particular, we find that strategies including historical information from the learning process and maximizing entity information yield the most significant improvements. Our findings can guide researchers and practitioners in optimizing their annotation efforts.

1 INTRODUCTION

Supervised model training is a widely adopted approach that requires annotated data. This data is obtained through an annotation process, which often necessitates the expertise of domain specialists, particularly in fields such as biology, medicine, and law. The involvement of experts is costly (Finlayson and Erjavec, 2017). To alleviate the costs, researchers have introduced various methods to reduce the annotation effort (Sintayehu and Lehal, 2021; Lison et al., 2021; Feng et al., 2021; Wang et al., 2019; Yang, 2021). A popular method is Active Learning (AL). It is based on the principle that not all data points are equally valuable for the learning process and thus strives to select a particularly informative subset for annotation (Settles, 2009).

Despite the development of numerous AL strategies, their performance across different use cases is not well understood. We consider the case of *entity recognition (ER)* in NLP and conduct a comparative performance analysis (Jehangir et al., 2023). A representative subset of corpora and AL strategies is included, which we selected from a specialized scoping review (Kohl et al., 2024).

Our contributions are as follows:

- We establish a comprehensive framework for evaluating AL strategies for ER. This includes identifying a subset of datasets (*corpora*) that covers a wide range of domains (e.g., newspapers, medicine, etc.) and significant AL parameters, selecting a broad range of AL strategies for diverse evaluation, and designing a suitable model architecture that balances both performance and runtime for testing.
- We conduct an extensive analysis to determine the best-performing AL strategies for ER, identifying strategies that perform consistently well across different domains. We also evaluate the robustness and stability of these strategies, considering the impact of random processes in model training and the AL process.

The paper is structured as follows: Section 2 starts with an overview of the research field and related work. Then, in Section 3, we delve into the fundamental concepts of AL, ER, and the *Active Learning Evaluation (ALE) Framework*. Afterward, Section 4 explains how we selected the subset of corpora

[a] https://orcid.org/0000-0002-5972-8413
[b] https://orcid.org/0009-0006-7326-3268

Kohl, P., Krämer, Y., Fohry, C. and Kraft, B.
Comparative Performance Analysis of Active Learning Strategies for the Entity Recognition Task.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 480-488
ISBN: 978-989-758-716-0; ISSN: 2184-3228

and strategies tested in this study. We then present a description of the experimental setup in Section 5. While Section 6 presents the results and analyzes our experimental findings, Section 7 concludes the paper.

Our results, including code, figures, and extensive tables, can be found on GitHub[1].

2 RELATED WORK

Researchers introduced numerous AL strategies for areas such as computer vision or NLP (Settles, 2009; Ren et al., 2022; Schröder and Niekler, 2020; Zhang et al., 2022; Kohl et al., 2024). The strategies have been classified into taxonomies to provide a structured domain understanding. However, the existing surveys typically refrain from ranking the strategies based on their efficacy (Zhan et al., 2022). There is a general lack of comparative performance data. While any new strategy is backed by performance data, these typically refer to a limited and arbitrary subset of existing strategies. Direct comparisons are further complicated by variability in parameter selection and implementation details. The present paper helps to close this gap by providing a systematically designed comparative performance analysis for AL strategies in the ER domain. The limited knowledge of the relative performance of advanced AL methods may explain why current annotation tools such as Inception (Klie et al., 2018), Prodigy (Montani and Honnibal,), and Doccano (Nakayama et al., 2018) focus on basic AL strategies, potentially overlooking more sophisticated ones.

Several frameworks support the implementation and evaluation of AL strategies in other areas. *libact* (Yang et al., 2017) focuses on comparing AL strategies with scikit-learn models, while DeepAL (Huang, 2021; Zhan et al., 2022) is tailored for image vision tasks. We utilize the Active Learning Evaluation (ALE) framework (Kohl et al., 2023), which has a sophisticated, modular design, supports integration with various deep learning libraries and cloud computing environments, and has a strong focus on reproducible research.

Besides AL, there are other approaches that can reduce the annotation effort: semi-supervised learning (Sintayehu and Lehal, 2021) leverages a small labeled dataset to annotate unlabeled data, and weak supervision (Lison et al., 2021) uses heuristics or labeling functions to annotate data automatically. Data augmentation (Feng et al., 2021) generates new examples by replacing words or reformulating sentences, enhancing the training dataset without additional manual effort. Zero-shot (Wang et al., 2019) and few-shot learning (Yang, 2021; Brown et al., 2020) techniques transfer knowledge from one domain to another, reducing the need for extensive new datasets. Large language models (LLMs) are inherently few-shot learners (Brown et al., 2020), but they are not always applicable due to offline scenarios, hardware limitations, or the need for smaller models in specialized domains (Jayakumar et al., 2023).

3 FUNDAMENTALS

In this section, we introduce core concepts and a common taxonomy of AL, which will be used in Section 4. We also define and embed the ER task into the active learning domain. Finally, we provide some details on the ALE framework.

3.1 Active Learning

Active learning (AL) addresses the reduction of annotation effort and, therefore, is embedded into the *annotation process* (Settles, 2009). This process consists of three steps: (a) selecting unlabeled documents (*batch)*, (b) annotating these documents, and (c) training a classifier. These steps are repeated until performance metrics (e.g., F1 score) reach a desired value. AL modifies step (a) so that data points are selected with an AL strategy instead of randomly or sequentially. AL is based on the assumption that different data points have different information gains for the learning process. The AL strategies quantify these gains (Settles, 2009; Finlayson and Erjavec, 2017).

The AL strategies can be divided into three categories (Settles, 2009; Zhan et al., 2022; Kohl et al., 2024):

Exploitation depends on model feedback (e.g., confidence scores) to compute an informativeness score. For example, *least confidence* selects data points the model is most uncertain about.

Exploration is solely based on the corpora and uses similarities and dissimilarities between data points. For example, some strategies embed the data points into a high-dimensional vector space and utilize cluster methods to select a batch of data points from different clusters.

Hybrid strategies combine exploitation and exploration approaches, for instance, by merging their scores. Several hybrid approaches start with exploration to identify a subset of the data points, which is then analyzed using exploitation. This way the need

[1] https://github.com/philipp-kohl/comparative-performance-analysis-al-ner

for costly model feedback is reduced to the selected subset.

3.2 Entity Recognition

Entity Recognition (ER) is a subtask of *information extraction* (Jehangir et al., 2023). Given some unstructured text, ER finds arbitrary, predefined domain-specific *entities* (e.g., persons, diseases, time units, etc.). On the technical level (see Figure 1), a model tokenizes the text and classifies these tokens. Thus, the model feedback (e.g., confidence scores) is present for each token.

AL strategies select whole documents for annotation. Some AL strategies rely on model feedback, which requires to aggreate the token-wise information to a document-wise score. Figure 1 visualizes the aggregation process.

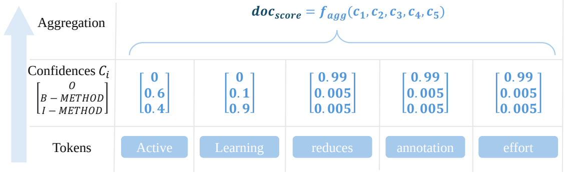

Aggregation
$doc_{score} = f_{agg}(c_1, c_2, c_3, c_4, c_5)$
Confidences C_i
$\begin{bmatrix} O \\ B-METHOD \\ I-METHOD \end{bmatrix}$
$\begin{bmatrix} 0 \\ 0.6 \\ 0.4 \end{bmatrix}$
$\begin{bmatrix} 0 \\ 0.1 \\ 0.9 \end{bmatrix}$
$\begin{bmatrix} 0.99 \\ 0.005 \\ 0.005 \end{bmatrix}$
$\begin{bmatrix} 0.99 \\ 0.005 \\ 0.005 \end{bmatrix}$
$\begin{bmatrix} 0.99 \\ 0.005 \\ 0.005 \end{bmatrix}$
Tokens
Active
Learning
reduces
annotation
effort

Figure 1: Tokenized text on the lowest level (whitespace tokenization for simplicity) on which the model infers predictions with the IOB2 (Ramshaw and Marcus, 1995) schema and computes confidence scores. At the top level, an aggregation function would compute a document-wise score based on the confidences per token.

3.3 Active Learning Evaluation Framework

We use the Active Learning Evaluation (ALE) framework (Kohl et al., 2023) for comparing different AL strategies against each other. ALE simulates the annotation process (see Subsection 3.1), which we call an *AL cycle*: (a) proposing new data points using an AL strategy. (b) annotating the data. Instead of forwarding the selected batches to human annotators, ALE uses provided gold labels of the corpora for the simulation. (c) Training and evaluation of the model.

Figure 2 gives an overview of ALE. The framework spans different stages. The first stage represents an experiment, which simulates a single strategy. The experiment follows a pipeline approach to preprocess the data and start so-called *seed runs*. Each seed run simulates one annotation process (*AL cycle*) with some random seed. Multiple seed runs are conducted to assess the stability and robustness of the AL strategies. Table 1 shows the connection between seed runs and AL cycles: A row represents the annotation process for a single seed with a growing corpus, while the column provides information on the robustness.

Table 1: Example F1 scores for seed runs across AL cycle iterations in a single experiment. Each cell shows the F1 score measured on the test corpus after each data proposal. For instance, AL cycle 2 represents the F1 scores after the second data proposal.

Seed Run	AL Cycle 1	AL Cycle 2	…	ALCycle N
Seed 1	0.01	0.05	…	0.85
Seed 2	0.01	0.06	…	0.83
…	…	…	…	…
Seed M	0.02	0.04	…	0.87

ALE has many configuration parameters. These and the corresponding experimental outcomes are reported to MLflow[2], which is an MLOps platform that supports reproducible research. The two core parameters are the *seeds* and the *step size*. The seeds-parameter is an integer list defining which and how many seed runs ALE starts. The step size defines how many documents the AL strategy selects in step (a) of the AL cycle.

ALE comes with an implementation for *spaCy*[3], which we have replaced by *PyTorch Lightning*[4] as deep learning library for step (c) of the AL cycle. PyTorch Lightning gives us finer control of the learning process.

The framework provides evaluation functions to address two critical aspects of AL: data bias and model calibration. It is crucial to avoid AL strategies that exacerbate existing biases within the dataset (see Section 6). Additionally, reliable model feedback requires well-calibrated models. To assess model calibration, ALE employs the *expected calibration error* (ECE) and *reliability diagrams* (Wang et al., 2021).

4 SELECTION OF CORPORA & STRATEGIES

We base our selection of corpora and strategies on the scoping review (Kohl et al., 2024), which reviewed 62 papers and collected information about the used AL strategies and other aspects of the evaluation environment.

4.1 Corpora

(Kohl et al., 2024) provide a collection of 26 publicly available corpora used to evaluate AL strategies for

[2] https://mlflow.org/
[3] https://spacy.io/
[4] https://lightning.ai/docs/pytorch/stable/

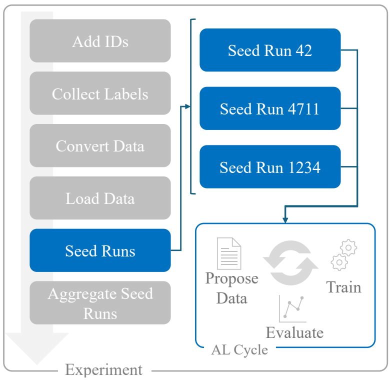


Figure 2: The ALE framework introduces three key concepts: *Experiments*, *Seed Runs*, and *AL cycles*. Each experiment involves a pipeline execution, with Seed Runs as the core element. A single Seed Run represents one AL cycle.

ER. We selected seven corpora based on the following criteria: frequency of use, diversity of domains (e.g., newspapers, medicine, social media), varying language complexity measured by the *moving average type-token ratio (MATTR)* (Covington and McFall, 2010; Kettunen, 2014)), label complexity and distribution (e.g., number of labels per sample), and average document length (limited to 512 tokens for compatibility with our model). The selected corpora are CoNLL2003, MedMentions, JNLPBA, GermEval, SCIERC, WNUT, and AURC-7. Further details are provided in Table 2.

4.2 Strategies

For strategy selection, we followed (Kohl et al., 2024), which highlights a focus on uncertainty exploitation strategies, particularly *entropy*, *margin*, and *least confidence*. These strategies use token-level confidences to compute scores, which are aggregated using methods such as average, minimum, maximum, sum, and standard deviation (Subsection 3.2). In addition to these three uncertainty strategies, we included count-based, round-robin, and two specialized strategies considering past predictions, as well as three exploration and two hybrid approaches.

Exploitation Strategies.

Least Confidence (LC) measures the uncertainty of the model for each token. The strategy strives to select documents the model is most uncertain about to receive a high information gain (Esuli et al., 2010; Şapci et al., 2023).

Margin Confidence computes the difference (margin) of the confidences for the two most probable labels per token. The intention is that a confident decision would have a high margin (e.g., $0.93 - 0.03 = 0.9$) because the decision boundary is learned well, while not confident decisions have very low margins (e.g., $0.45 - 0.4 = 0.05$). The strategy selects documents with low aggregated margins (Settles, 2009; Şapci et al., 2023).

Entropy Confidence uses the Shannon entropy to quantify the expected information gain. The strategy selects documents with a high entropy (Yao et al., 2020; Şapci et al., 2023).

Max Tag Count sums the number of entities the model predicts in a document (label different from the O-tag). The strategy favors documents with many entities because the authors hypothesize that the information gain is higher (Esuli et al., 2010).

Round Robin by Label strives to achieve a balanced distribution of labels in the batches. The strategy employs a round-robin approach to select documents based on their labels. The strategy maintains a score for each label per document. This differs from the previous strategies, which compute a single score per document (Esuli et al., 2010).

Fluctuation of Historical Sequence measures the uncertainty over the last *n* predictions (*historical*) instead of only considering the current prediction. The authors define a formula for a weighted sum of the current confidence and the historical confidence scores. The intuition is that volatile confidence scores indicate a higher impact on the learning process than stable ones because they might influence the decision boundary (Yao et al., 2020).

Tag Flip of Historical Sequence measures the instability of the model's decisions for a document. It counts the label changes (*tag flip*) for each token in a document across the last *n* predictions. Documents with many flips can be an indicator to influence the decision boundaries and, therefore, are beneficial for the training process (Zheng et al., 2018).

Exploration Strategies.

Diversity embeds the dataset into a vector space and precomputes pair-wise cosine similarities. The strategy selects data points that are most dissimilar to already labeled data points. In that way, the dataset should be diverse (Chen et al., 2015).

Maximum Representativeness-Diversity extends the previous strategy by adding the condition to not only select data points that are most dissimilar to already labeled data points (diversity) but also most similar to unlabeled documents (representative). The authors (Kholghi et al., 2015) use the product of the diversity and the representative score as document score.

K-Means Cluster Centroids embeds the data points into a vector space and clusters them with the k-means algorithm. The strategy selects data points nearest to cluster centroids (Van Nguyen et al., 2022).

Table 2: Overview of the seven selected corpora: Besides the domain as a selection criterion, the characteristics highlighted in bold also served as criteria. The row *number of labels* also states information about the label balance.

Corpus	**CoNLL03**	**MedMent.**	**JNLPBA**	**SCIERC**	**WNUT16**	**GermEval**	**AURC7**
Domain	**News**	**Medicine**	**Bio-medicine**	Scientific papers	Twitter posts	Encyclo-pedia	Politics
MATTR	**0.96**	**0.77**	0.9	0.79	0.95	0.96	0.89
Size (s=sample, t=token)	20744 s ⌀ **15 t**	4392 s ⌀ **275 t**	**22402 s** ⌀ 26 t	500 s ⌀ **131 t**	7244 s ⌀ 18 t	31300 s ⌀ 19 t	7977 s ⌀ 27 t
# of labels	4 (unbal.)	1 (bal.)	5 (unbal.)	6 (unbal.)	**10** (unbal.)	3 (unbal.)	2 **(bal.)**
Language	English	English	English	English	English	**German**	English
Data ratio without labels	0.205	**0**	0.113	0.002	0.537	0.411	0.436
# of labels per sample	1.691	**80**	2.674	16.188	0.771	1.206	0.634

Hybrid Strategies.

Representative LC sequentially applies an exploration and then an exploitation strategy. At first, the exploration strategy selects data points that represent the unlabeled documents best. The least confidence strategy selects data points from this subset the model is most uncertain about (Kholghi et al., 2017).

Information Density uses a combination of the representative and the entropy strategy. For each document, the strategy independently computes the cosine similarity with the unlabeled dataset and the entropy score. Afterward, the product of these scores represents the document (Settles and Craven, 2008).

5 EXPERIMENTAL SETUP

We conducted four experiment series, which are illustrated in Figure 3: The results of the first three *pre-series* led to our *standard series*, which we applied to all strategies. For all experiment series, we defined two test concepts:

Performance Tests: measure the F1 macro score at each iteration of the AL cycle. Following each data proposal, ALE retrains the model on the growing training corpus and evaluates the model on the corresponding complete and immutable test corpus. The scores are averaged across the seed runs (Table 1). Good-performing AL strategies show a steeper increase than the randomizer in model performance (see Figure 5).

Variance Tests: measure the variance and standard deviation of the F1 macro scores for each iteration of the AL cycle across the seed runs (Table 1). AL strategies with lower variance are preferable because they do not seem to be sensitive to random processes. We also call strategies fulfilling this characteristic *robust*.

The *Model Architecture* series explored various models from the RoBERTa family (Liu et al., 2019), taking into account the large number of experiments and their associated runtime. To ensure reliable confidence estimates, we tested label smoothing (Wang et al., 2021) and a CRF layer (Liu et al., 2022). Label smoothing yielded better model calibration. In the *Seed Settings* series, we assessed the number of seed runs required to obtain stable variance and performance estimates. Additionally, in *Aggregation Methods*, we evaluated different aggregations for uncertainty strategies, selecting only the most effective ones for use in the *Comprehensive Comparison*: We summarize the main parameters as follows: We use the *Distil RoBERTa Base* model(Liu et al., 2019; Sanh et al., 2020)[5] with label smoothing of 0.2. To realize a fair comparison between the different strategies, we set fixed hyperparameters for the model. Therefore, we always used 50 training epochs, a learning rate of $2e-5$, and a weight decay of 0.01 as recommended by (Liu et al., 2019; Kaddour et al., 2023). We used a batch size of 64. For ALE we use 3 seed runs for performance tests and 20 seed runs for variance tests. We chose the *step size* per corpus so that each data proposal delivers a similar amount of tokens.

At this stage, we use only the best-performing aggregation method for the uncertainty strategies found in the pre-series *Aggregation Methods*. This results in 12 strategies. For each strategy, we run 2 variance tests and 7 performance tests. For the randomizer baseline, we only conducted the performance tests. This results in 115 single experiments.

We used a workstation with 96 CPU cores and 3 Nvidia Quadro RTX 8000, each with 48GB of VRAM. The experiments took about 720 hours (30 days).

[5] https://huggingface.co/distilbert/distilroberta-base

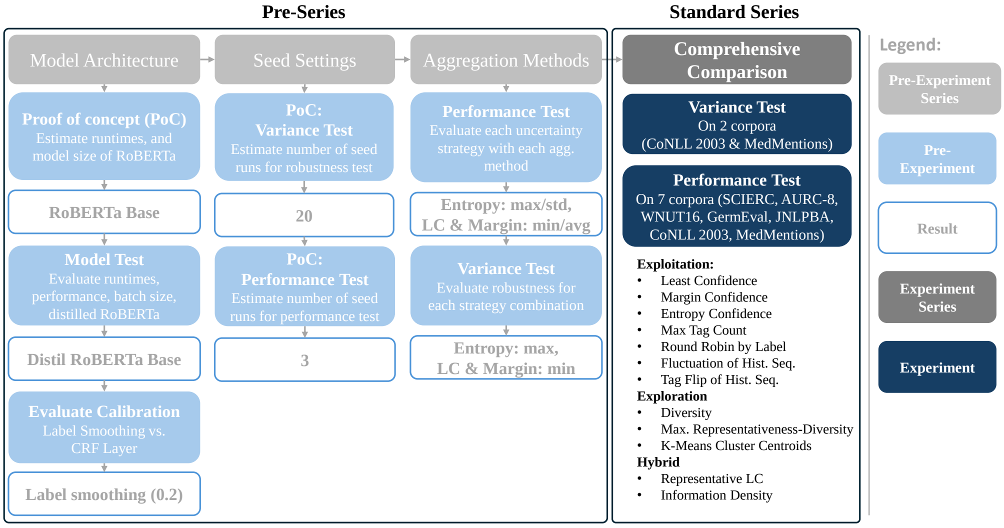

Pre-Series
Standard Series
Model Architecture
Seed Settings
Aggregation Methods
Comprehensive Comparison
Proof of concept (PoC)
Estimate runtimes, and model size of RoBERTa
PoC: Variance Test
Estimate number of seed runs for robustness test
Performance Test
Evaluate each uncertainty strategy with each agg. method
Variance Test
On 2 corpora (CoNLL 2003 & MedMentions)
RoBERTa Base
20
Entropy: max/std, LC & Margin: min/avg
Performance Test
On 7 corpora (SCIERC, AURC-8, WNUT16, GermEval, JNLPBA, CoNLL 2003, MedMentions)
Model Test
Evaluate runtimes, performance, batch size, distilled RoBERTa
PoC: Performance Test
Estimate number of seed runs for performance test
Variance Test
Evaluate robustness for each strategy combination
Distil RoBERTa Base
3
Entropy: max, LC & Margin: min
Evaluate Calibration
Label Smoothing vs. CRF Layer
Label smoothing (0.2)
Exploitation:
Least Confidence
Margin Confidence
Entropy Confidence
Max Tag Count
Round Robin by Label
Fluctuation of Hist. Seq.
Tag Flip of Hist. Seq.
Exploration
Diversity
Max. Representativeness-Diversity
K-Means Cluster Centroids
Hybrid
Representative LC
Information Density
Legend:
Pre-Experiment Series
Pre-Experiment
Result
Experiment Series
Experiment

Figure 3: Process to derive our standard evaluation setting, which was applied to each selected strategy.

6 RESULTS

The following sections describe our results regarding the performance, robustness, and data bias of the considered AL strategies.

6.1 Performance and Robustness Comparison

We assessed the performance with two methods: *Area under the learning curve (AUC)* and *Wilcoxon Signed-Rank Test (WSRT)*. AUC serves as an empirical measure to compare different strategies with each other based on the F1 macro score depending on the number of data points (see Figure 5). The larger the area under the curve, the better the strategy (Settles and Craven, 2008). The authors of (Rainio et al., 2024) recommend the WSRT to compare two models with each other based on evaluation metrics (here F1 macro score). We use it to determine which strategies are statistically significantly better than the randomizer. Then, AUC ranks these AL strategies. Figure 4 depicts the performance of each strategy and corpus. In the following, we call each combination of AL strategy and corpus a *use case* (single cell in the figure), and a *domain* is represented by a corpus and constitutes a row in the figure.

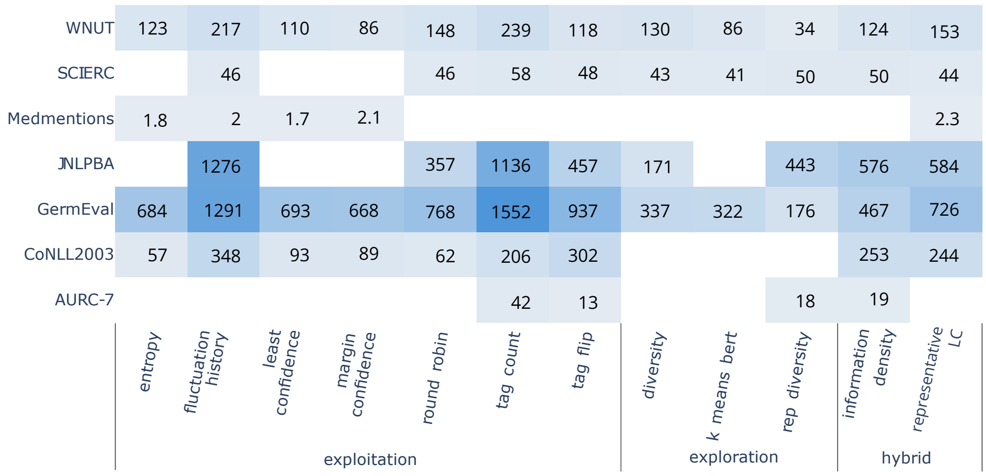
Exploration strategies show the smallest benefit. Among the selected subset of strategies — diversity (*diversity*), representative (*k means bert*), and their combination (*rep diversity*) — the combination performed best across various domains, improving 5 out of 7 use cases, while the other two improved only 3 to 4 use cases. A more extensive evaluation of further exploration strategies could provide deeper insights into this area.

The selected hybrid approaches have shown similar performance. Both improved 6 out of 7 use cases. The sequential approach (*representative LC*) was slightly better.

Among the exploitation strategies, three exhibit strong performance (*fluctuation history*, *tag count*, and *tag flip*), especially for the corpora GermEval and JNLPBA. Across the domains, they improved 6 out of 7 use cases. The other strategies show a moderate impact. Based on these results, it seems helpful to use historical information (fluctuation or flips) and documents with many entities (tag count).

We compared the hybrid strategies and their underlying exploitation methods. The integration of an exploitation approach with an exploration approach appears to extend the coverage across the use cases. For instance, the representative LC strategy, which utilizes the *least confidence* strategy, improved performance in 6 out of 7 use cases. When least confidence is applied alone, it improved 4 out of 7 use cases. A similar pattern is observed with information density, where the combination of entropy and density information demonstrates enhanced efficacy.

From the domain perspective, we made the following observations: None of the strategies is suitable for AURC-7 and Medmentions. AURC-7 is a balanced corpus with argumentation documents: each argument follows a counter-argument. Medmentions has a very high average number of entities per document (80) with only one label. In both cases, the ran-



	exploitation							exploration			hybrid	
	entropy	fluctuation history	least confidence	margin confidence	round robin	tag count	tag flip	diversity	k means bert	rep diversity	information density	representative LC
WNUT	123	217	110	86	148	239	118	130	86	34	124	153
SCIERC		46			46	58	48	43	41	50	50	44
Medmentions	1.8	2	1.7	2.1								2.3
JNLPBA		1276			357	1136	457	171		443	576	584
GermEval	684	1291	693	668	768	1552	937	337	322	176	467	726
CoNLL2003	57	348	93	89	62	206	302				253	244
AURC-7						42	13			18	19	

Figure 4: The chart displays the performance of each strategy compared to the randomizer on each corpus. White areas indicate cases where no statistically significant improvements against the randomizer were detected. All blue-shaded areas indicate statistically significant gains. The darker the shade, the better the strategy performed against the randomizer measured by AUC differences. The AUC differences are depicted in each cell.

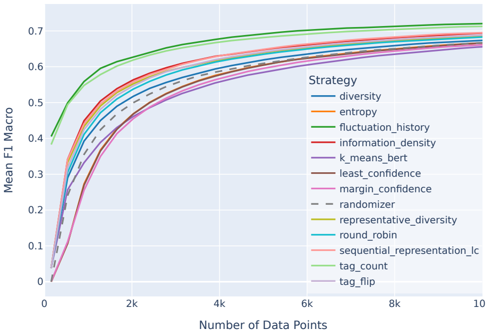


Figure 5: Mean F1 macro score on the JNLPBA test corpus. The score is averaged across three seed runs depending on the number of data points used for training (Entropy is almost fully covered by the least confidence strategy).

dom selection might gain sufficient information and cannot be improved with AL.

The strongest impact was detected for GermEval and JNLPBA, which represent the largest corpora in our test suite. See Figure 5 for the learning curves for JNLPBA as an example. Although the size of CoNLL2003 is similar to JNLPBA, we cannot see the same improvement for CoNLL2003. For GermEval and WNUT every strategy performs better than the randomizer.

We assessed the strategies' robustness via the standard deviation (see Section 5). We require that the random processes in the training process or the selection of the initial subset should not significantly impact good-performing strategies. The results show that the two best-performing strategies (fluctuation history and tag count) are also the most robust strategies. The least robust strategies are information density, representative LC, and diversity.

6.2 Bias Comparison

We also assessed the data bias and the amplification by the strategies. Inspired by (Hassan and Alikhani, 2023) on classification tasks, we extended their approach to ER. They showed that unequal label distributions infer a data bias. The authors compare the inherent label distribution of the corpora with the error distribution of the trained model. Good AL strategies should not introduce high error rates for low-frequent labels. We derived the following formula to measure the bias in our use case. Requirements:

(I) Compute the error err_l (analog to accuracy) for each label l except the O-tag. (II) Compute the normalized data distribution d_l per label l, so that you obtain values from the interval $[0,1]$ per label.

The bias per label is defined as:

$$b_l = -err_l \cdot log(d_l)$$

Errors associated with low-frequency labels tend to exacerbate bias more significantly than those linked to high-frequency labels. This measurement of bias is effective only as a comparative score within the same corpus and cannot be applied nominally across different corpora.

Our findings indicate that the strategies with the least susceptibility to bias are tag count and fluctuation history. In contrast, the strategies most amplifying bias include random selection, representative diversity, and diversity strategies. We hypothesize that the random selection strategy amplifies data bias because it mirrors the inherent data distribution. Conversely, strategies like tag count or fluctuation history appear to select beneficial subsets of data, thereby mitigating errors in low-frequency labels. This is also illustrated in Figure 5, where these strategies outperform random selection even in the region where the data sets begin to converge ($\sim$ 10k documents), further demonstrating their efficacy in reducing bias.

7 CONCLUSION

This paper conducted a comparative performance analysis of *Active Learning (AL)* strategies in the context of *entity recognition (ER)*. Based on a systematic selection of corpora and strategies, guided by a comprehensive scoping review, we conducted 115 experiments within a standardized evaluation setting. Our assessment referred to both performance and runtime. We identified conditions where AL achieved significant improvements, as well as situations where its results are more limited. Two strategies came out as clear winners: *tag count* and *fluctuation history*.

Future work may expand the evaluation to a broader range of AL strategies and corpora, including those that do not adhere to the rigorous construction standards of benchmark datasets, to explore their specific challenges.

REFERENCES

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Chen, Y., Lasko, T. A., Mei, Q., Denny, J. C., and Xu, H. (2015). A study of active learning methods for named entity recognition in clinical text. *Journal of Biomedical Informatics*, 58:11–18.

Covington, M. A. and McFall, J. D. (2010). Cutting the Gordian Knot: The Moving-Average Type–Token Ratio (MATTR). *Journal of Quantitative Linguistics*, 17(2):94–100.

Esuli, A., Marcheggiani, D., and Sebastiani, F. (2010). Sentence-based active learning strategies for information extraction. In *CEUR Workshop Proceedings*, volume 560, pages 41–45.

Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., and Hovy, E. (2021). A Survey of Data Augmentation Approaches for NLP. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 968–988, Online. Association for Computational Linguistics.

Finlayson, M. A. and Erjavec, T. (2017). Overview of Annotation Creation: Processes and Tools. In Ide, N. and Pustejovsky, J., editors, *Handbook of Linguistic Annotation*, pages 167–191. Springer Netherlands, Dordrecht.

Hassan, S. and Alikhani, M. (2023). D-CALM: A Dynamic Clustering-based Active Learning Approach for Mitigating Bias. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5540–5553, Toronto, Canada. Association for Computational Linguistics.

Huang, K.-H. (2021). DeepAL: Deep Active Learning in Python.

Jayakumar, T., Farooqui, F., and Farooqui, L. (2023). Large Language Models are legal but they are not: Making the case for a powerful LegalLLM. In Preo\textcommabelowtiuc-Pietro, D., Goanta, C., Chalkidis, I., Barrett, L., Spanakis, G., and Aletras, N., editors, *Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 223–229, Singapore. Association for Computational Linguistics.

Jehangir, B., Radhakrishnan, S., and Agarwal, R. (2023). A survey on Named Entity Recognition — datasets, tools, and methodologies. *Natural Language Processing Journal*, 3:100017.

Kaddour, J., Key, O., Nawrot, P., Minervini, P., and Kusner, M. J. (2023). No Train No Gain: Revisiting Efficient Training Algorithms For Transformer-based Language Models. *Advances in Neural Information Processing Systems*, 36:25793–25818.

Kettunen, K. (2014). Can Type-Token Ratio be Used to Show Morphological Complexity of Languages? *Journal of Quantitative Linguistics*, 21(3):223–245.

Kholghi, M., De Vine, L., Sitbon, L., Zuccon, G., and Nguyen, A. (2017). Clinical information extraction using small data: An active learning approach based on sequence representations and word embeddings. 68(11):2543–2556.

Kholghi, M., Sitbon, L., Zuccon, G., and Nguyen, A. (2015). External knowledge and query strategies in active learning: A study in clinical information extraction. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 143–152, New York, NY, USA. Association for Computing Machinery.

Klie, J.-C., Bugert, M., Boullosa, B., Eckart de Castilho, R., and Gurevych, I. (2018). The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico. Association for Computational Linguistics.

Kohl, P., Freyer, N., Krämer, Y., Werth, H., Wolf, S., Kraft, B., Meinecke, M., and Zündorf, A. (2023). ALE: A

Simulation-Based Active Learning Evaluation Framework for the Parameter-Driven Comparison of Query Strategies for NLP. In Conte, D., Fred, A., Gusikhin, O., and Sansone, C., editors, *Deep Learning Theory and Applications*, Communications in Computer and Information Science, pages 235–253, Cham. Springer Nature Switzerland.

Kohl, P., Krämer, Y., Fohry, C., and Kraft, B. (2024). Scoping Review of Active Learning Strategies and Their Evaluation Environments for Entity Recognition Tasks. In Fred, A., Hadjali, A., Gusikhin, O., and Sansone, C., editors, *Deep Learning Theory and Applications*, pages 84–106, Cham. Springer Nature Switzerland.

Lison, P., Barnes, J., and Hubin, A. (2021). Skweak: Weak Supervision Made Easy for NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 337–346.

Liu, M., Tu, Z., Zhang, T., Su, T., Xu, X., and Wang, Z. (2022). LTP: A new active learning strategy for CRF-Based named entity recognition. *Neural Processing Letters*, 54(3):2433–2454.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Montani, I. and Honnibal, M. Prodigy: A modern and scriptable annotation tool for creating training data for machine learning models. *Prodigy*, Explosion.

Nakayama, H., Kubo, T., Kamura, J., Taniguchi, Y., and Liang, X. (2018). Doccano: Text Annotation Tool for Human. https://github.com/doccano/doccano.

Rainio, O., Teuho, J., and Klén, R. (2024). Evaluation metrics and statistical tests for machine learning. *Scientific Reports*, 14(1):6086.

Ramshaw, L. and Marcus, M. (1995). Text Chunking using Transformation-Based Learning. In *Third Workshop on Very Large Corpora*.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Gupta, B. B., Chen, X., and Wang, X. (2022). A Survey of Deep Active Learning. *ACM Computing Surveys*, 54(9):1–40.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2020). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter.

Şapci, A., Kemik, H., Yeniterzi, R., and Tastan, O. (2023). Focusing on potential named entities during active label acquisition. *Natural Language Engineering*.

Schröder, C. and Niekler, A. (2020). A Survey of Active Learning for Text Classification using Deep Neural Networks.

Settles, B. (2009). Active Learning Literature Survey. Technical Report, University of Wisconsin-Madison Department of Computer Sciences.

Settles, B. and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 1070–1079, USA. Association for Computational Linguistics.

Sintayehu, H. and Lehal, G. S. (2021). Named entity recognition: A semi-supervised learning approach. *International Journal of Information Technology*, 13(4):1659–1665.

Van Nguyen, M., Ngo, N., Min, B., and Nguyen, T. (2022). FAMIE: A Fast Active Learning Framework for Multilingual Information Extraction. In *NAACL 2022 - 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Demonstrations Session*, pages 131–139.

Wang, D.-B., Feng, L., and Zhang, M.-L. (2021). Rethinking Calibration of Deep Neural Networks: Do Not Be Afraid of Overconfidence. In *Advances in Neural Information Processing Systems*, volume 34, pages 11809–11820. Curran Associates, Inc.

Wang, W., Zheng, V. W., Yu, H., and Miao, C. (2019). A Survey of Zero-Shot Learning: Settings, Methods, and Applications. *ACM Trans. Intell. Syst. Technol.*, 10(2):13:1–13:37.

Yang, M. (2021). A Survey on Few-Shot Learning in Natural Language Processing. In *2021 International Conference on Artificial Intelligence and Electromechanical Automation (AIEA)*, pages 294–297.

Yang, Y.-Y., Lee, S.-C., Chung, Y.-A., Wu, T.-E., Chen, S.-A., and Lin, H.-T. (2017). Libact: Pool-based Active Learning in Python.

Yao, J., Dou, Z., Nie, J., and Wen, J. (2020). Looking Back on the Past: Active Learning with Historical Evaluation Results. *IEEE Transactions on Knowledge and Data Engineering*.

Zhan, X., Wang, Q., Huang, K.-h., Xiong, H., Dou, D., and Chan, A. B. (2022). A Comparative Survey of Deep Active Learning.

Zhang, Z., Strubell, E., and Hovy, E. (2022). A Survey of Active Learning for Natural Language Processing. In Goldberg, Y., Kozareva, Z., and Zhang, Y., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zheng, G., Mukherjee, S., Dong, X. L., and Li, F. (2018). OpenTag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp; Data Mining*, KDD '18, pages 1049–1058, New York, NY, USA. Association for Computing Machinery.

Intrinsic Evaluation of RAG Systems for Deep-Logic Questions*

Junyi (Edward) Hu[a], You Zhou[b] and Jie Wang[c]
Miner School of Computer & Information Sciences, University of Massachusetts, Lowell, MA, U.S.A.
{junyi_hu, you_zhou1}@student.uml.edu, jie_wang@uml.edu

Keywords: Retrieval Augmented Generation, Logical-Relation Correctness Ratio, Overall Performance Index.

Abstract: We introduce the Overall Performance Index (OPI), an intrinsic metric to evaluate retrieval-augmented generation (RAG) mechanisms for applications involving deep-logic queries. OPI is computed as the harmonic mean of two key metrics: the Logical-Relation Correctness Ratio and the average of BERT embedding similarity scores between ground-truth and generated answers. We apply OPI to assess the performance of LangChain, a popular RAG tool, using a logical relations classifier fine-tuned from GPT-4o on the RAG-Dataset-12000 from Hugging Face. Our findings show a strong correlation between BERT embedding similarity scores and extrinsic evaluation scores. Among the commonly used retrievers, the cosine similarity retriever using BERT-based embeddings outperforms others, while the Euclidean distance-based retriever exhibits the weakest performance. Furthermore, we demonstrate that combining multiple retrievers, either algorithmically or by merging retrieved sentences, yields superior performance compared to using any single retriever alone.

1 INTRODUCTION

A RAG system typically consists of two major components: Indexing and Retrieval. The former is responsible for indexing a reference text document before any queries are made to it. The latter is responsible for retrieving relevant data from the indexed document in response to a query and passing that information, along with the query, to a large language model (LLM) to generate an answer. The Retrieval component is typically a framework that supports a variety of retrieval methods, each referred to as a retriever.

To assess the effectiveness of a retriever in uncovering the logical relationship for an answer to a query with respect to the reference document, we introduce the Overall Performance Index (OPI). This metric measures both the correctness of the answers generated by an LLM and the accuracy of the logical relations produced by a classifier. The OPI is calculated as the harmonic mean of the BERT embedding similarity between ground-truth and generated answers, and the logical-relation correctness ratio.

To demonstrate the effectiveness of the OPI metric, we use the RAG-Dataset-12000 provided by Hugging Face (D.H., 2024) as the training and testing dataset. We fine-tune GPT-4o to construct a classifier to generate logical relations between a query and an answer, with respect to the reference document. We then evaluate LangChain (LangChain, 2024), a popular RAG tool, with seven common retrievers, extracting relevant sentences from the reference document for each query. Using GPT-4o as the underlying LLM, we generate an answer to the query and use the fine-tuned GPT-4o classifier to generate a logical relation.

To rank retrievers, we calculate the average OPI score across all 13 logical relations provided in RAG-Dataset-12000. We then use OPI to analyze the strengths and weaknesses of individual retrievers. Moreover, we demonstrate that several variations of combining multiple retrievers, either algorithmically or by merging retrieved sentences, outperform a single retriever alone.

2 PRELIMINARIES

The technique of RAG was introduced by Lewis et al. (2020) (Lewis et al., 2020) a few years before the widespread adoption of LLMs. The performance of a RAG system relies on the quality of the underlying retriever and the ability of the underlying LLM.

[a] https://orcid.org/0000-0001-8524-0123
[b] https://orcid.org/0009-0005-0919-5793
[c] https://orcid.org/0000-0003-1483-2783
* This work was supported in part by Librum Technologies, Inc.

Hu, J., Zhou, Y. and Wang, J.
Intrinsic Evaluation of RAG Systems for Deep-Logic Questions.
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 489-496
ISBN: 978-989-758-716-0; ISSN: 2184-3228

LangChain is a popular RAG tool, which divides a reference document into overlapping text chunks of equal size. The suffix of each chunk overlaps with the prefix of the next.

To the best of our knowledge, no previous research has comprehensively evaluated the performance of RAG systems in the context of deep-logic question answering.

Given below are seven common sentence retrievers supported by LangChain:

DPS (dot-product similarity) converts a query and a text chunk as BERT-based (Devlin et al., 2018) embedding vectors and compute their dot product as a similarity score. It returns k chunks with the highest scores to the query. (DPS in LangChain is referred to as Cosine Similarity.)

kNN (k-Nearest Neighbors) in LangChain is the normalized dot-product similarity by the L2-norm, which is widely referred to as the cosine similarity. It returns k chunks with the highest cosine similarity scores to the query.

BM25 (Robertson and Zaragoza, 2009) is a probabilistic information retrieval model that ranks documents based on the term frequency in a chunk and the inverse chunk frequency in the reference document. Let q be a query, T a chunk of text, $f(t_i,T)$ the frequency of term t_i in T, $|T|$ the size of T, avgTL the average chunk length, N the total number of chunks, and $n(t_i)$ the number of chunks that contain t_i. Then $\mathrm{BM25}(q,T)$ is defined by

$$\mathrm{BM25}(q,T) = \sum_{i=1}^{n} \ln\left(\frac{N-n(t_i)+0.5}{n(t_i)+0.5}+1\right) \cdot \frac{(f(t_i,T)\cdot(\kappa+1))}{f(t_i,T)+\kappa\cdot(1-b+b\cdot\frac{|T|}{\mathrm{avgTL}})},$$

where κ and b are parameters. Return k chunks of text with the highest BM25 scores to the query.

SVM (Support Vector Machine) (Cortes and Vapnik, 1995) is a supervised learning model that finds the hyperplane that best separates data points in a dataset. To use SVM as a retriever, first represent each chunk of text as a feature vector. This can be done using word embeddings, TF-IDF, or any other vectorization method. Then use the labeled dataset to train an SVM model. Convert the query into the same feature vector space as the chunks. Apply the SVM model to the query vector to produce a score that indicates how similar the query is to each chunk. Extract k chunks with the highest scores.

TF-IDF (Sammut and Webb, 2011) measures the importance of a word in a chunk of text relative to the set of chunks in the reference document, combining term frequency and inverse chunk frequency. In particular,

$$\mathrm{TF\text{-}IDF}(t,T) = \mathrm{TF}(t,T) \times \mathrm{IDF}(t),$$

where t is a term, T is a chunk, and $\mathrm{IDF}(t)$ is the inverse chunk frequency of t. Given a query q, select k chunks with the highest $\mathrm{TF\text{-}IDF}(q,T)$ values.

MMR (Carbonell and Goldstein, 1998) is a retrieval algorithm that balances relevance and diversity in the selection of k chunks. It iteratively selects chunks that are both relevant to the query and minimally redundant with respect to the chunks already selected.

EDI (Euclidean Distance) (Bishop, 2006) measures the straight-line distance between a query and a chunk, represented in bag-of-words vectors. Return k chunks with the shortest distance to the query.

A data point in RAG-Database-12000 contains the following attributes: ‘context’, ‘question’, ‘answer’, ‘retrieved_sentences’, ‘logical_relation’, where ‘context’ is the reference document. There are thirteen categories of logical reasoning in the dataset. Their names, abbreviations, descriptions, and the distribution of counts are presented in Table 1. All but the last category involve deep logical reasoning, meaning that arriving at the correct answer requires complex, multi-step processes involving multiple concepts, facts, or events extracted from the content. The table includes eleven specific types of deep reasoning, with an additional category for general deep reasoning, referred to as multi-hop reasoning.

3 OVERALL PERFORMANCE INDEX

Let A and LR denote, respectively, the ground-truth answer and logical relation to the question with respect to the question Q, the context C, and the retrieved sentences S. Let A' and LR' denote, respectively, the answer and the logical relation generated by a RAG system with an LLM. We represent A and A' using BERT embeddings and compute the cosine similarity of the embeddings.

For a given dataset D with respect to a particular logical relation LR, let $\mathrm{BERTSim}_D$ denote the average BERT similarity scores of all (A,A') pairs and LRCR_D (logical-relation correctness ratio) denote the proportion of data points where the predicted logical relation matches LR. Namely,

$$\mathrm{LRCR}_D = \frac{|\{d \in D \mid LR = LR'\}|}{|D|} \qquad (1)$$

The OPI for dataset D is defined by the following parameterized harmonic mean of $\mathrm{BERTSim}_D$ and LRCR_D, similar to defining the F-measure (Lewis and Gale, 1994).

Table 1: Information of logical relations.

Logical Relation	Description	Count	Total
Adversarial (ADV)	The answer involves opposing perspectives or arguments.	156	5,080
Analogical (ANA)	The answer is based on similarities between different things or situations.	116	
Causal (CAU)	The answer is based on cause-and-effect relationships.	2,477	
Comparative (COM)	The answer compares and contrasts different items.	129	
Conditional (CON)	The answer is based on conditions or if-then statements.	161	
Deductive (DED)	The answer is a conclusion of multiple statements	140	
Fuzzy (FUZ)	The answer is a generalizing statement.	106	
Inductive (IND)	The answer involves uncertain or imprecise information.	152	
Multi-hop (MUH)	The answer involves multiple steps or connections.	198	
Predictive (PRE)	The answer makes predictions.	151	
Spatial (SPA)	The answer involves relationships based on locations.	1,077	
Temporal (TEM)	The answer involves relationships based on time.	217	
Direct Matching	The answer is based on a straightforward match of the extracted content.	6,920	6,920

$$\text{OPI}(\beta)_D = \frac{(1+\beta^2)\cdot \text{BERTSim}_D \cdot \text{LRCR}_D}{(\beta^2 \cdot \text{BERTSim}_D) + \text{LRCR}_D}. \quad (2)$$

$\text{OPI}(1)_D$ weighs answer accuracy and logical relation accuracy equally. $\text{OPI}(\beta)_D$ weighs answer accuracy more heavily when $\beta > 1$ (e.g., $\beta = 2$), and weighs logical relation accuracy more heavily when $\beta < 1$ (e.g., $\beta = 0.5$).

When there is no confusion in the context, the subscript D is omitted. Denote $\text{OPI}(1)$ as OPI-1, $\text{OPI}(2)$ as OPI-2, and $\text{OPI}(0.5)$ as OPI-0.5.

In addition to BERTSim, other metrics may be used to measure the similarity between the generated answer and the ground-truth answer, such as Hugging Face's MoverScore, as applied in the study of content significance distributions of text blocks in a document (Zhou and Wang, 2023). We choose BERTSim because MoverScore uses IDF to compute word weights, which is better suited for extractive answers but less appropriate for generative answers produced by LLMs.

Experimental results show that the BERTSim metric aligns well with the outcomes of extrinsic comparisons of the ground-truth answers with the generated answers (see Section 4.2 for details).

In what follows, we will use OPI-1 as the default intrinsic measure to study the performance of RAG systems for answering deep-logic questions.

4 EVALUATION

As seen in Table 1, the data points in RAG-Dataset-12000 are unevenly distributed across the 13 logical relations, with significant disparities, such as only 106 data points in Fuzzy Reasoning compared to 6,920 data points in Direct Matching. To fine-tune GPT-4o and construct a classifier for identifying logical relations, a balanced dataset is preferred. To achieve this, we randomly select 100 data points from each logical relation category, forming a new dataset called RAG-QA-1300 that consists of 1,300 data points. This dataset is then split with an 80-20 ratio to create a training set and a test set.

Fine-tuning was performed by combining the context, question, and answer from each data point into a cohesive input text, labeled with its corresponding logical relation. The process involved approximately 800 training steps, resulting in a validation loss of 10^{-4}. This specific checkpoint was selected for its optimal performance.

The fine-tuned GPT-4o classifier for logical relations significantly improves the accuracy to 75.77% on the test set, compared to 49.23% when using GPT-4o out-of-the-box without fine-tuning.

We used LangChain with the seven common retrievers mentioned in Section 2. We used GPT-4o to generate answers and the fine-tuned GPT-4o classifier to generate logical relations. LangChain supports a wide range of retrievers and allows for the seamless integration of pre-trained LLMs.

4.1 Intrinsic Evaluation

We set the chunk size to 100 (words) with a chunk overlap of 20 % in the setting of LangChain, where paragraph breaks, line breaks, periods, question marks, and exclamation marks are set to be the separators. These settings were fed into the LangChain function `RecursiveCharacterTextSplitter` to split a reference document into chunks, where each chunk contains up to 100 words, ending at a specified separator to break naturally such that the chunk is as large as possible, and adjacent chunks have a 20%

Table 2: Intrinsic comparisons across all logical relations, where "Retr" is an abbreviation of Retriever, "B" stands for BERTSim, "L" for LRCR, and "O-1" for OPI-1.

															Rank		
Retr	**Metrics**	**ADV**	**ANA**	**CAU**	**COM**	**CON**	**DED**	**DIM**	**FUZ**	**IND**	**MUH**	**PRE**	**SPA**	**TEM**	**Avg**	**B**	**L**	**O-1**
DPS	BERSim	0.78	0.79	0.85	0.80	0.82	0.84	0.81	0.67	0.84	0.82	0.83	0.87	0.84	0.8113	2		
	LRCR	0.45	0.70	0.60	0.75	0.80	0.70	0.75	0.60	0.60	0.60	0.70	0.75	0.75	0.6731		2	
	OPI-1	0.57	0.74	0.70	0.78	0.81	0.76	0.75	0.63	0.70	0.69	0.76	0.81	0.79	0.7358			2
kNN	BERTSim	0.77	0.79	0.83	0.81	0.82	0.81	0.83	0.73	0.85	0.82	0.84	0.90	0.80	0.8162	**1**		
	LRCR	0.55	0.70	0.60	0.70	0.65	0.70	0.80	0.60	0.60	0.65	0.75	0.80	0.70	0.6769		**1**	
	OPI-1	0.64	0.74	0.70	0.75	0.73	0.75	0.81	0.66	0.70	0.72	0.79	0.85	0.75	0.7401			**1**
BM25	BERTSim	0.80	0.80	0.82	0.81	0.78	0.81	0.79	0.75	0.82	0.78	0.81	0.88	0.76	0.8020	6		
	LRCR	0.60	0.60	0.60	0.70	0.60	0.65	0.70	0.65	0.60	0.65	0.65	0.75	0.75	0.6538		4	
	OPI-1	0.69	0.69	0.69	0.75	0.68	0.72	0.74	0.70	0.69	0.71	0.72	0.81	0.76	0.7204			4
SVM	BERTSim	0.76	0.74	0.83	0.82	0.85	0.80	0.80	0.67	0.84	0.82	0.79	0.90	0.84	0.8039	5		
	LRCR	0.58	0.61	0.64	0.63	0.69	0.61	0.68	0.62	0.62	0.60	0.63	0.72	0.71	0.6418		6	
	OPI-1	0.66	0.67	0.72	0.71	0.76	0.69	0.74	0.65	0.71	0.69	0.70	0.80	0.77	0.7137			6
TF-IDF	BERTSim	0.80	0.83	0.81	0.80	0.82	0.80	0.79	0.73	0.82	0.80	0.77	0.90	0.83	0.8069	4		
	LRCR	0.65	0.80	0.80	0.65	0.60	0.70	0.75	0.55	0.65	0.65	0.45	0.60	0.70	0.6577		3	
	OPI-1	0.72	0.81	0.80	0.72	0.69	0.75	0.77	0.63	0.72	0.72	0.57	0.72	0.76	0.7247			3
MMR	BERTSim	0.75	0.81	0.83	0.84	0.82	0.82	0.85	0.71	0.84	0.79	0.83	0.85	0.76	0.8074	3		
	LRCR	0.58	0.63	0.63	0.62	0.67	0.63	0.72	0.64	0.61	0.61	0.64	0.69	0.66	0.6420		5	
	OPI-1	0.66	0.71	0.72	0.71	0.74	0.71	0.78	0.67	0.71	0.69	0.73	0.76	0.71	0.7153			5
EDI	BERTSim	0.74	0.71	0.83	0.79	0.77	0.82	0.81	0.70	0.84	0.77	0.76	0.66	0.74	0.7646	7		
	LRCR	0.50	0.55	0.70	0.70	0.60	0.70	0.75	0.55	0.65	0.60	0.70	0.45	0.55	0.6154		7	
	OPI-1	0.60	0.62	0.76	0.74	0.68	0.76	0.78	0.62	0.73	0.67	0.73	0.53	0.63	0.6819			7

overlap.

We used the default settings for each retriever to return four chunks in the context with the best scores—highest for similarity and ranking measures, smallest for distance measures—from the underlying retriever as the most relevant to the query. We then converted the four chunks extracted by the retriever back into complete sentences as they appeared in the original article. These sentences and the query were then fed to GPT-4o to generate an answer. Moreover, we instructed GPT-4o to determine the logical relationship for the answer with respect to the input text.

We consider the accuracy of the generated answers and logical relations to be equally important. Table 2 presents the evaluation results of OPI-1 on the test data of RAG-QA-1300. The OPI-1 score with respect to each retriever is calculated for each set of data points of the same logical relation. The average OPI-1 score for each retriever across all logical relations is calculated by

$$\text{OPI-1} = \frac{2/|L| \cdot \sum_{\ell \in L} \text{BERTSim}_\ell \cdot \sum_{\ell \in L} \text{LRCR}_\ell}{\sum_{\ell \in L} \text{BERTSim}_\ell + \sum_{\ell \in L} \text{LRCR}_\ell}, \quad (3)$$

where L is the set of the 13 logical relations, and BERTSim_ℓ and LRCR_ℓ denote, respectively, the corresponding BERTSim score and LRCR value for the logical relation ℓ.

An alternative is to calculate the average OPI-1 score across all logical relations. While this differs slightly from Formula (3), the difference is minimal. We prefer Formula (3) for practical efficiency, as it bypasses the need to compute individual OPI-1 scores for each logical relation when these scores are not needed in applications, streamlining the process and reducing unnecessary computations.

4.2 Extrinsic Evaluation

The extrinsic evaluation uses a 0-3-7, 3-point scoring system to score A' for each pair (A, A'), where A is the ground-truth answer and A' is the answer generated directly by GPT-4o based on the question Q and the extracted sentences in the corresponding data point. The scoring system works as follows:

- A' receives 7 points if it exactly matches the meaning of A.
- A' receives 3 points if it partially matches the meaning of A, with or without extra information not contained in A.
- A' receives 0 points if there is no meaningful match.

This scoring system was designed to simplify extrinsic evaluation while widening the gaps between perfect, partial, and no matches, providing a more nuanced assessment compared to the traditional 1-2-3, 3-point scoring system. The intention is to create a ranking with greater discrepancy between scores, enabling clearer differentiation of performance levels and allowing for more meaningful comparisons.

Given below are examples of each scenario.

- 7 points: A' receives 7 points if A and A' have exactly the same meaning, with neither additional nor missing information. For example, the following A' using DPS is a 7-point answer.
  - Q: What is the debate about what Desmond said in the context?
  - A: The debate is about what Desmond said to Locke, specifically whether he said “snowman” or “silhouette”.
  - A': The debate is about whether Desmond said “snowman” or “silhouette”.
- 3 points. A' receives 3 points in either of the following two cases:

  Case 1: Extra information. A' matches the meaning of A, but includes additional information not present in A. For example, the following A' using MMR receives 3 points for the reason that A' contains extra information that the Ford GT40 was produced in limited runs from 1964 to 1969.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: The Ford GT40 was created as a result of a grudge between Henry Ford II and Enzo Ferrari. After a failed attempt to acquire Ferrari, Henry Ford II decided to compete against Ferrari in the prestigious 24 Hours of Le Mans race. To break Ferrari’s dominance in the race, Ford’s chief engineer, Roy Lunn, came up with the concept of the GT40.
  - A': The Ford GT40 was born out of a rivalry between Henry Ford II and Enzo Ferrari. After a failed attempt to acquire Ferrari, Henry Ford II resolved to challenge Ferrari in the prestigious 24 Hours of Le Mans race. To disrupt Ferrari’s dominance, Ford’s chief engineer, Roy Lunn, conceived the idea for the GT40. It was produced in limited runs from 1964 to 1969.

  Case 2: Partial match. The meaning of A' partially overlaps with the meaning of A, but not fully. For example, the following A' generated using BM25 receives 3 points. Reason: A' clearly leaves out information that the Ford GT40 was conceived by Ford’s chief engineer, Roy Lunn.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: See Case 1 above.
  - A': The reason behind the creation of the Ford GT40 was to compete against Ferrari in racing events, as evidenced by Ford’s continued efforts to improve the GT40 and best the Italians.
- 0 points: A' receive 0 points if A and A' are distinct from each other with no overlap in meaning. For example, the following A' generated through EDI receives 0 points.
  - Q: What was the reason behind the creation of the Ford GT40?
  - A: See Case 1 above.
  - A': The Ford GT40 was created to take full advantage of the benefits associated with a mid-engine design, including a slinky aerodynamic shape and benign handling characteristics.”

Table 3 shows the average scores of comparing answers by freelance annotators as well as the corresponding BERTSim scores. The integers in the row below the row of evaluation scores represent the respective rankings.

Table 3: Evaluation scores by extrinsic evaluation and intrinsic BERTSim metric with rankings, where “Extr” stands for “extrinsic evaluation” and “Intr” for “intrisic evaluation”.

	DPS	kNN	BM25	SVM	TF-IDF	MMR	EDI
Extr	2.8654	2.8654	2.7923	2.8615	2.8654	2.9077	2.6038
	2	2	6	5	2	1	7
Intr	0.8113	0.8162	0.8020	0.8039	0.8069	0.8074	0.7646
	2	1	6	5	4	3	7

It is evident that the extrinsic evaluation scores align well with the BERTSim scores, demonstrating consistency in ranking. In particular, both evaluations are in complete agreement for the 2nd, 5th, 6th, and 7th places, with only minor variations in the other rankings. For instance, MMR is ranked 1st by extrinsic evaluation and 3rd by BERTSim, which is quite close. Similarly, TF-IDF is ranked 2nd by extrinsic evaluation and 4th by BERTSim. Notably, DPS, kNN, and TF-IDF all share the 2nd rank in extrinsic evaluation, likely due to the coarseness of human annotation. Since DPS and kNN are essentially the same measures, they should logically be ranked closer to each other than to TF-IDF. Therefore, the extrinsic rank of TF-IDF, while differing slightly from BERTSim, can still be considered reasonably aligned. Overall, this suggests a strong correlation between the two evaluation methods.

4.3 Combining Multiple Retrievers

LangChain supports combining multiple retrievers into a new retriever. We use the default setting to return four chunks for each combination. This approach diversifies the retrieved content from the reference document, potentially improving overall performance.

Table 4: Evaluation results of various combinations of retrievers and sentences.

																Rank	
Retr	**Metrics**	**ADV**	**ANA**	**CAU**	**COM**	**CON**	**DED**	**DIM**	**FUZ**	**IND**	**MUH**	**PRE**	**SPA**	**TEM**	**Avg**	**B**	**L**	**O-1**
A-Seven	BERSim	0.81	0.81	0.86	0.85	0.85	0.84	0.86	0.66	0.86	0.84	0.81	0.91	0.85	0.8325	**1**		
	LRCR	0.65	0.70	0.75	0.70	0.75	0.70	0.75	0.55	0.70	0.70	0.65	0.75	0.70	0.6962		**1**	
	OPI-1	0.72	0.75	0.80	0.77	0.80	0.76	0.75	0.60	0.77	0.76	0.72	0.82	0.77	0.7583			**1**
A-Four	BERTSim	0.81	0.81	0.86	0.84	0.85	0.82	0.82	0.72	0.86	0.84	0.82	0.90	0.80	0.8276	2		
	LRCR	0.50	0.70	0.75	0.75	0.70	0.75	0.70	0.60	0.70	0.70	0.65	0.75	0.70	0.6885		2	
	OPI-1	0.62	0.75	0.80	0.79	0.77	0.78	0.76	0.65	0.77	0.76	0.72	0.82	0.75	0.7516			2
A-Two	BERTSim	0.76	0.79	0.83	0.80	0.84	0.82	0.76	0.73	0.86	0.82	0.83	0.89	0.80	0.8099	3		
	LRCR	0.50	0.70	0.60	0.70	0.80	0.65	0.75	0.65	0.60	0.65	0.70	0.75	0.70	0.6731		3	
	OPI-1	0.60	0.74	0.70	0.75	0.82	0.72	0.76	0.69	0.71	0.73	0.76	0.81	0.75	0.7352			3
S-Seven	BERSim	0.81	0.84	0.87	0.89	0.83	0.85	0.82	0.68	0.88	0.85	0.77	0.90	0.80	0.8310	**1**		
	LRCR	0.65	0.70	0.75	0.75	0.70	0.65	0.75	0.55	0.65	0.75	0.65	0.75	0.75	0.6962		**1**	
	OPI-1	0.72	0.76	0.81	0.81	0.76	0.74	0.75	0.61	0.75	0.80	0.70	0.82	0.78	0.7576			**1**
S-Four	BERTSim	0.82	0.81	0.87	0.85	0.86	0.83	0.83	0.73	0.87	0.84	0.82	0.91	0.80	0.8330	2		
	LRCR	0.50	0.75	0.75	0.70	0.70	0.75	0.70	0.60	0.70	0.75	0.65	0.75	0.70	0.6923		2	
	OPI-1	0.62	0.78	0.80	0.77	0.77	0.79	0.76	0.66	0.78	0.79	0.73	0.82	0.75	0.7562			2
S-Two	BERTSim	0.76	0.79	0.83	0.81	0.84	0.82	0.76	0.73	0.86	0.82	0.83	0.89	0.80	0.8120	3		
	LRCR	0.50	0.70	0.60	0.70	0.80	0.70	0.75	0.65	0.60	0.65	0.65	0.75	0.70	0.6731		3	
	OPI-1	0.60	0.74	0.70	0.75	0.82	0.75	0.76	0.69	0.71	0.73	0.73	0.81	0.75	0.7360			3

As examples, we combine all seven retrievers, denoted as A-Seven; three retrievers with the highest OPI-1 scores: kNN, DPS, and TF-IDF, plus MMR for its strength in balancing relevance and diversity, denoted as A-Four; and two retrievers with the highest OPI-1 scores: kNN and DPS, denoted as A-Two.

We may also combine the sentences retrieved by individual retrievers, removing any duplicates, and use the remaining set of sentences with the corresponding questions to generate answers and logical relations. Let S-Seven, S-Four, and S-Two denote the sets of sentences obtained this way by the corresponding retrievers as in A-Seven, A-Four, and A-Two.

The experimental results of both types of combinations are shown in Table 4.

5 ANALYSIS

We first analyze the performance of individual retrievers, followed by examining the combinations of retrievers and the sentences retrieved by multiple retrievers.

5.1 Individual Retrievers

For each retriever, we first analyze the performance for each logical relation individually and then assess the overall performance across all logical relations.

5.1.1 Individual Logical Relation

We use the OPI-1 scores to help identify the strengths and weaknesses of individual retrievers across the 13 logical relations. For example, as seen in Table 2, almost all retrievers tend to perform the worst on adversarial reasoning, followed by fuzzy reasoning. For other logical relations, the performance of retrievers varies, indicating that certain retrievers may be more suited to specific types of reasoning tasks while struggling with others. For example, even for the worst-performing retriever, EDI, which consistently ranks the lowest in both extrinsic and intrinsic evaluations of answer accuracy as seen in Table 3, it still performs best on deductive reasoning. This suggests that while EDI may generally be less effective across various logical relations, it has a particular strength in handling tasks that involve deductive reasoning. This example highlights the nuanced performance of retrievers, where even a generally weaker retriever can excel in specific logical tasks. This variability in performance highlights the importance of selecting the appropriate retriever.

5.1.2 Across all Logical Relations

The average OPI-1 scores provide a means to identify, across all 13 logical relations, which retrievers are more suitable for specific tasks and which retrievers should be avoided. For example, as shown in Table 2, EDI has the lowest and SVM the second-lowest average OPI-1 scores, indicating they should generally be avoided. This is likely due to the limitations of the underlying features used to compute SVM scores and the coarseness of L2-norms when representing text chunks as bag-of-word vectors, which may fail to capture the nuanced relationships required for deep logical reasoning tasks.

On the other hand, kNN has the highest and DPS the second-highest average OPI-1 scores, indicating that these retrievers would be the best choices for answering deep-logic questions. kNN (cosine similarity) and DPS are similar measures, with kNN being a normalized version of DPS, which explains their comparable performance. However, kNN takes slightly more time to compute than DPS, as DPS is the fastest among all seven retrievers—dot products are the simplest and quickest to compute compared to the operations used by other retrievers.

The MMR retriever allows GPT-4o to generate better answers across all logical relations, as shown in Table 3 . However, it does not perform as well in producing the correct logical relations. This discrepancy may be attributed to MMR's focus on balancing relevance and diversity in retrieved content, which improves answer quality but doesn't necessarily align with capturing accurate logical relations.

BM25 is in general more effective for retrieving longer documents in a document corpus with the default parameter values for κ and b. However, to retrieve sentences from an article, it was shown that BM25 would should use different parameter values (Zhang et al., 2021). This explains why BM25 is the second worse for generating answers as shown in Table 3 by both extrinsic and intrinsic evaluations. It is not clear, however, why it produces a relatively higher LRCR value.

TF-IDF's performance falls in the middle range, which is expected. As a frequency-based approach, it may struggle to capture deeper semantic information, but it remains relatively effective because it retains lexical information, ensuring that important terms are still emphasized in the retrieval process.

5.2 Performance of Various Combinations

We first analyze the performance of combinations of retrievers versus individual retrievers, followed by an analysis of combining retrievers algorithmically versus combining sentences retrieved by individual retrievers within the combination.

5.2.1 Combinations vs. Individuals

It can be seen from Table 4 that A-Seven outperforms A-Four, which in turn outperforms A-Two. A similar ranking is observed with S-Seven, S-Four, and S-Two. Moreover, both A-Seven and A-Four are substantially better than the top performer, kNN, when only a single retriever is used (see both Tables 2 and 4). A similar result is observed with S-Seven and S-Four, where combining more retrieved sentences from different retrievers also enhances performance, reinforcing the benefits of increased diversity in the retrieval process. These results all confirm the early suggestion that combining more retrievers generally enhances performance in both algorithmic and sentence-based combinations, supporting the idea that diverse retrieval methods contribute positively to the overall effectiveness of the RAG system.

However, we also observe that some combinations of retrievers may actually lead to poorer performance compared to using the individual retrievers alone. This is evident in the case of A-Two and S-Two, the algorithmic and sentence combinations of kNN and DPS, both result in slightly lower average OPI-1 scores than kNN alone. This is probably due to the fact that kNN and DPS are very similar measures, and combining them doesn't significantly increase diversity. Worse, the extra information provided through their combination seems to have led to diminishing returns, negating the potential benefits of combining retrievers to improve performance. This phenomenon warrants further investigation.

Nevertheless, combining retrievers based on different retrieval methodologies could help increase diversity and, consequently, improve overall performance. This is evident in the case of A-Seven and S-Seven, which combine retrievers utilizing diverse retrieval methods, as well as in A-Four and S-Four, where MMR—a retrieval method that balances relevance and diversity—complements kNN. By leveraging varied retrieval techniques, we can ensure that a broader range of relevant content is retrieved, potentially leading to greater accuracy and more robust logical reasoning in the generated answers.

5.2.2 Combining Algorithms vs. Combining Sentences

We compare the outcomes of combining retrievers at the algorithm level versus the sentence level. Combining retrievers at the algorithm level is a feature supported by LangChain, which returns the same default number of chunks before sentences are extracted. In contrast, combining retrievers at the sentence level involves merging sentences retrieved by individual retrievers, which may include more sentences than the algorithmic combination, and so should lead to a slightly better performance. This is evident when comparing A-Four with S-Four and A-Two with S-Two (see Table 4).

However, having more sentences may not always lead to improvement, as it can introduce conflicting information. This is evident when comparing A-Seven with S-Seven, where S-Seven has a lower average OPI-1 score than A-Seven. This is likely because A-Seven has already saturated the useful sentences, while S-Seven introduces additional sentences that negatively impact the average OPI-1 score.

In summary, these analyses suggest that, when combining appropriate retrievers, both algorithmic and sentence-level approaches offer performance improvements, with each method providing distinct advantages in terms of retrieval diversity and the quality of generated answers. Selecting appropriate retrievers requires a deeper understanding of the underlying retrieval mechanisms, making this an interesting topic for further investigation.

6 FINAL REMARKS

This paper presents an effective intrinsic evaluate method for the performance of RAG systems in connection to question-answering involving deep logical reasoning.

LangChain supports a wide range of retrievers and allows users to integrate custom retrievers. Additionally, there are numerous large language models (LLMs) such as the Gemini series (Google, 2024), LlaMA series (Meta, 2024), and Claude series (Claude AI, 2024), among others, as well as various retrieval-augmented generation (RAG) tools like LLAMAINDEX (LlamaIndex, 2024), HayStack (Deepset, 2024), EmbedChain (EmbedChain, 2024), and RAGatouille (AnswerDotAI, 2024). Evaluating the performance of these models and tools, particularly for answering deep-logic questions where identifying logical relations is essential, represents an intriguing direction for future research.

Regularly reporting the findings of such investigations would significantly contribute to the advancement of RAG technologies. Furthermore, we aim to develop a tool that quantitatively assesses the depth of logical relations in question-answering systems relative to the underlying context. This effort would necessitate the creation of a new dataset that annotates the depth of each logical relation for every triple consisting of a question, an answer, and a set of reference sentences.

REFERENCES

AnswerDotAI (2024). Ragatouille. Accessed: 2024-09-06.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Carbonell, J. and Goldstein, J. (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336.

Claude AI (2024). Claude. Accessed: 2024-09-06.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Deepset (2024). Haystack - deepset. Accessed: 2024-09-06.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

D.H., C. (2024). Rag dataset 12000.

EmbedChain (2024). Embedchain. Accessed: 2024-09-06.

Google (2024). Gemini - google. Accessed: 2024-09-06.

LangChain (2024). Langchain official website.

Lewis, D. D. and Gale, W. A. (1994). A study of f-measure in information retrieval. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, pages 187–199. Citeseer.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., et al. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

LlamaIndex (2024). Retrieval-augmented generation (rag) - llamaindex. Accessed: 2024-09-06.

Meta (2024). Llama - meta. Accessed: 2024-09-06.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389.

Sammut, C. and Webb, G. I. (2011). *TF–IDF*. Springer.

Zhang, H., Zhou, Y., and Wang, J. (2021). Contextual networks and unsupervised ranking of sentences. In *Proceedings of the 33rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2021)*.

Zhou, Y. and Wang, J. (2023). Content significance distributions of sub-text blocks in articles and its application to article-organization assessment. In *Proceedings of the 15th Knowledge Discovery and Information Retrieval (KDIR 2023)*.

Prediction of Response to Intra-Articular Injections of Hyaluronic Acid for Knee Osteoarthritis

Eva K. Lee[1,2,3][a], Fan Yuan[2], Barton J. Mann[4] and Marlene DeMaio[4,5]

[1]Center for Operations Research in Medicine and Healthcare, The Data and Analytics Innovation Institute, Atlanta, U.S.A.
[2]Georgia Institute of Technology, Atlanta, U.S.A.
[3]AccuHealth Technologies, Atlanta, Georgia, U.S.A.
[4]The American Orthopedic Society for Sports Medicine, Chicago, U.S.A.
[5]Medical Corps, United States Navy, U.S.A.
evalee-gatech@pm.me

Keywords: Knee Osteoarthritis, Injections of Hyaluronic Acid, Machine Learning for Evidence-Based Practice, Branch-and-Bound, Particle Swarm Optimization.

Abstract: Osteoarthritis (OA) is a degenerative joint disease, with the knee the most frequently affected joint. Fifty percent of knee OA patients eventually undergo surgical procedures such as knee replacement to address pain and functional limitations. A significant number of these surgeries may be unnecessary, with intra-articular injections of hyaluronic acid (HA) serving as a non-invasive, cost-effective alternative. Although research studies have clearly demonstrated that HA improves knee function, the efficacy of this treatment remains controversial. Many physicians have observed that effects depend on several patient characteristics such as age, weight, gender, severity of the OA, and technical issues such as injection site and placement. In this study, a multi-stage, multi-group machine learning model is utilized to uncover discriminatory features that can predict the response status of knee OA patients to different types of HA treatment. The algorithm can identify certain subgroups of knee OA patients who respond well to HA therapy. The baseline results, based on factors such as patients' weight, smoking status and frequency, identifies the patients most suitable for HA injection. The model can achieve more than 89% blind prediction accuracy. The data derived from this study allows physicians to administer HA products more selectively, resulting in a higher therapy success rate. Information on the predicted responses could also be shared with patients beforehand to incorporate their values and preferences into treatment selection. The model's decision support tools also allow physicians to quickly determine whether a patient is exhibiting at least the expected treatment response, and if not, to potentially take corrective action. To the best of our knowledge, this work represents the first machine learning approach that predicts patient responses to HA injections for knee osteoarthritis. The model is generalizable and can be used to predict patient responses to other treatments and conditions.

1 INTRODUCTION

Osteoarthritis (OA) is a degenerative joint disease that can affect the many tissues of the joint. It is one of the most prevalent and costly chronic medical conditions. affecting more than 32.5 million adults in the United States (United States Bone and Joint Initiative 2018). During 2019–2021, 21.2% of U.S. adults (53.2 million) reported an arthritis diagnosis. (Elgaddal, et al., 2022; Fallon, et. al., 2023) and by 2040, it is projected to increase to 78.4 million Americans.

Arthritis increasingly is reported as the main cause of disability among U.S. adults (Theis, K.A. et al., 2019). Annual direct medical care expenditures for osteoarthritis in the U.S. is estimated to exceed $495.5 billion (United States Bone and Joint Initiative, 2019; Lo, et al., 2020). Worldwide, about 528 million people were living with osteoarthritis in 2019 (WHO 2023, GBD 2019). It is estimated that those with OA pain lost 31% of productive time at work due to presenteeism and 8% due to absenteeism, compared to 16% and 4%, respectively, for those who did not report OA pain (Leifer et al., 2022).

[a] https://orcid.org/0000-0003-0415-4640

Lee, E., Yuan, F., Mann, B. and DeMaio, M.
Prediction of Response to Intra-Articular Injections of Hyaluronic Acid for Knee Osteoarthritis.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 497-508
ISBN: 978-989-758-716-0; ISSN: 2184-3228

There is no known cure for OA. Instead, treatments aim to reduce pain, maintain or improve joint mobility, and limit functional impairment. Treatments are usually non-operative, such as physical therapy, rest, modification of daily activities, analgesics, and anti-inflammatory medication. For individuals who desire or require a high level of physical activity, rest and activity reduction are not viable treatment options. Oral non-steroidal anti-inflammatory drugs (NSAIDs) are often recommended, although frequent and serious adverse effects of NSAIDs have been reported (Zhang et al., 2010, Salis and Sainsbury, 2024). Over the past 25 years, intra-articular injection of hyaluronic acid (and similar hyaluronan preparations) has emerged as an additional tool for managing the symptoms of OA for patients who fail to respond to other conservative treatments. However, controversies exist regarding its safety and efficacy, the number of injections and courses, type of preparation, duration of its effects, and combining it with other drugs or molecules (Chavda et al., 2022). Other factors include patient characteristics such as age, weight, gender, and severity of the OA.

Knee OA happens when the cartilage in the knee joint breaks down, enabling the bones to rub together. The friction makes the knees hurt, become stiff, and sometimes swell. Knee OA is a leading cause of arthritis disability (Cui et al., 2020). Of significance for sport medicine, heavy physical activity, participation in high intensity contact sports, participation in certain elite level sports, and knee injury have all been linked to the development of knee OA (Chan, et al., 2020; Driban, et al., 2017; Lohmander, et al., 2007; McAlindon et al., 1999; Sharma, 2001; Spector et al., 1996; Turner, et al., 2000). Although it cannot be cured, treatments are available to slow its progression and ease the symptoms. Knee OA alone results in the loss of an average of 13 days of work per year (versus 3 days for those without Knee OA (Ayis & Dieppe, 2009).

Knee osteoarthritis affects more than 14 million Americans, and its symptoms often lead to physical inabilities, disabilities, and all sorts of inconveniences for patients. It is estimated that knee osteoarthritis is associated with approximately $27 billion in total healthcare costs every year, with about 800,000 knee surgeries performed annually. Specifically, 99% of these knee replacements are done to address pain and functional limitations (Barbour et al., 2017). In a multicenter longitudinal cohort study, it was reported that about one-third of knee replacements may be unnecessary (Riddle et al., 2014).

The management of knee pain depends on the diagnosis, inciting activity, underlying medical conditions, body mass, and chronicity. In general, non-operative management is the mainstay of initial treatment and includes rehabilitation, activity modification, weight loss when indicated, shoe orthoses, local modalities, and medication. The oral medication often prescribed is an analgesic, usually with anti-inflammatory properties. Supplements, such as chondroitin sulfate and glucosamine, have been shown to have a role. Since 1997, the regimen has expanded to include viscosupplementation. These agents are preparations of hyaluronic acid or their derivatives (HA) which are sterilely injected into the knee. Although research studies have clearly demonstrated that HA improves knee function, the efficacy of this treatment remains controversial. Many physicians have observed that effects seem to depend on several patient characteristics, such as age, weight, gender, severity of the OA and technical issues such as injection site and placement (Mora et al., 2018).

This study aims to answer an important question: whether different types of patients may respond differently to HA treatment. Is it possible to identify certain subgroups of knee OA patients who respond well (or those who don't) to HA therapy? Further, we question whether it is possible prior to treatment to predict a patient's response to HA injections based on patient and treatment characteristics. Physicians could then make empirically informed decisions about whether to treat a particular patient with HA and perhaps which type of HA preparation is most likely to produce the best treatment response for that individual patient.

The goal of this study is to evaluate which patient population, or patient characteristics, would benefit most from HA injection. Since at least 18% of out-patient visits to military treatment facilities by active-duty personnel are attributed to painful knee disorders, our study focuses on these patients. The study uses a prospective, double-blinded clinical trial. A multi-stage, multi-group machine learning model (Lee et al., 2016b; Lee, 2017; Lee & Egan, 2022; Lee et al., 2021, 2023a, 2023b) described in Section 2.3 is used to uncover discriminatory patterns that can predict suitability of treatment and outcomes. The resulting predictive rule can be implemented as part of a clinical practice guideline for evidence-based intervention. The model enables physicians to administer HA products more selectively and effectively to the targeted population to maximize cost effectiveness and the percentage of patients who experience a successful HA injection.

2 METHODS AND STUDY DESIGN

2.1 Patient Cohort, Treatment, and Outcome Measures

2.1.1 Patient Data

Three group of patients (active-duty military personnel, military retirees, and their families) through the Department of Orthopaedics at the Naval Medical Center Portsmouth were included. The cohort includes those between 18 and 65 who sought treatment for symptomatic osteoarthritis of the knee. All patients were evaluated by a board-certified orthopaedic surgeon. Each patient has had radiographic evidence of knee OA with a minimum Kellgren-Lawrence score of 1, has experienced symptoms for more than three months, has failed a minimum of three months of non-operative treatment, including, but not limited to, analgesic and anti-inflammatory medication, cortisone injection, physical therapy, bracing, and/or heel wedge. The cohort excludes patients with precautions or contraindications for viscosupplementation, those who had a cortisone injection within the past three months, those who had prior HA injections at any point, those with a history of deep knee infection, those currently experiencing peripheral neuropathy, chondrocalcinosis, or knee ligament instability, and those who were candidates for knee surgery.

Patients were randomly assigned to receive either Hylan G-F 20 (Synvisc®) [Sanofi Biosurgery, Cambridge, MA, USA], a high molecular weight (MW = 6000 kDa) cross-linked HA product derived from an avian source, or EUFLEXXA® [bioengineered 1% sodium hyaluronate (IA-BioHA); Ferring Pharmaceuticals, Inc., Parsippany, NJ], a medium weight (MW = 2400 - 2600 kDa) HA product derived from bacterial fermentation.

Treatment allocations were randomly assigned by the study pharmacist using the RANDBETWEEN(0,1) function in Microsoft Excel. Physicians, physicians performing the injections, patients, and research personnel were blinded to treatment assignment. To maintain blinding, the pharmacy removed the original manufacturer's label prior to dispensing and relabelled with the protocol title, subject identifier and expiration date. The two HA products had the same volume and color, so there was no ability to discern one from the other at the time of injection.

During a baseline evaluation before the first injection, the following data were collected:

- patient demographic data: age, sex, height, weight, BMI (as calculated from height and weight), and smoking history.
- the Western Ontario and McMaster Universities Osteoarthritis Index (WOMAC; Bellamy, 2002) as a measure of knee OA symptoms and functioning.
- the RAND-36 (Hays et al., 1993) as a measure of general health status.
- the MARX Knee Activity Rating Scale (Marx et al., 2001) to assess activity level (running, deceleration, cutting (changing directions while running) and pivoting.
- patient-rated health conditions (a) using a comorbidity questionnaire (Sangha et al., 2003) and (b) quality of life as measured by the EuroQOL EQ-5D (Brooks, 1996).
- a patient-completed Arthritis Self-Efficacy Scale (Lorig et al., 1989), an eight-item instrument that assesses patient's perceived ability to manage arthritis symptoms.

Specific patient treatment expectations (e.g., “Improve ability to go up and down stairs”) and the importance of these expectations were evaluated with the scale developed by Mancuso (Mancuso et al., 2001). Patients were also asked to rate their global expectation for their response to the HA injections on a seven-point scale ranging from “No improvement. I don’t have much hope that this treatment will help my symptoms at all” to “Excellent improvement. I expect complete or nearly complete relief from knee symptoms.” Patients with bilateral OA were instructed to rate only the knee they perceived to be more severe in terms of pain and functional impairment on all instruments and to rate the same knee at baseline and follow-ups.

Prior to the first injection, a physician assessed quadriceps atrophy, presence of antalgic gait, knee effusion, pain on palpation of the knee, range of motion and alignment, and use of medication. Patients also received four baseline radiographs. These included (a) a standing anteroposterior (AP) of the knee weight-bearing view; (b) weight-bearing flexed view 400 posterior-anterior (PA) Rosenberg view; (c) a lateral x-ray at 300; and (d) a Merchant view. Digitized radiographs were evaluated for osteoarthritis severity and for alignment by a board-certified musculoskeletal radiologist and an orthopaedic surgeon blinded to assigned treatment or other patient characteristics. OA severity was rated using the Kellgren-Lawrence Grading System which incorporates joint space narrowing, osteophyte formation, sclerosis and bony deformation observed on x-rays. Scores range from 0 (no radiographic features of OA) to 4 (large osteophytes, marked joint space narrowing, severe sclerosis, and definite bony deformity). Alignment was determined by measuring the following angles from x-rays: (a) condylar-hip angle of the femoral condylar tangent with respect to the mechanical axis of the femur expressed as degrees of deviation from 90°, negative for varus and positive for valgus; (b) plateau-ankle angle between the tibial margin tangent and the mechanical axis of the tibia expressed as degrees of deviation from 90°, negative for varus and positive for valgus; (c) condylar-plateau angle between the femoral and tibial joint surface tangents; and (d) hip-knee-ankle angle between a line drawn from the center of the femoral head to the midpoint of the tibial eminential spine and another line from this midpoint to the center of the talus surface of the ankle joint. The medial angle between the lines is the HKA angle (varus < 180°).

2.1.2 HA Treatments

Patients received injections every seven days for a total of three injections. Physicians received specific instructions to standardize injection technique. All injections were performed using an anteromedial approach with a 21-gauge 1½” needle. Physicians aspirated the knee joint prior to injection of the HA product to ensure needle placement. Patients were asked to flex and extend their knee a few times following injection to maximize dispersal into the joint. Patients were provided with written post injection and standardized physical therapy instructions. Patients were allowed full weight bearing and full range of motion (active and passive) after injections but were advised to avoid strenuous activity (such as jogging, tennis, etc.) or prolonged weight bearing for the first 48 hours after injection. Patients were also instructed to use ice 30 minutes on and 30 minutes off for 48 hours and take up to 4 gram of acetaminophen per day as need for knee pain, but not to take any 24 hours prior to each visit.

Patients were not offered a second course of HA treatment within the first six months following the final injection. Following the standard clinical practice, those who received a second series of injections after the first six months were not considered treatment failures. Patients who had surgery on the target knee to relieve arthritis symptoms within the first six months following the last HA injection were considered treatment failures.

The protocol was approved by the Institutional Review Board at the data collection site and was registered with ClinicalTrials.gov (identifier: NCT01557868). A physician at the site served as the medical monitor and an independent data and safety board monitored the study.

2.1.3 Primary and Secondary Outcomes

The primary outcome was treatment responder status defined a priori by improvement in the Western Ontario and McMaster Universities Osteoarthritis Index (WOMAC) Pain Scale (Hochberg et al., 1997; Riddle & Perera, 2020) between baseline and 3-month assessments. The WOMAC Pain Scale is comprised of 5 items and the response format used in this study was the 5-point rating scale. Scores were calculated to range from 0 (worst) to 100 (best). The reliability, validity and responsiveness of the WOMAC Pain Scale have been supported in numerous studies (Bellamy, et al., 2011; Burgers, et al. 2015) and the WOMAC is one of the most widely used outcome instruments in arthritis research. Patients whose pain scores decreased by 20% or more compared with their baseline scores were classified as treatment responders and those whose scores did not meet this criterion were classified as non-responders.

2.2 Machine Learning Predictive Analysis

We apply a multi-stage machine learning approach to analyze how different types of patients may respond differently to HA treatment. The system will uncover discriminatory features in the HA data that will reveal patient and treatment characteristics that predict optimal response to intra-articular injections of hyaluronic acid for knee osteoarthritis. The model determines which patient variables lead to the best outcomes of HA.

Detail of the multi-stage multi-group discriminant analysis via mixed-integer program (DAMIP) model and computational framework is reported in Lee et al. (Lee, 2017; Lee & Egan, 2022; Lee, Wang, et al., 2016; Lee et al., 2021, 2023a, 2023b). Briefly we include the DAMIP formulation below.

Let u_{hgi} represent the binary variable that indicates whether observation i in group g is classified to group h, $h \in \{0\} \cup \mathcal{G}$. Thus, $u_{ggi} = 1$ denotes a correct classification for observation i in group g. The multi-group model with a reserved judgement region is formulated as:

$$\max \sum_{g \in \mathcal{G}} \sum_{j \in \mathcal{O}_g} u_{ggj} \qquad \textbf{(DAMIP)}$$

subject to

$$L_{hgj} = \pi_g f_g(\boldsymbol{x}_j) - \textstyle\sum_{h \in \mathcal{G}, h \neq g} \lambda_{hg} f_h(\boldsymbol{x}_j), \forall\, h, g \in \mathcal{G}, j \in \mathcal{O}_g \quad (1)$$

$$y_{gj} - L_{hgj} \leq M(1 - u_{hgj}), \qquad \forall\, h, g \in \mathcal{G},\ j \in \mathcal{O}_g \quad (2)$$

$$y_{gj} \leq M(1 - u_{0gj}), \qquad \forall\, g \in \mathcal{G},\ j \in \mathcal{O}_g \tag{3}$$
$$y_{gj} - L_{hgj} \geq \varepsilon(1 - u_{hgj}), \qquad \forall\, h, g \in \mathcal{G},\ j \in \mathcal{O}_g \tag{4}$$
$$y_{gj} \geq \varepsilon\, u_{hgj}, \qquad \forall\, h, g \in \mathcal{G},\ j \in \mathcal{O}_g \tag{5}$$
$$\textstyle\sum_{h \in \{0\} \cup \mathcal{G}} u_{hgj} = 1, \qquad \forall\, g \in \mathcal{G},\ j \in \mathcal{O}_g \tag{6}$$
$$\textstyle\sum_{j \in \mathcal{O}_g} u_{hgj} \leq \lfloor \alpha_{hg} n_g \rfloor, \qquad \forall\, h, g \in \mathcal{G}, g \neq h \tag{7}$$
$$u_{hgj} \in \{0,1\} \qquad \forall\, h \in \{0\} \cup \mathcal{G}, g \in \mathcal{G}, j \in \mathcal{O}_g \tag{8}$$
$$y_{gj} \geq 0, \qquad \forall\, h, g \in \mathcal{G},\ j \in \mathcal{O}_g \tag{9}$$
$$\lambda_{hg} \geq 0 \qquad \forall\, h, g \in \mathcal{G}, g \neq h \tag{10}$$

Here, π_g is the prior probability of group g and $f_g(\boldsymbol{x})$ is the conditional probability density function of group g, $g \in \mathcal{G}$ for the data point $\boldsymbol{x} \in \mathbb{R}^m$. $\mathcal{O}_g$ denote the set of observations in group g, and n_g denote the number of observations in group $g \in \mathcal{G}$. $\alpha_{hg} \in (0,\ 1)$, h, $g \in \mathcal{G}$, $h \neq g$ represents the predetermined limit on the inter-group misclassification rate where the observations of group g are misclassified to group h. The group assignment decisions of observations that are classified into a reserved judgment region are denoted by group $g = 0$.

Constraints (1) define the loss functions; constraints (2)-(6) guarantee an observation is uniquely assigned to the group with the maximum value of $L_g(\boldsymbol{x})$ among all group, and constraints (7) set the misclassification limits. With the reserved judgment region in place, the mathematical system ensures that a solution that satisfies the pre-set misclassification rate always exists.

Theorem 1. Given prior probabilities π_g and conditional group density functions $f_g(\boldsymbol{x})$, allocation according to modified posterior probabilities defined by the solution to (DAMIP) is a *universally strongly consistent* method for classification.

Theorem 2. The DAMIP optimization problem is $\mathcal{NP} - Complete$ when the number of groups is greater than 2. The theoretical result holds for DAMIP variants: (a) maximize the minimum value of correct classification rates among all groups; (b) maximize the minimum difference between correct classification and misclassification; and (c) maximize correct classification while constraining the percentage of reserved judgment for each group.

The multi-stage classification approach utilizes the reserved judgment region in DAMIP to improve the classification performance, especially among highly inseparable data. At each stage, DAMIP partitions the observations into an '*easy–to-classify*' subset that is classified to specific groups, and a '*difficult-to-classify*' subset that is classified to a *reserved judgment region*. The group assignment of the difficult-to-classify observations are delayed, thus allowing the DAMIP classifier to maintain a low misclassification error. The observations in the reserved judgment region are moved to the next stage where a new feature set is selected and a new DAMIP classifier is developed. In this way, the multi-stage framework constructs a chain of successive classifiers using different subsets of features. The classifier at the ith stage, denoted by f_i, can be represented by a discriminant function $f(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$, which is determined by the feature subset $\boldsymbol{x}_i$, and the decision variables $\boldsymbol{\lambda}_i$ in DAMIP.

At each stage, two models are performed: a single-stage model that solves a DAMIP model without a reserved judgment region and a multi-stage model that solves a DAMIP model with a reserved judgment region. The computational framework selects the better of the two results. The algorithm naturally terminates when there are no observations in the reserved judgment region. To avoid overfitting using too few observations for training, two additional stopping criteria are used to terminate the process: (a) the number of observations is less than a preset minimum value, *n*, and (b) the maximum allowed depth, *d,* is reached. The parameters *n* and *d* are predetermined according to the number of observations and the number of input features in the given data.

Computationally, DAMIP classifier has some distinct characteristics: (a) it is applicable for classification of any number of groups; (b) there is always a feasible solution to the model; (c) the reserved judgement region facilitates successive stage of classification to be performed; (d) DAMIP is able to establish classification rules with good predictive accuracy even when the training set is relatively small; (e) DAMIP classifier can handle imbalanced data; and (f) DAMIP classifier is totally universally consistent.

Figure 1 shows the machine learning framework where features are first selected via an exact branch-and-bound algorithm (BB) and a fast heuristic particle swarm optimization (PSO) (Lee et al., 2023a). The resulting classification rule is subsequently established via the DAMIP classifier. To quantify the accuracy, ten-fold cross validation evaluation is performed. If the results satisfy some pre-set accuracy level, the classification rule is reported. Blind prediction using this rule is then performed. We contrast the BB-PSO/DAMIP results with eight commonly used classifiers: Bernoulli Naïve Bayes, Decision Tree, Gradient Boosting, K-nearest neighbors, Logistic Regression, Neural Network, Random Forest, and Support Vector Machine (SVM).

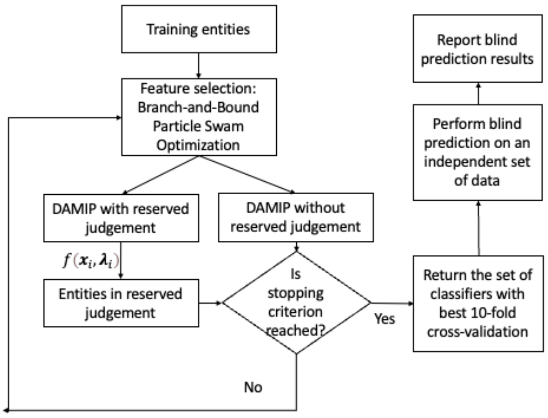

Training entities
Feature selection: Branch-and-Bound Particle Swam Optimization
DAMIP with reserved judgement
DAMIP without reserved judgement
$f(\boldsymbol{x}_i, \boldsymbol{\lambda}_i)$
Entities in reserved judgement
Is stopping criterion reached?
Yes
No
Return the set of classifiers with best 10-fold cross-validation
Perform blind prediction on an independent set of data
Report blind prediction results

Figure 1: Multi-stage machine learning framework for HA predictive analytics.

In 10-fold cross validation, the training set is partitioned into 10 roughly equal subsets. In each run, 9-fold are selected to train and establish the rule, and the remaining 1-fold is then tested, counting how many of them are classified into which group. Through 10 folds procedure (where each fold is being validated exactly once), we obtain an unbiased estimate of the classification accuracy.

Blind prediction is performed on patients that are independent of the training set to gauge the predictive power of the established rule. These patients have never been used in the feature selection and the machine learning analysis. We run each patient in the blind set through the rule, which returns a group status of the patient. The status is then checked against the clinical status to confirm the accuracy.

The classifier response and outcome prediction rules will culminate in a clinical decision algorithm for the use of viscosupplementation in the treatment of knee OA. For example, a physician determines that HA is indicated for a particular patient. The physician would then enter specific variables (those discriminatory features identified by the classifier) into a clinical computer program and a response set would be generated for the potential outcome after using hyaluronic acid injections. The optimal HA agent(s) would be ranked. The physician would then take this information into account as part of the clinical decision process to select the HA agent for the individual patient.

3 RESULTS

3.1 Patient Characteristics

Of the 273 patients assessed for eligibility, 45 did not meet study criteria, 13 eligible patients declined to participate, and 12 eligible patients could not complete study participation due to anticipated deployment or relocation. The other 203 eligible patients were randomized to treatment: 107 assigned to the Synvisc group and 96 to the Euflexxa group. After randomization, 6 patients were non-compliant with the study protocol, 9 received an excluded intervention, 6 were reassigned, 10 were lost to follow-up and 6 missed the follow-up appointment. Consequently, these patients were not included in the analyses, leaving a total of 166 (87 in the Synvisc group and 79 in the Euflexxa group).

Table 1 summarizes the baseline characteristics of the study participants. The Synvisc and the Euflexxa groups did not differ on demographic or anthropometric variables. The groups also did not differ on co-morbid conditions with the exception that a greater proportion of patients in the Euflexxa group reported depression (21% vs. 10%, p = 0.02). The baseline scores from the patient report measures did

Table 1: Baseline Characteristics of the Study Participants.

Characteristic	Synvisc (N = 107)	Euflexxa (N = 96)	Combined Sample (N = 203)
Age – year	46±10	43±10	45±10
Female sex – no. (%)	44 (41)	36 (38)	80 (39)
Body mass index	30±5	29±5	30±5
Race			
Asian	1 (1)	5 (5)	6 (3)
Black/African-American	36 (34)	21 (22)	57 (28)
Hispanic	5 (5)	6 (6)	11 (5)
White	63 (59)	63 (66)	126 (62)
Other	2 (2)	1 (1)	3 (2)
Married – no. (%)	86 (80)	79 (82)	165 (81)
Current smoker – no. (%)	16 (15)	11(12)	27 (13)
Kellgren-Lawrence Score			
Grade I – no. (%)	28 (26)	37 (39)	65 (32)
Grade II – no. (%)	44 (41)	33 (34)	77 (38)
Grade III – no. (%)	29 (27)	18 (19)	47 (23)
Grade IV–no. (%)	6 (6)	8 (8)	14 (7)
WOMAC Pain Scale	59±17	61±19	60±18
SF-36			
Physical functioning	51±23	54±24	53±23
Mental health	79±15	74±18	77±17
Marx Activity Scale	5±5	5±5	5±5
EuroQOL EQ-5D Health Rating	71±16	70±20	70±19
Arthritis Self-Efficacy Scale	6±2	6±2	6±2
Treatment response expectation	5±1	5±1	5±1
Bilateral HA injections – no. (%)	54 (51)	45 (47)	99 (49)

not significantly differ between the two treatment groups either.

3.2 Primary End Points

Of the 166 patients who completed the 3-month assessment, 84 (50.6%) were classified as treatment responders. Within the Synvisc group, 57.5% were responders compared to 43% of the Euflexxa group (p = 0.04). This outcome, as well as those at the 2-week and 6-month follow-ups, is shown in Table 2. Table 3 displays the percentage of patients who were classified as "recovered" based on both statistically reliable improvement in WOMAC Pain Scale scores and a follow-up score that fell within the range of age- and sex-matched patients who reported having no knee problems or any history of knee surgery (see Mann, et al., 2012).

Table 2: Treatment Responders (20% Reduction in WOMAC Pain) by Treatment Group.

Follow-Up	Synvisc	Euflexxa	P Value
2 weeks	56.3%	56.3%	0.55
3 months	57.5%	43.0%	**0.04**
6 months	51.3%	41.5%	0.31

Table 3: Return to Normal on WOMAC Pain Scale by Treatment Group.

Follow-Up	Synvisc	Euflexxa	P Value
2 weeks	36.5%	25.4%	0.20
3 months	38.0%	22.6%	0.06
6 months	33.9%	30.0%	0.31

3.3 Response and Outcome Prediction

We analyze the HA data to uncover patient and treatment factors that predict optimal response to intra-articular injections of hyaluronic acid for knee osteoarthritis. The treatment responder status six months after final injection is measured by 'WOMACP20," Treatment Responder Status Using 20% Reduction in WOMAC Pain Scale. Recovery status is assessed via the KOOS Scale. The machine learning model determines which patient variables lead to the best outcomes of HA. We also perform the prediction for each of the two HA products to gauge their similarities and differences in treatment outcome characteristics.

Table 4 shows the number of patients in the training set and the blind prediction set for predicting reinjection status. In this analysis, for every attribute in which there is missing data, an associated binary attribute is created to capture whether data is missing or not for this field. The number of attributes at three time-points: (a) baseline screening before first injection; (b) prior to second injection (prefix: T0); and (c) six months after final injection (prefix: T5) are 27, 483, and 1215 respectively. Table 5 shows the training set and blind prediction statistics used for predicting treatment responder status and recovery status.

Table 4: Training set and blind prediction set characteristics for predicting reinjection status.

Training set			Blind Prediction Set		
Total	No reinjection	Reinjection	Total	No reinjection	Reinjection
150	111	39	53	40	13

Table 5: Training set and blind prediction set characteristics for predicting treatment responder status and recovery status.

Training set			Blind Prediction Set		
Total	Non-Responder	Responder	Total	Non-Responder	Responder
71	34	37	70	41	29
Synvisc					
40	18	22	36	19	17
Euflexxa					
35	21	14	30	17	13

We summarize herein the best predictive rules for each of the analyses. Table 6 shows the prediction accuracy for no-reinjection versus re-injection using attributes collected up to the three stated time-points.

For the baseline results, factors that appear to be critical includes "Weight," "Currently Smoke Cigarettes," and "Smoking: Number per day." Baseline prediction results are comparable to Pap Smear test accuracy (~70%).

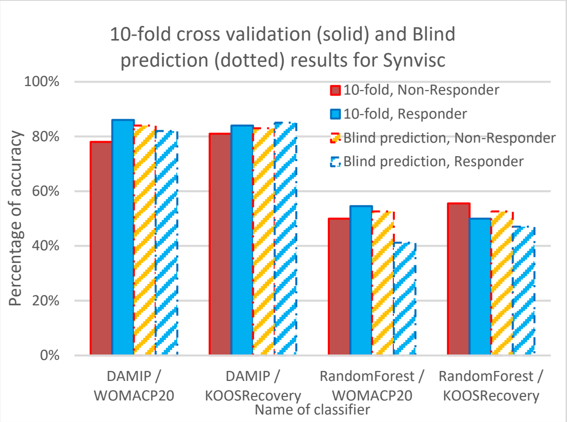
We can observe high accuracy in predicting success for patients using screening and T0 attributes alone (86% blind predictive accuracy). This is very promising for identifying patients early (just after the first injection) who should be targeted for HA intervention (with an expected success outcome). The discriminatory features selected includes the Marx Activity Scale "T0MarxCuttingSymptomFree", "T0MarxCutting", effectiveness of exercise "T0ExerciseEffective", confidence in the injector "T0ConfidenceInjector", and other medications "T0MedicationXEffective."

Including attributes until T5 significantly increases the accuracy for predicting the reinjection group (from 71% to 89%). Early attributes include "T0PhysicalTherapyEffective", "T0MedicationXEffective," and overall health "T0EQRateHealth" continue to appear among the selected features.

Table 6: Best predictive rule for re-injection status when using attributes (a) baseline screening before first injection, (b) prior to second injection, and (c) 6 months after final injection.

Input attributes	**10-fold cross validation**		**blind prediction**	
	No-reinjection	Re-injection	No-reinjection	Re-injection
Baseline screening	**71%**	**71%**	**72%**	71%
Prior to 2nd injection	**89%**	**74%**	**86%**	71%
Input attributes	84%	83%	81%	**89%**

Figure 2 show the 10-fold cross validation and blind prediction accuracies for predicting treatment responder status and recovery status for patients injected with Synvisc and Euflexxa, respectively. For each HA injection, four measurement frameworks are graphed: PSO/DAMIP results for predicting treatment responder and recovery respectively versus the best results from the eight commonly used classifiers, Random Forest. Our PSO/DAMIP framework selected 3-8 discriminatory features whereas Random Forest uses over 40 features with poor results. Although the size of the two groups is rather balanced, the challenge here is due to the highly inseparable data that makes it difficult to classify using traditional approaches. A multi-stage approach allows the partitioning of patients from the same group via different rules (associated with different features).



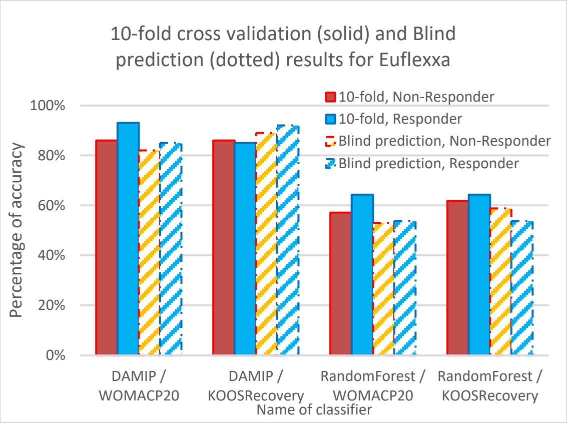


Figure 2: Comparison of the best DAMIP classification rules for predicting treatment responder status and recovery status using Synvisc(top) and Euflexxa (bottom) against the Random Forest approach.

Our study shows that early predictors can be used to determine the group of patients who benefit the most from HA injection. It also allows evidence-based correction to be made during the course of treatment. For example, after T0, the physician can quit treatment based on results from the predictive rule.

4 CONCLUSIONS

In 2019, about 528 million people worldwide were living with osteoarthritis, an increase of 113% since 1990. For 365 million, the knee was the most frequently affected joint. On average, the total cost of knee replacement surgery ranges from $30,000 to $50,000. This includes the cost of the surgery itself, the hospital stays, anesthesia and other associated medical expenses. HA treatment, on the other hand, costs about $900 to $3,000 for a full course (three to five injections administered over several weeks). The range reflects the variations due to the type of HA product and the physician's fees. Although 50% of knee osteoarthritis patients eventually receive surgical procedures, almost one third of these surgeries are unnecessary. Hence intra-articular injections of hyaluronic acid can serve as a non-invasive cost-effective alternative to surgery for knee osteoarthritis.

Unlike surgical options, HA injections do not require incisions or extensive recovery periods. HA is a substance that naturally occurs in the synovial fluid of the joints, which helps lubricate and cushion them. In osteoarthritis, this fluid becomes less effective, leading to pain and reduced mobility. Thus, HA injected directly into the knee joint helps restore the lubricating properties of the synovial fluid and reduce inflammation. By restoring lubrication, HA injections can help improve joint mobility and reduce stiffness. The procedure is relatively low risk, with mild potential side effects, such as temporary swelling or discomfort.

However, the benefits of HA injections are not permanent; they typically last for several months. Repeated injections may be needed for ongoing relief. More importantly, controversies exist regarding its safety and efficacy, the number of injections and courses, type of preparation, duration of its effects, and combining it with other drugs or molecules. Other factors include patient characteristics such as age, weight, gender, and severity of the OA. The study uses a prospective, double-blinded clinical trial. A multi-stage, multi-group DAMIP-based machine learning model is utilized to uncover discriminatory features that can predict the response status of knee OA patients to different types of HA treatment. The algorithm can identify certain subgroups of knee OA patients who respond well (or those who don't) to HA therapy. The study's baseline result, including factors such as patients' weight, smoking status and smoking frequency, gives physicians insight for patient treatment recommendations by identifying those most suitable for HA injection.

To the best of our knowledge, this work presents the first machine learning approach that predicts patient responses to HA injections for knee osteoarthritis. Another uniqueness of this study is that this is the first prospective clinical trial designed such that in addition to clinical data, patient self-reporting data is also carefully collected. The latter is challenging since patients often refuse or bypass questionnaires or miss filling in forms. Self-reported answers may be exaggerated; respondents may be too embarrassed to reveal private details; various biases may affect the results, like social desirability bias. However, knee pains, whether patients can move or do certain activities are standard questions used by physicians and are rather routine evaluation for active-duty personnel and athletes, and hence their self-reporting are rather reliable. Further, there has been no study indicating that patients would exaggerate their pain to receive treatment to their knee pain.

Traditional data collection methods, primarily focusing on clinical settings, limits our understanding of drug efficacy and *patient* wellbeing. Patient self-reporting data is crucial for machine learning in healthcare because it provides a unique, subjective perspective on a patient's health experience, including their symptoms, quality of life, and perception of treatment effectiveness, which can be vital for accurate diagnosis, treatment planning, and overall patient care, often not captured by solely objective medical data like lab results or imaging scans. There is growing interest and support for the utility and importance of patient-reported outcome measures (PROMs) (Kingsley & Patel, 2017; Verma, et al., 2021). This is one of the strengths of our study since it includes a broad spectrum of patient wellbeing data.

DAMIP classifier was chosen partly due to earlier DAMIP models have produced good predictive accuracy on blind data for numerous clinical studies where the training patient size is relatively small (e.g., in early cancer detection to uncover genomic signatures that predict CpG islands methylation (Feltus, et al., 2003), vaccine immunogenicity prediction that accelerates vaccine design and target delivery (Lee, Nakaya, et al., 2016a; Nakaya, et al, 2011, 2015; Querec, et al., 2009;) in which DAMIP results were instrumental in the eventual world-wide clinical trial of the Malaria vaccines (Kazmin, et al., 2017; Lee, Nakaya et al., 2016a)). DAMIP has also been used for studies involving very large number patient sets with equally consistent predictive accuracy (Lee, Wang, et al., 2016b). Multi-stage is performed herein to manage the highly inseparable data.

With the established predictive rule, prior to treatment physicians can predict a patient's response to HA injections based on patient and treatment characteristics. Physicians can then make empirically informed decisions about whether to treat a particular patient with HA and perhaps which type of HA preparation is most likely to produce the best treatment response for that individual patient.

Predicting treatment response based on clinically measured variables and patient-centered well-being data will empower physicians with an evidence-based decision-making tool to administer the most cost-effective intervention for the patients.

The study's follow-up period is focused on six months after the final injection. Since knee osteoarthritis is incurable, treatment for patients includes rehabilitation, activity modification, weight loss when indicated, shoe orthoses, local modalities, and medication. For more severe cases, either HA injections or knee surgery is selected. And HA injections are typically given as a series of 3-5 injections, spaced one week apart, with *repeat courses* usually needed every six months, depending on the individual's pain relief duration and the severity of their arthritis; most people experience pain relief for several months after a full course of injections.

The data and model derived from this study allows physicians to administer HA products more selectively and effectively, which will increase the percentage of patients who experience a successful HA therapy. Information about predicted responses could easily be shared with patients to incorporate their values and preferences into treatment selection. Specifically, the classification rule can be implemented within the electronic health record system as an Application Programming Interface (API). In addition, this decision support tool would allow physicians to quickly determine whether a patient is exhibiting at least an expected treatment response and if not, to potentially take corrective action. Of note, this model can also be used to predict patient responses to other forms of treatment and conditions.

There is a clear demand for evidence-based medical decision-making in addition to expert opinion, clinical experience and case reports. Additionally, there is an increased demand for clinical studies of prospective, rather than retrospective, treatment assessment options. While each of these study types has a role, the value of evidence-based, single studies or meta-analyses of published reports is that clinical criterion or criteria are analyzed globally with respect to outcome. Quantified variables that are uncovered by predictive models are evaluated and analyzed and can serve as important decision variables to help physicians select the best course of treatment for patients. Evidence-based decision-making increases outcome success. Trends, impressions and opinions are minimized and objective, evidence-based, outcome-driven targeted delivery is maximized.

ACKNOWLEDGEMENTS

A portion of the results from this project (the machine learning advances) received the first runner-up prize at the Caterpillar and INFORMS Innovative Applications in Analytics Award. This work is partially supported by grants from the National Science Foundation (IIP-1361532), and the American Orthopedic Society for Sports Medicine. Findings and conclusions in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation and or the American Orthopedic Society for Sports Medicine. The authors extend their deepest respect and gratitude to the late Dr. Barton J. Mann PhD, with whom we collaborated on the knee osteoarthritis research, and to Dr. Captain Marlene DeMaio (retired) from Medical Corps, United States Navy, for her clinical guidance and collaboration on the project. We thank the anonymous reviewers for their insightful comments.

REFERENCES

Ayis S, Dieppe P. (2009). The natural history of disability and its determinants in adults with lower limb musculoskeletal pain. J Rheumatol. 36:583–91. doi: 10.3899/jrheum.080455

Barbour, K. E., Helmick, C. G., Boring, M., & Brady, T. J. (2017). Vital Signs: Prevalence of Doctor-Diagnosed Arthritis and Arthritis-Attributable Activity Limitation — United States, 2013–2015. MMWR. Morbidity and Mortality Weekly Report, 66(9).

Bellamy, N. (2002). WOMAC osteoarthritis index user guide. Version V. Brisbane, Australia.

Bellamy, N., Wilson, C., Hendrikz, J., Whitehouse, S. L., Patel, B., Dennison, S., & Davis, T. (2011). Osteoarthritis Index delivered by mobile phone (m-WOMAC) is valid, reliable, and responsive. Journal of Clinical Epidemiology, 64(2).

Brooks, R., & De Charro, F. (1996). EuroQol: The current state of play. Health Policy, 37(1). https://doi.org/10.1016/0168-8510(96)00822-6

Burgers, P. T. P. W., Poolman, R. W., Van Bakel, T. M. J., Tuinebreijer, W. E., Zielinski, S. M., Bhandari, M., Patka, P., & Van Lieshout, E. M. M. (2015). Reliability, validity, and responsiveness of the Western Ontario and McMaster Universities osteoarthritis index for elderly patients with a femoral neck fracture. Journal of Bone and Joint Surgery - American Volume, 97(9).

Chan, L. L. Y., Wong, A. Y. L., & Wang, M. H. (2020). Associations between sport participation and knee symptoms: A cross-sectional study involving 3053 undergraduate students. BMC Sports Science, Medicine and Rehabilitation, 12(1).

Chavda, S., Rabbani, S. A., & Wadhwa, T. (2022). Role and Effectiveness of Intra-articular Injection of Hyaluronic Acid in the Treatment of Knee Osteoarthritis: A Systematic Review. Cureus. https://doi.org/10.7759/cureus.24503

Cui, A., Li, H., Wang, D., Zhong, J., Chen, Y., & Lu, H. (2020). Global, regional prevalence, incidence and risk factors of knee osteoarthritis in population-based studies. EClinicalMedicine, 29–30.

Driban, J. B., Hootman, J. M., Sitler, M. R., Harris, K. P., & Cattano, N. M. (2017). Is participation in certain sports associated with knee osteoarthritis? A systematic review. In Journal of Athletic Training (Vol. 52, Issue 6). https://doi.org/10.4085/1062-6050-50.2.08

Elgaddal N, Kramarow EA, Weeks JD, Reuben C. Arthritis in adults age 18 and older: United States, 2022. NCHS Data Brief, no 497. Hyattsville, MD: National Center for Health Statistics. 2024.

Fallon, E. A., Boring, M. A., Foster, A. L., Stowe, E. W., Lites, T. D., Odom, E. L., & Seth, P. (2023). Prevalence of Diagnosed Arthritis — United States, 2019–2021.

MMWR. Morbidity and Mortality Weekly Report, 72(41). https://doi.org/10.15585/mmwr.mm7241a1

Feltus, F. A., Lee, E. K., Costello, J. F., Plass, C., & Vertino, P. M. (2003). Predicting aberrant CpG island methylation. Proceedings of the National Academy of Sciences of the United States of America, 100(21).

GBD 2019: Global burden of 369 diseases and injuries in 204 countries and territories, 1990–2019: a systematic analysis for the Global Burden of Disease Study 2019.

Hays, R. D., Sherbourne, C. D., & Mazel, R. M. (1993). The rand 36-item health survey 1.0. Health Economics, 2(3). https://doi.org/10.1002/hec.4730020305

Hochberg, M. C., Altman, R. D., Brandt, K. D., & Moskowitz, R. W. (1997). Design and conduct of clinical trials in osteoarthritis: Preliminary recommendations from a Task Force of the Osteoarthritis Research Society. Journal of Rheumatology, 24(4).

Kazmin, D., Nakaya, H. I., Lee, E. K., Johnson, M. J., Van Der Most, R., Van Den Berg, R. A., Ballou, W. R., Jongert, E., Wille-Reece, U., Ockenhouse, C., Aderem, A., Zak, D. E., Sadoff, J., Hendriks, J., Wrammert, J., Ahmed, R., & Pulendran, B. (2017). Systems analysis of protective immune responses to RTS,S malaria vaccination in humans. Proceedings of the National Academy of Sciences of the United States of America, 114(9). https://doi.org/10.1073/pnas.1621489114

Kingsley, C., & Patel, S. (2017). Patient-reported outcome measures and patient-reported experience measures. BJA Education, 17(4).

Lee, E. K. (2017). Innovation in big data analytics: Applications of mathematical programming in medicine and healthcare. Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017, 2018-January.

Lee, E. K., & Egan, B. (2022). A Multi-stage Multi-group Classification Model: Applications to Knowledge Discovery for Evidence-based Patient-centered Care. International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K - Proceedings, 1.

Lee, E. K., Li, Z., Wang, Y., Hagen, M. S., Davis, R., & Egan, B. M. (2021). Multi-Site Best Practice Discovery: From Free Text to Standardized Concepts to Clinical Decisions. 2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2766–2773. https://doi.org/10.1109/BIBM52615.2021.9669414

Lee, E. K., Nakaya, H. I., Yuan, F., Querec, T. D., Burel, G., Pietz, F. H., Benecke, B. A., & Pulendran, B. (2016a). Machine learning for predicting vaccine immunogenicity. Interfaces, 46(5).

Lee, E. K., Wang, Y., Hagen, M. S., Wei, X., Davis, R. A., & Egan, B. M. (2016b). Machine learning: Multi-site evidence-based best practice discovery. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10122 LNCS.

Lee, E. K., Wang, Y., He, Y., & Egan, B. M. (2019). An efficient, robust, and customizable information extraction and pre-processing pipeline for electronic health records. IC3K 2019 - Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 1. https://doi.org/10.5220/0008071303100321

Lee, E. K., Yuan, F., Man, B. J., & Egan, B. (2023a). A General-Purpose Multi-stage Multi-group Machine Learning Framework for Knowledge Discovery and Decision Support. Communications in Computer and Information Science, 1842 CCIS. https://doi.org/10.1007/978-3-031-43471-6_4

Lee, E. K., Yuan, F., Mann, B. J., & Egan, B. (2023b). Handling Imbalanced and Poorly Separated Data: a Multi-Stage Multi-Group Machine Learning Approach. Proceedings - 2023 2023 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2023. https://doi.org/10.1109/BIBM58861.2023.10386028

Leifer, V. P., Katz, J. N., & Losina, E. (2022). The burden of OA-health services and economics. Osteoarthritis and Cartilage, 30(1). https://doi.org/10.1016/j.joca.2021.05.007

Lo, J., Chan, L., & Flynn, S. (2021). A Systematic Review of the Incidence, Prevalence, Costs, and Activity and Work Limitations of Amputation, Osteoarthritis, Rheumatoid Arthritis, Back Pain, Multiple Sclerosis, Spinal Cord Injury, Stroke, and Traumatic Brain Injury in the United States: A 2019 Update. In Archives of Physical Medicine and Rehabilitation (Vol. 102, Issue 1). https://doi.org/10.1016/j.apmr.2020.04.001

Lohmander, L. S., Englund, P. M., Dahl, L. L., & Roos, E. M. (2007). The long-term consequence of anterior cruciate ligament and meniscus injuries: Osteoarthritis. In American Journal of Sports Medicine (Vol. 35, Issue 10). https://doi.org/10.1177/0363546507307396

Long, H., Liu, Q., Yin, H., Wang, K., Diao, N., Zhang, Y., Lin, J., & Guo, A. (2022). Prevalence Trends of Site-Specific Osteoarthritis From 1990 to 2019: Findings From the Global Burden of Disease Study 2019. Arthritis and Rheumatology, 74(7). https://doi.org/10.1002/art.42089

Lorig, K., Chastain, R. L., Ung, E., Shoor, S., & Holman, H. R. (1989). Development and evaluation of a scale to measure perceived self-efficacy in people with arthritis. Arthritis & Rheumatism, 32(1). https://doi.org/10.1002/anr.1780320107

Mancuso, C. A., Sculco, T. P., Wickiewicz, T. L., Jones, E. C., Robbins, L., Warren, R. F., & Williams-Russo, P. (2001). Patients' expectations of knee surgery. Journal of Bone and Joint Surgery, 83(7). https://doi.org/10.2106/00004623-200107000-00005

Mann, B. J., Gosens, T., & Lyman, S. (2012). Quantifying clinically significant change: A brief review of methods and presentation of a hybrid approach. In American Journal of Sports Medicine (Vol. 40, Issue 10). https://doi.org/10.1177/0363546512457346

Marx, R. G., Stump, T. J., Jones, E. C., Wickiewicz, T. L., & Warren, R. F. (2001). Development and evaluation of an activity rating scale for disorders of the knee.

American Journal of Sports Medicine, 29(2). https://doi.org/10.1177/03635465010290021601

McAlindon, T. E., Wilson, P. W. F., Aliabadi, P., Weissman, B., & Felson, D. T. (1999). Level of physical activity and the risk of radiographic and symptomatic knee osteoarthritis in the elderly: The Framingham study. American Journal of Medicine, 106(2). https://doi.org/10.1016/S0002-9343(98)00413-6

Mora, J. C., Przkora, R., & Cruz-Almeida, Y. (2018). Knee osteoarthritis: Pathophysiology and current treatment modalities. In Journal of Pain Research (Vol. 11). https://doi.org/10.2147/JPR.S154002

Nakaya, H. I., Hagan, T., Duraisingham, S. S., Lee, E. K., Kwissa, M., Rouphael, N., Frasca, D., Gersten, M., Mehta, A. K., Gaujoux, R., Li, G. M., Gupta, S., Ahmed, R., Mulligan, M. J., Shen-Orr, S., Blomberg, B. B., Subramaniam, S., & Pulendran, B. (2015). Systems Analysis of Immunity to Influenza Vaccination across Multiple Years and in Diverse Populations Reveals Shared Molecular Signatures. Immunity, 43(6). https://doi.org/10.1016/j.immuni.2015.11.012

Nakaya, H. I., Wrammert, J., Lee, E. K., Racioppi, L., Marie-Kunze, S., Haining, W. N., Means, A. R., Kasturi, S. P., Khan, N., Li, G. M., McCausland, M., Kanchan, V., Kokko, K. E., Li, S., Elbein, R., Mehta, A. K., Aderem, A., Subbarao, K., Ahmed, R., & Pulendran, B. (2011). Systems biology of vaccination for seasonal influenza in humans. Nature Immunology, 12(8). https://doi.org/10.1038/ni.2067

Querec, T. D., Akondy, R. S., Lee, E. K., Cao, W., Nakaya, H. I., Teuwen, D., Pirani, A., Gernert, K., Deng, J., Marzolf, B., Kennedy, K., Wu, H., Bennouna, S., Oluoch, H., Miller, J., Vencio, R. Z., Mulligan, M., Aderem, A., Ahmed, R., & Pulendran, B. (2009). Systems biology approach predicts immunogenicity of the yellow fever vaccine in humans. Nature Immunology, 10(1). https://doi.org/10.1038/ni.1688

Riddle, D. L., Jiranek, W. A., & Hayes, C. W. (2014). Use of a validated algorithm to judge the appropriateness of total knee arthroplasty in the United States: A multicenter longitudinal cohort study. Arthritis and Rheumatology, 66(8). https://doi.org/10.1002/art.38685

Riddle, D. L., & Perera, R. A. (2020). The WOMAC pain scale and crosstalk from co-occurring pain sites in people with knee pain: A causal modeling study. Physical Therapy, 100(10). https://doi.org/10.1093/ptj/pzaa098

Salis, Z., & Sainsbury, A. (2024). Association of long-term use of non-steroidal anti-inflammatory drugs with knee osteoarthritis: a prospective multi-cohort study over 4-to-5 years. Scientific Reports, 14(1). https://doi.org/10.1038/s41598-024-56665-3

Sangha, O., Stucki, G., Liang, M. H., Fossel, A. H., & Katz, J. N. (2003). The Self-Administered Comorbidity Questionnaire: A new method to assess comorbidity for clinical and health services research. Arthritis Care and Research, 49(2). https://doi.org/10.1002/art.10993

Saxon, L., Finch, C., & Bass, S. (1999). Sports participation, sports injuries and osteoarthritis implications for prevention. In Sports Medicine (Vol. 28, Issue 2). https://doi.org/10.2165/00007256-199928020-00005

Sharma, L., Song, J., Felson, D. T., Cahue, S., Shamiyeh, E., & Dunlop, D. D. (2001). The role of knee alignment in disease progression and functional decline in knee osteoarthritis. JAMA, 286(2). https://doi.org/10.1001/jama.286.2.188

Spector, T. D., Harris, P. A., Hart, D. J., Cicuttini, F. M., Nandra, D., Etherington, J., Wolman, R. L., & Doyle, D. V. (1996). Risk of osteoarthritis associated with long-term weight-bearing sports: A radiologic survey of the hips and knees in female ex-athletes and population controls. Arthritis and Rheumatism, 39(6). https://doi.org/10.1002/art.1780390616

Theis, K. A., Steinweg, A., Helmick, C. G., Courtney-Long, E., Bolen, J. A., & Lee, R. (2019). Which one? What kind? How many? Types, causes, and prevalence of disability among U.S. adults. Disability and Health Journal, 12(3). https://doi.org/10.1016/j.dhjo.2019.03.001

Turner, A. P., Barlow, J. H., & Heathcote-Elliot, C. (2000). Long term health impact of playing professional football in the United Kingdom. British Journal of Sports Medicine, 34(5). https://doi.org/10.1136/bjsm.34.5.332

United States Bone and Joint Initiative: The Burden of Musculoskeletal Diseases in the United States (BMUS) Fourth Edition. Forthcoming; 4th:http://www.boneandjointburden.org. Accessed December 20, 2023.

United States Bone and Joint Initiative. The Burden of Musculoskeletal Diseases in the United States (BMUS). In: In. Fourth ed. Rosemont, IL. 2018: Available at https://www.boneandjointburden.org/fourth-edition. Accessed June 12, 2023

Verma, D., Bach, K., & Mork, P. J. (2021). Application of machine learning methods on patient reported outcome measurements for predicting outcomes: A literature review. In Informatics (Vol. 8, Issue 3). https://doi.org/10.3390/informatics8030056

World Health Organization. Osteoarthritis. 14 July 2023. https://www.who.int/news-room/fact-sheets/detail/osteoarthritis

Zhang, W., Nuki, G., Moskowitz, R. W., Abramson, S., Altman, R. D., Arden, N. K., Bierma-Zeinstra, S., Brandt, K. D., Croft, P., Doherty, M., Dougados, M., Hochberg, M., Hunter, D. J., Kwoh, K., Lohmander, L. S., & Tugwell, P. (2010). OARSI recommendations for the management of hip and knee osteoarthritis. Part III: Changes in evidence following systematic cumulative update of research published through January 2009. Osteoarthritis and Cartilage, 18(4).

Comparative Analysis of Single and Ensemble Support Vector Regression Methods for Software Development Effort Estimation

Mohamed Hosni[a]
MOSI Research Team, LM2S3 Laboratory, ENSAM, Moulay Ismail Iniversity of Meknes, Meknes, Morocco
m.hosni@umi.ac.ma

Keywords: Support Vector Regression, Kernels, Software Effort Estimation, Ensemble Effort Estimation.

Abstract: Providing an accurate estimation of the effort required to develop a software project is crucial for its success. These estimates are essential for managers to allocate resources effectively and deliver the software product on time and with the desired quality. Over the past five decades, various effort estimation techniques have been developed, including machine learning (ML) techniques. ML methods have been applied in software development effort estimation (SDEE) for the past three decades and have demonstrated promising levels of accuracy. Numerous ML methods have been explored, including the Support Vector Regression (SVR) technique, which has shown competitive performance compared to other ML techniques. However, despite the plethora of proposed methods, no single technique has consistently outperformed the others in all situations. Prior research suggests that generating estimations by combining multiple techniques in ensembles, rather than relying solely on a single technique, can be more effective. Consequently, this research paper proposes estimating SDEE using both individual ML techniques and ensemble methods based on SVR. Specifically, four variations of the SVR technique are employed, utilizing four different kernels: polynomial, linear, radial basis function, and sigmoid. Additionally, a homogeneous ensemble is constructed by combining these four variants using two types of combiners. An empirical analysis is conducted on six well-known datasets, evaluating performance using eight unbiased criteria and the Scott-Knott statistical test. The results suggest that both single and ensemble SVR techniques exhibit similar predictive capabilities. Furthermore, the SVR variant with the polynomial kernel is deemed the most suitable for SDEE. Regarding the combiner rule, the non-linear combiner yields superior accuracy for the SVR ensemble.

1 INTRODUCTION

Accurately predicting the effort required to develop a new software system during the initial phases of the software lifecycle remains a significant challenge in software project management. This estimation process, known as software development effort estimation (SDEE) (Wen et al., 2012), is critical for effective resource allocation and project planning.

Accurate estimates are critical, as errors can lead to major challenges for software managers. Charette (Charette, 2005) notes that inaccurate resource estimates are a significant contributor to software project failures. To address this issue, numerous effort estimation methods have been proposed and studied (de Barcelos Tronto et al., 2008), with machine learning (ML) techniques emerging as a particularly promising solution.

A systematic review (SLR) conducted by Wen et al. (Wen et al., 2012) identified seven ML techniques proposed for estimating software development effort. The review found that these ML techniques generally provide more accurate results than traditional non-ML methods. Additionally, the ensemble method, known as Ensemble Effort Estimation (EEE), has garnered significant attention within the SDEE research community. EEE involves combining estimates from multiple effort estimators using specific combination rules. Studies within the SDEE literature have extensively explored EEE techniques, with results suggesting that they yield more accurate estimates compared to single estimation methods.

The SLR performed a SLR focused on ensemble approaches in SDEE (Idri et al., 2016). This review analyzed 24 studies published between 2000 and 2016 and found that ensemble methods generally outperformed single techniques, demonstrating consistent performance across various datasets. The review identified 16 distinct techniques used for con-

[a] https://orcid.org/0000-0001-7336-4276

Hosni, M.
Comparative Analysis of Single and Ensemble Support Vector Regression Methods for Software Development Effort Estimation.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 509-516
ISBN: 978-989-758-716-0; ISSN: 2184-3228

structing ensembles, with Artificial Neural Networks (ANN) and Decision Trees (DT) being the most commonly employed. Additionally, it noted that 20 different combiners were utilized to generate ensemble outputs, with linear combiners being the most prevalent. An updated review by Cabral et al. (Cabral et al., 2023) in 2022, which covered studies from 2016 to 2021, confirmed these findings.

The Support Vector Regression (SVR) technique, introduced by Oliveira in 2006 for predicting software development effort (Oliveira, 2006), has been the subject of extensive research. Evidence suggests that SVR often provides more accurate results than many other ML techniques used in SDEE (Braga et al., 2008; Mahmood et al., 2022).

A key feature of SVR is its kernel, which maps the input space to a higher-dimensional feature space. Variations in SVR techniques, defined by different kernels, can lead to different estimation results.

This paper aims to assess the effectiveness of the Ensemble Effort Estimation approach based on SVR. The objective is to determine whether combining multiple SVR techniques with various kernels yields better performance than using a single SVR technique.

To achieve this, the paper explores an EEE approach that integrates four SVR techniques, each with distinct kernels and hyperparameter settings optimized using Particle Swarm Optimization (PSO). The study employs several combination rules, including three linear combiners (average, median, inverse ranked weighted mean) and one non-linear combiner (Multilayer Perceptron), to evaluate their impact on estimation accuracy.To address this objective, the paper investigates three key research questions (RQs):

- **(RQ1). Which of the four kernel methods used in the SVR techniques is most suitable for SDEE datasets?**
- **(RQ2). Does the SVR-EEE approach consistently outperform the single SVR technique, regardless of the combiners used?**
- **(RQ3). Among the combiners utilized, which one provides the highest accuracy for the proposed ensemble?**

The main features of this empirical work are as follows:

1. Development of an SVR-Ensemble technique that integrates four SVR methods with different kernels and hyperparameter settings.
2. Application of Particle Swarm Optimization (PSO) to optimize the hyperparameters of the four SVR variants.
3. Evaluation of various combiners for generating the final output of the ensemble.

The structure of this paper is as follows: Section 2 provides background information and reviews previous research on the topic. Section 3 offers an overview of the SVR technique. Section 4 details the materials and methods used in the study. Section 5 presents and discusses the empirical results. Finally, Section 6 concludes the paper and proposes directions for future research.

2 RELATED WORK

This section begins by defining Ensemble Effort Estimation (EEE) and then reviews the main findings from EEE studies in the context of SDEE literature.

EEE is an approach that combines multiple individual predictors using a specific combination rule. The literature distinguishes between two types of ensembles (Hosni et al., 2019; Hosni et al., 2018a; Hosni et al., 2021; Kocaguneli et al., 2011): homogeneous and heterogeneous. Homogeneous ensembles consist of multiple variants of the same ML technique or a combination of a single ML technique with meta-ensemble methods such as Bagging, Boosting, or Random Subspace. In contrast, heterogeneous ensembles combine at least two different ML techniques. The final output of an ensemble is obtained by aggregating the individual estimates from its components using a defined combination rule.

To explore the application of ensemble approaches in SDEE, Idri et al. (Idri et al., 2016) conducted a SLR analyzing papers published between 2000 and 2016. Their review, covering 24 papers, yielded the following main conclusions:

- Homogeneous ensembles were the most frequently studied, appearing in 17 out of the 24 papers.
- A total of 16 different effort estimation techniques were used to construct EEE.
- Machine learning techniques were the predominant choice for ensemble components, with Artificial Neural Networks (ANN) and Decision Trees (DT) being the most frequently investigated individual techniques.
- The Support Vector Regression (SVR) technique was explored in five studies, primarily for constructing heterogeneous ensembles.
- Twenty combination rules were employed to generate the final output of ensemble methods. These rules were categorized into linear and non-linear

types, with linear rules being the most extensively investigated.

- Overall, ensemble methods demonstrated better performance compared to single techniques.

It is also noteworthy that the SLR conducted by Cabral et al. (Cabral et al., 2023) reached similar conclusions regarding the use of EEE.

3 SUPPORT VECTOR REGRESSION: A BRIEF DESCRIPTION

Support Vector Regression (SVR) is a supervised ML technique tailored for regression tasks, extending the principles of Support Vector Machines, which are primarily employed for classification (Vapnik et al., 1998). The main concept behind SVR is to find a hyperplane that optimally fits the data while minimizing prediction errors. SVR is capable of modeling both linear and non-linear relationships between independent and dependent variables by utilizing kernel functions to map input features into a high-dimensional space. Commonly used kernels include linear, polynomial, radial basis function (RBF), and sigmoid. SVR is also robust to outliers, making it highly effective across various scenarios.

SVR was first applied to Software Development Effort Estimation by Oliveira (Oliveira, 2006; Oliveira et al., 2010). Subsequent studies in the SDEE literature have shown that SVR achieves competitive accuracy compared to other ML techniques (Braga et al., 2008; Hosni et al., 2018b; Braga et al., 2007; Mahmood et al., 2022; López-Martín, 2021).

Several parameters significantly influence SVR performance:

- **Regularization Parameter (C):** Controls the trade-off between model complexity and error minimization.
- **Kernel Parameters:** Determine the nature of the non-linear mapping.

Careful tuning of these parameters is crucial for optimizing SVR's predictive performance.

4 EMPIRICAL DESIGN

This section first introduces the performance metrics used to evaluate the accuracy of the proposed SDEE techniques and the statistical test employed to assess their significance. It then covers the hyperparameter optimization methods applied in the study. The dataset utilized for developing the SDEE techniques is also presented. Lastly, the section details the methodology for constructing and evaluating the predictive model.

4.1 Performance Measures and Statistical Test

To evaluate the accuracy of the proposed techniques, we employed eight commonly used performance criteria in the SDEE literature. These criteria include Mean Absolute Error (MAE), Mean Balanced Relative Error (MBRE), Mean Inverted Balanced Relative Error (MIBRE), and their corresponding median values, Logarithmic Standard Deviation (LSD), and Pred (25%) (Miyazaki et al., 1991; Foss et al., 2003; Hosni, 2023; Mustafa and Osman, 2024; Kumar et al., 2020).

Additionally, we used Standardized Accuracy (SA) and Effect Size to determine whether the SDEE techniques provided better estimates compared to random guessing (Shepperd and MacDonell, 2012). The mathematical formulas for these performance indicators are detailed in Equations (1)–(8).

$$AE_i = |e_i - \widehat{e_i}| \tag{1}$$

$$Pred(0.25) = \frac{100}{n}\sum_{i=1}^{n}\begin{cases}1 & \text{if } \frac{AE_i}{e_i} \leqslant 0.25\\ 0 & \text{otherwise}\end{cases} \tag{2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} AE_i \tag{3}$$

$$MBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{AE_i}{\min(e_i, \widehat{e_i})} \tag{4}$$

$$MIBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{AE_i}{\max(e_i, \widehat{e_i})} \tag{5}$$

$$LSD = \sqrt{\frac{\sum_{i=1}^{n}(\lambda_i + \frac{s^2}{2})^2}{n-1}} \tag{6}$$

$$SA = 1 - \frac{MAE_{p_i}}{\overline{MAE}_{p_0}} \tag{7}$$

$$\triangle = \frac{MAE_{p_i} - \overline{MAE}_{p_0}}{S_{p_0}} \tag{8}$$

where:

- The actual effort and predicted effort for the i-th project are denoted by e_i and $\widehat{e_i}$, respectively.
- The average Mean Absolute Error (MAE) from multiple random guessing runs is represented as $\overline{MAE}p_0$. This value is obtained by randomly sampling (with equal probability) from the remaining $n-1$ cases and setting $\widehat{e_i} = e_r$, where r is a random index from 1 to n, excluding i. This randomization

approach is robust as it does not assume specific distribution characteristics of the data.

- The Mean Absolute Error for prediction technique i, denoted as $MAEp_i$, is used as a benchmark in comparison with the sample standard deviation of the random guessing strategy.
- The value of λ_i is calculated as the difference between the natural logarithm of e_i and the natural logarithm of $\widehat{e_i}$.
- The estimator s^2 is employed to estimate the residual variance associated with λ_i.

To group the developed SDEE techniques based on their predictive capabilities, we applied the Scott-Knott statistical test (Hosni et al., 2018b). For validation, we utilized the Leave-One-Out Cross-Validation (LOOCV) technique to construct and evaluate these SDEE techniques.

4.2 Hyperparameters Optimization Techniques

In this paper, the optimal parameters for the developed SVR techniques were determined using the **Particle Swarm Optimization (PSO)** technique. Table 1 details the range of hyperparameters considered by PSO to identify the optimal settings. For the Multi-Layer Perceptron (MLP) combination rule, used to generate the final prediction of the proposed ensemble, hyperparameters were optimized using the **Grid Search (GS)** technique. Table 1 outlines the parameter ranges explored by GS. Both optimization techniques utilized the MAE as the fitness function, with the goal of minimizing the MAE value.

4.3 Datasets

To evaluate the performance of the proposed techniques for estimating software development effort, we selected six well-established datasets from two different repositories (Kocaguneli et al., 2011; Kumar and Srinivas, 2024). Five datasets—Albrecht, COCOMO81, Desharnais, Kemerer, and Miyazaki—were sourced from the PROMISE repository. Additionally, one dataset was obtained from the ISBSG data repository. Comprehensive details about these datasets, including their size, number of attributes, and descriptive statistics of effort (such as minimum, maximum, mean, median, skewness, and kurtosis), are provided in Table 2.

4.4 Methodology Used

This subsection details the methodology used to address our RQs, with the analysis performed independently for each dataset. We developed four SVR techniques, each employing a distinct kernel: Linear, Polynomial, Radial Basis Function (RBF), and Sigmoid. The homogeneous ensemble integrates these four SVR variants. The steps of the empirical analysis are outlined below:

- **Step 1:** Construct SVR models using Particle Swarm Optimization (PSO) with 10-fold cross-validation to determine the optimal hyperparameters for each kernel variant.
- **Step 2:** Select the optimal hyperparameters identified in Step 1 for each SVR variant.
- **Step 3:** Rebuild the SVR models with the selected hyperparameters using LOOCV.
- **Step 4:** Evaluate the performance of the SVR models using SA and effect size, and compare these results to the 5% quantile of random guessing.
- **Step 5:** Evaluate the accuracy of the SVR models using eight performance metrics: MAE, MdAE, MIBRE, MdIBRE, MBRE, MdBRE, LSD, and Pred (25
- **Step 6:** Construct SVR ensembles by combining the four SVR variants using the following combination rules: median, average, inverse-ranked weighted mean (IRWM), and Multi-Layer Perceptron (MLP).
- **Step 7:** Evaluate and report the performance of the SVR ensembles using the same eight metrics.
- **Step 8:** Rank the single SVR models and the ensembles using the Borda count voting system.
- **Step 9:** Apply the Scott-Knott statistical test based on Absolute Error (AE) to group the techniques and identify clusters with similar predictive capabilities.

For ease of reference, the following abbreviations will be used:

- **Single SVR Models: SVR** followed by the kernel type.
  - SVR with Linear Kernel: SVRL
  - SVR with Polynomial Kernel: SVRP
  - SVR with Radial Basis Function Kernel: SVRR
  - SVR with Sigmoid Kernel: SVRS
- **Ensemble SVR Models: E** followed by the combiner type.

Table 1: Range of Hyperparameters for PSO and GS.

SVR-Linear Kernel	C{1, 100}, Epsilon {0.001, 0.5}
SVR-RBF Kernel	C{1, 100}, Epsilon {0.001, 0.5}, gamma {0.001, 1}
SVR-Poly Kernel	C{1, 100}, Epsilon {0.001, 0.5}, degree {1, 10}
SVR-Sigmoid Kernel	C{1, 100}, Epsilon {0.001, 0.5}, Coef0 {0.001, 1}
MLP Combiner	hidden_layer_sizes: {(8,), (8,16), (8, 16, 32)}, activation: {'relu', 'tanh', 'identity', 'logistic'}, solver: {'adam', 'lbfgs', 'sgd'}, learning_rate: {'constant', 'adaptive', 'invscaling'}

Table 2: Overview of Descriptive Statistics for the Six Selected Datasets.

Dataset	**Size**	**#Features**	**Effort**					
			Min	**Max**	**Mean**	**Median**	**Skewness**	**Kurtosis**
Albrecht	24	7	0.5	105	21.87	11	2.30	4.7
COCOMO81	252	13	6	11400	683.44	98	4.39	20.5
Desharnais	77	12	546	23940	4833.90	3542	2.03	5.3
ISBSG	148	10	24	60270	6242.60	2461	3.05	11.3
Kemerer	15	7	23	1107	219.24	130	3.07	10.6
Miyazaki	48	8	5.6	1586	87.47	38	6.26	41.3

- Ensemble SVR with MLP as the combiner: EMLP
- Ensemble SVR with average as the combiner: EAVR
- Ensemble SVR with median as the combiner: EMED
- Ensemble SVR with IRWM as the combiner: EIRWM

5 EMPIRICAL RESULTS

In this section, we present the empirical results from our experiments. The experiments were executed using Python and its associated libraries, while the Scott-Knott (SK) test was conducted using the R programming language.

5.1 Single SVR Techniques

The initial phase of our empirical analysis involved identifying the optimal parameters for the various SVR techniques. To achieve this, we employed PSO technique to fine-tune the hyperparameters of the SVR models. This optimization process was applied to the four SVR variants across the six selected datasets, utilizing 10-fold cross-validation.

Following parameter optimization, we constructed the SVR models using the identified optimal parameters. The performance of these models was then compared against the 5% quantile of random guessing, which served as our baseline estimator. Specifically, we assessed whether the MAE of the SVR variants on each dataset was lower than the 5% quantile of random guessing. This comparison helped determine if the SVR techniques were effectively making predictions.

To further validate the results, we evaluated the effect size to assess the significance of the improvement over the baseline estimator. Table 3 presents the SA and effect size of the constructed SVR techniques. The results demonstrate a significant improvement over the baseline estimator, confirming that all SVR variants produced better predictions. Thus, we can confidently assert that the proposed SVR techniques are effective in estimating software development effort.

The next phase of our experimental protocol involves evaluating the predictive performance of the proposed techniques using eight established performance indicators. These indicators, recognized for their objectivity, are crucial for assessing the accuracy of the techniques. To synthesize the results from these indicators, we utilized the Borda count voting system. The final rankings of the single SVR techniques across the selected datasets are detailed in Table 4.

The rankings of the SVR techniques varied depending on the dataset and the kernel used. Notably, the SVR technique with a polynomial kernel (SVRP) emerged as the most effective, achieving the highest rank in five out of six datasets. The SVR technique with a linear kernel (SVRL) performed well, securing

Table 3: SA and Effect size values of the SVR techniques across the six datasets.

Dataset	**COCOMO**		**ISBSG**		**Miyazaki**		**Desharnais**		**Albrecht**		**Kemerer**	
SA5%	15%		13%		34%		15%		30%		34%	
Technique	SA	Delta	SA	Delta	SA	Delta	SA	Delta	SA	Delta	SA	Delta
SVRL	53%	-5.41	40%	-4.68	66%	-2.40	42%	-4.42	76%	-3.83	65%	-2.50
SVRR	53%	-5.47	40%	-4.60	63%	-2.29	41%	-4.31	91%	-4.62	63%	-2.44
SVRP	96%	-9.84	55%	-6.35	88%	-3.17	54%	-5.72	89%	-4.51	89%	-3.42
SVRS	39%	-3.98	37%	-4.32	11%	-0.39	35%	-3.69	33%	6.56	45%	-1.75

the second position in four out of six datasets.

In contrast, the SVR technique using the sigmoid kernel consistently ranked the lowest across all datasets, indicating its comparatively inferior performance.

The following summarizes the ranking of the four SVR techniques across the six selected datasets:

- **Polynomial Kernel (SVRP):**
  - **Top Ranking:** Achieved the highest ranking in 5 out of 6 datasets.
  - **Overall Performance:** Demonstrated superior performance in most cases.
- **Linear Kernel (SVRL):**
  - **Top Ranking:** Achieved the highest ranking in 1 out of 6 datasets.
  - **Second Position:** Secured the second position in 4 out of 6 datasets.
  - **Overall Performance:** Consistently performed well, ranking second most frequently.
- **Radial Basis Function Kernel (SVRR):**
  - **Top Ranking:** Did not achieve the highest ranking in any dataset.
  - **Overall Performance:** Exhibited variable performance, generally not leading but still competitive.
- **Sigmoid Kernel (SVRS):**
  - **Top Ranking:** Did not achieve the highest ranking in any dataset.
  - **Overall Performance:** Consistently ranked the lowest in all datasets, indicating the least effectiveness.

Table 4: Ranking of the four SVR techniques on the selected datasets.

COC.	**ISBSG**	**Miyazaki**	**Desh.**	**Albrecht**	**Kemerer**
SVRP	SVRP	SVRP	SVRP	SVRR	SVRP
SVRR	SVRL	SVRL	SVRL	SVRP	SVRL
SVRL	SVRR	SVRR	SVRR	SVRL	SVRR
SVRS	SVRS	SVRS	SVRS	SVRS	SVRS

5.2 SVR Ensembles

The next phase of our experimental design involves constructing a homogeneous ensemble from the four SVR techniques. We develop four different ensembles, each distinguished by its combination rule. Specifically, we use two types of combiners to generate the final output of the proposed ensembles:

- **Linear Combiners:** AVG, MED, IRWM.
- **Non-Linear Combiner:** MLP.

The hyperparameters of the MLP combiner were optimized using the grid search technique.

The ensemble approach combines four SVR variants, each utilizing a different kernel. These variants have demonstrated superior performance compared to random guessing, as shown in the previous section. Therefore, the four ensembles constructed are expected to outperform the baseline estimator.

To assess the performance of the proposed ensembles, we utilize eight performance metrics and compare them with the individual SVR techniques. The final rankings are determined using the Borda count voting system, with results presented in Table 5.

The results reveal that ensemble methods achieved the top ranking only twice. In comparison, SVRP was ranked first in three datasets, and SVRR secured the top position in one dataset. It is evident that no single ensemble approach consistently outperformed all other techniques across every dataset. The performance of the ensembles varied depending on the dataset. However, it is noteworthy that, in the majority of cases, the ensemble methods outperformed the SVRS technique. On the other hand, certain SVR variants outperformed the ensemble methods in several datasets, with the exception of the ISBSG and Desharnais datasets, where ensembles generally surpassed the single SVR techniques, except for SVRP. Consequently, there is no definitive evidence to establish the superiority of any specific technique over others.

To statistically assess the significant differences between the proposed techniques, we employed the

Table 5: Rank of Single and Ensemble SVR techniques over the six datasets.

Rank	**COCOMO**	**ISBSG**	**Miyazaki**	**Desharnais**	**Albrecht**	**Kemerer**
1	SVRP	**EMLP**	**EMLP**	SVRP	SVRR	SVRP
2	**EMLP**	SVRP	SVRP	**EIRWM**	**EMLP**	**EMLP**
3	SVRR	**EIRWM**	**EIRWM**	**EAVR**	SVRP	**EIRWM**
4	**EIRWM**	**EAVR**	SVRL	**EMLP**	**EMED**	SVRL
5	**EMED**	**EMED**	SVRR	SVRL	SVRL	SVRR
6	**EAVR**	SVRL	**EMED**	**EMED**	**EIRWM**	**EAVR**
7	SVRL	SVRR	SVRS	SVRR	SVRS	**EMED**
8	SVRS	SVRS	**EAVR**	SVRS	**EAVR**	SVRS

Table 6: Clusters identified by SK test.

Technique	**COCOMO**	**ISBSG**	**Miyazaki**	**Desharnais**	**Albrecht**	**Kemerer**
EAVR	2	2	3	2	5	2
EIRWM	1	2	3	1	4	2
EMED	2	2	3	2	3	2
EMLP	1	1	1	1	1	1
SVRL	2	2	3	2	3	2
SVRP	1	1	2	1	2	1
SVRR	2	2	3	2	1	2
SVRS	2	2	4	2	6	3

Scott-Knott statistical test. This test was used to identify clusters of techniques with comparable predictive capabilities based on AE. The identified clusters for each dataset are detailed in Table 6.

The SK test revealed two clusters in the Desharnais, COCOMO, and ISBSG datasets, four clusters in the Miyazaki dataset, and three clusters in the Kemerer dataset. The Albrecht dataset had the highest number of clusters. In the COCOMO dataset, the SK test showed no significant difference between the SVRP and EMLP techniques. Similar findings were observed for the ISBSG and Kemerer datasets. For the Desharnais dataset, the EIRWM, EMLP, and SVRP techniques were grouped into the same cluster, indicating that they have similar predictive capabilities. In the Albrecht dataset, the most effective cluster included both EMLP and SVRR techniques. For the Miyazaki dataset, the EMLP ensemble was part of the top-performing cluster. Notably, the SVRS technique consistently appeared in the lowest-performing cluster across all datasets, while other ensemble methods, such as those using average or median combiners, did not fall into the worst cluster.

These results suggest that ensemble methods, particularly those incorporating non-linear rules like MLP, show promising performance.

6 CONCLUSIONS AND FUTURE WORK

This paper investigates the potential of Support Vector Regression in SDEE. The study evaluates four SVR variants tailored for SDEE and proposes a homogeneous ensemble of these variants, employing three linear and one non-linear combiner. The optimization of the SVR variants is performed using the PSO technique. Six widely recognized datasets are used to assess the proposed approaches, and various performance indicators are applied, with the LOOCV method utilized for validation. The research addresses three RQs, with the key findings summarized as follows:

- **(RQ1).** The SVR variant using the polynomial kernel proves to be the most suitable for SDEE. Overall results show that this variant outperforms others using different kernels in terms of accuracy.
- **(RQ2).** There is no conclusive evidence of the superiority of SVR ensembles over single SVR techniques. Empirical results suggest that both approaches achieve similar predictive accuracy, with no statistically significant differences.
- **(RQ3).** The results indicate that the SVR ensemble using the non-linear MLP rule achieves higher performance accuracy compared to ensembles using linear rules.

Ongoing work focuses on evaluating SVR techniques incorporating feature selection methods and developing a statistical framework for dynamically selecting SVR variants as ensemble members. Further exploration of alternative combination rules, particularly non-linear ones, is essential to validate and extend the study's findings.

REFERENCES

Braga, P. L., Oliveira, A. L., and Meira, S. R. (2007). Software effort estimation using machine learning techniques with robust confidence intervals. In *7th international conference on hybrid intelligent systems (HIS 2007)*, pages 352–357. IEEE.

Braga, P. L., Oliveira, A. L., and Meira, S. R. (2008). A ga-based feature selection and parameters optimization for support vector regression applied to software effort estimation. In *Proceedings of the 2008 ACM symposium on Applied computing*, pages 1788–1792.

Cabral, J. T. H. d. A., Oliveira, A. L., and da Silva, F. Q. (2023). Ensemble effort estimation: An updated and extended systematic literature review. *Journal of Systems and Software*, 195:111542.

Charette, R. N. (2005). Why software fails. *IEEE spectrum*, 42(9):36.

de Barcelos Tronto, I. F., da Silva, J. D. S., and Sant'Anna, N. (2008). An investigation of artificial neural networks based prediction systems in software project management. *Journal of Systems and Software*, 81(3):356–367.

Foss, T., Stensrud, E., Kitchenham, B., and Myrtveit, I. (2003). A simulation study of the model evaluation criterion mmre. *IEEE Transactions on software engineering*, 29(11):985–995.

Hosni, M. (2023). On the value of combiners in heterogeneous ensemble effort estimation. In *KDIR*, pages 153–163.

Hosni, M., Idri, A., and Abran, A. (2018a). Improved effort estimation of heterogeneous ensembles using filter feature selection. In *ICSOFT*, pages 439–446.

Hosni, M., Idri, A., and Abran, A. (2019). Evaluating filter fuzzy analogy homogenous ensembles for software development effort estimation. *Journal of Software: Evolution and Process*, 31(2):e2117.

Hosni, M., Idri, A., and Abran, A. (2021). On the value of filter feature selection techniques in homogeneous ensembles effort estimation. *Journal of Software: Evolution and Process*, 33(6):e2343.

Hosni, M., Idri, A., Abran, A., and Nassif, A. B. (2018b). On the value of parameter tuning in heterogeneous ensembles effort estimation. *Soft Computing*, 22:5977–6010.

Idri, A., Hosni, M., and Abran, A. (2016). Systematic literature review of ensemble effort estimation. *Journal of Systems and Software*, 118:151–175.

Kocaguneli, E., Menzies, T., and Keung, J. W. (2011). On the value of ensemble effort estimation. *IEEE Transactions on Software Engineering*, 38(6):1403–1416.

Kumar, K. H. and Srinivas, K. (2024). An improved analogy-rule based software effort estimation using htrr-rnn in software project management. *Expert Systems with Applications*, 251:124107.

Kumar, P. S., Behera, H. S., Kumari, A., Nayak, J., and Naik, B. (2020). Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades. *Computer Science Review*, 38:100288.

López-Martín, C. (2021). Effort prediction for the software project construction phase. *Journal of Software: Evolution and Process*, 33(7):e2365.

Mahmood, Y., Kama, N., Azmi, A., Khan, A. S., and Ali, M. (2022). Software effort estimation accuracy prediction of machine learning techniques: A systematic performance evaluation. *Software: Practice and experience*, 52(1):39–65.

Miyazaki, Y., Takanou, A., Nozaki, H., Nakagawa, N., and Okada, K. (1991). Method to estimate parameter values in software prediction models. *Information and Software Technology*, 33(3):239–243.

Mustafa, E. I. and Osman, R. (2024). A random forest model for early-stage software effort estimation for the seera dataset. *Information and Software Technology*, 169:107413.

Oliveira, A. L. (2006). Estimation of software project effort with support vector regression. *Neurocomputing*, 69(13-15):1749–1753.

Oliveira, A. L., Braga, P. L., Lima, R. M., and Cornélio, M. L. (2010). Ga-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation. *information and Software Technology*, 52(11):1155–1166.

Shepperd, M. and MacDonell, S. (2012). Evaluating prediction systems in software project estimation. *Information and Software Technology*, 54(8):820–827.

Vapnik, V. N., Vapnik, V., et al. (1998). *Statistical learning theory*. wiley New York.

Wen, J., Li, S., Lin, Z., Hu, Y., and Huang, C. (2012). Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology*, 54(1):41–59.

Software Testing Effort Estimation Based on Machine Learning Techniques: Single and Ensemble Methods

Mohamed Hosni[1][a], Ibtissam Medarhri[2][b] and Juan Manuel Carrillo de Gea[3][c]

[1]*MOSI Research Team, LM2S3, ENSAM, Moulay Ismail University of Meknes, Morocco*

[2]*MMCS Research Team, LMAID, ENSMR-Rabat, Morocco*

[1]*Department of Informatics and Systems, Faculty of Computer Science, University of Murcia, Spain*
m.hosni@umi.ac.ma, medarhri@enim.ac.ma, jmcdg1@um.es

Keywords: Software Testing, Software Testing Effort, Machine Learning, Ensemble Method, ISBSG.

Abstract: Delivering an accurate estimation of the effort required for software system development is crucial for the success of any software project. However, the software development lifecycle (SDLC) involves multiple activities, such as software design, software build, and software testing, among others. Software testing (ST) holds significant importance in the SDLC as it directly impacts software quality. Typically, the effort required for the testing phase is estimated as a percentage of the overall predicted SDLC effort, typically ranging between 10% and 60%. However, this approach poses risks as it hinders proper resource allocation by managers. Despite the importance of this issue, there is limited research available on estimating ST effort. This paper aims to address this concern by proposing four machine learning (ML) techniques and a heterogeneous ensemble to predict the effort required for ST activities. The ML techniques employed include K-nearest neighbor (KNN), Support Vector Regression, Multilayer Perceptron Neural Networks, and decision trees. The dataset used in this study was obtained from a well-known repository. Various unbiased performance indicators were utilized to evaluate the predictive capabilities of the proposed techniques. The overall results indicate that the KNN technique outperforms the other ML techniques, and the proposed ensemble showed superior performance accuracy compared to the remaining ML techniques.

1 INTRODUCTION

The software development life cycle (SDLC) encompasses a comprehensive range of activities that cover multiple aspects of a software project. These activities include strategic planning, thorough requirements specification, meticulous analysis and design, precise programming, rigorous testing, seamless integration, smooth deployment, and various other supportive tasks. Together, they form a cohesive framework for the successful development and implementation of high-quality software systems (Radliński, 2023). Ensuring precise estimation of the effort needed to accomplish each of these activities is crucial for the overall success of the project (Charette, 2005). Despite the majority of research in the literature focusing on proposing automated techniques for accurate effort estimation in software development (Hosni et al., 2019a; Azzeh and Nassif, 2013), there has been relatively limited research conducted specifically on predicting the effort required to complete a specific activity in the SDLC, such as testing, even though it is a significant and challenging area. Therefore, this research work attempts to propose a software testing effort estimation technique based on machine learning methods.

Recently, a systematic literature review (SLR) was conducted on the use of ML in software testing (Ajorloo et al., 2024). This work systematically analyzes 40 studies published between 2018 and 2024, exploring various ML methods, including supervised, unsupervised, reinforcement, and hybrid approaches in software testing. It highlights ML's significant role in automating test case generation, prioritization, and fault detection, but also identifies a critical gap in the area of software test effort prediction—an important element for effective resource management, cost estimation, and project scheduling. Despite its importance, the review reveals that few studies specifically address this area, underscoring the urgent need for

[a] https://orcid.org/0000-0001-7336-4276
[b] https://orcid.org/0009-0003-0052-8702
[c] https://orcid.org/0000-0002-3320-622X

Hosni, M., Medarhri, I. and Carrillo de Gea, J.
Software Testing Effort Estimation Based on Machine Learning Techniques: Single and Ensemble Methods.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 517-524
ISBN: 978-989-758-716-0; ISSN: 2184-3228

further research on ML-based models to improve test effort predictions and enhance overall software testing efficiency.

Software testing holds significant importance in the SDLC as it serves to identify defects, errors, and inconsistencies within a software system (López-Martín, 2022). The primary objective of this important phase is to execute software components or systems to uncover bugs, verify adherence to specified requirements, and ensure the overall quality of the software product. By conducting comprehensive testing, developers can detect and rectify any flaws, ensuring that the software meets the desired standards and functions optimally (Radliński, 2023). The testing process plays a critical role in enhancing the reliability, performance, and user experience of the software, contributing to the success of the overall development project.

Software testing activities play a vital role in evaluating the functionality of software and determining the extent to which it meets stakeholders' expectations. Essentially, this phase ensures the software's desired quality. In terms of time and cost, software testing holds significant importance within the SDLC. Researchers have made several efforts to estimate the effort required for conducting testing activities (Radliński, 2023). Typically, the effort needed to test a software system is measured in person-hours (López-Martín, 2022). During the planning phase of a project, the overall effort required for the SDLC is estimated, and a certain percentage is allocated to account for software testing activities. However, accurately predicting the effort necessary for testing poses challenges due to the considerable variability in the percentage allocation for testing critical software components. This percentage can vary widely, from 10% to 60% or even higher (López-Martín, 2022). Thus, accurately estimating the effort required for testing remains a complex task.

ML techniques have been widely employed for over three decades to estimate software development effort with a higher degree of accuracy (Hosni and Idri, 2018). These techniques utilize historical data from completed projects to uncover complex relationships between various software factors and the effort required to develop a software system (Ali and Gravino, 2019; Wen et al., 2012). This enables ML models to generate more accurate predictions, overcoming the limitations of traditional software estimation techniques, such as parametric methods. Unlike traditional approaches, ML techniques can capture non-linear relationships between the target variable (i.e., effort) and the independent variables. This flexibility makes ML models well-suited for providing reliable estimations, which in turn assist project managers in making informed decisions regarding resource allocation and effectively monitoring overall project progress.

In Software Development Effort Estimation (SDEE), researchers have extensively explored a novel approach known as ensemble effort estimation (EEE) (Hosni et al., 2019b; Idri et al., 2016; d. A. Cabral et al., 2023). This technique involves combining multiple ML techniques into a single ensemble model, utilizing a combination rule to generate predictions. The EEE approach has demonstrated superior accuracy compared to using a single ML technique. Extensive literature reports consistently indicate that EEE outperforms individual ensemble members in most cases, highlighting the effectiveness of the ensemble approach in improving the accuracy of SDEE.

In this paper, our objective is to explore the application of well-established ML techniques in SDEE specifically for estimating the effort required in software testing activities. We have selected four widely used ML techniques: k-nearest neighbor (KNN), Support Vector Regression (SVR), Multilayer Perceptron (MLP) Neural Networks, and decision trees (DTs). Additionally, we propose an ensemble model that combines these four ML techniques. To obtain the final estimation from the ensemble, three combiners are employed: average, median, and inverse ranked weighted mean.

To conduct our study, we utilized a historical dataset obtained from the International Software Benchmarking and Standards Group (ISBSG) database, Release 12. In this research work, we address three research questions (RQs):

- **(RQ1). Among the four ML techniques used, which one generates the most accurate results?**
- **(RQ2). Is there any evidence that the proposed ensemble method performs better than the individual ML techniques?**
- **(RQ3). What are the main features that impact software testing effort (STE) among the input features used for the ML techniques?**

The main features of this paper are as follows:

- Utilizing four well-known ML techniques for estimating software testing effort (STE).
- Employing an ensemble method for estimating STE.
- Evaluating the predictive capabilities of these STE techniques using unbiased performance measures.

- Identifying the most significant features that impact the estimation of STE.

The organization of the remaining parts of this paper is as follows: Section 2 presents a comprehensive analysis of previous studies. Section 3 provides the list of the ML techniques employed in this research. Section 4 outlines the methodology implemented, including the materials utilized. Section 5 discusses the significant findings derived from the study. Lastly, the concluding section summarizes the paper and proposes future research directions.

2 RELATED WORK

This section presents some related work conducted in the literature of STE estimation and defines the EEE approach.

López-Martín (López-Martín, 2022) carried out an empirical study to explore the use of ML techniques for predicting software testing effort (STE) in the software development lifecycle (SDLC). The research examined five ML models—case-based reasoning, artificial neural networks (ANN), support vector regression (SVR), genetic programming, and decision trees (DTs)—to assess their accuracy in estimating software development effort (SDEE). The models were trained and evaluated using datasets from the ISBSG, which were chosen based on factors such as data quality, development type, platform, programming language, and resource level. The findings revealed that support vector regression (SVR) provided the most accurate predictions, particularly when evaluated using mean absolute error (MAE).

Labidi et al. (Labidi and Sakhrawi, 2023) conducted an empirical study aimed at predicting software testing effort (STE) using ensemble methods. The proposed approach combined three machine learning techniques: ANN, SVR, and DTs, with each model optimized through grid search. The ISBSG dataset was employed after a preprocessing step for empirical evaluation. Results indicated that the ensemble model outperformed the individual ML techniques based on performance metrics such as root mean square error (RMSE), R-squared, and MAE. However, the study lacks specific details about the dataset used for training and testing, only mentioning that 17 features were used as inputs for the predictive models. To the best of the authors' knowledge, this study, along with another, represents the limited research exploring ML techniques for predicting software testing effort.

In the last decade, there has been significant investigation into the ensemble approach in the context of SDEE. This approach involves predicting the effort needed to develop a software system by using multiple estimators. Ensembles can be categorized into two types (Azzeh et al., 2015; Elish et al., 2013): homogeneous and heterogeneous. Homogeneous ensembles combine at least two variants of the same estimation technique or combine one estimation technique with a meta-learner such as Bagging, Boosting, or Random Subspace. Heterogeneous ensembles, on the other hand, involve combining at least two different techniques. A review conducted by Idri et al. (Idri et al., 2016) identified 16 SDEE techniques that have been used to construct EEE techniques. The review revealed that the homogeneous type of ensemble was the most frequently investigated. In terms of combiners, the review identified 20 different combiners that were adopted to merge the individual estimates provided by the ensemble members. It was found that linear rules were the most commonly used type of combiner.

3 MACHINE LEARNING

Four ML techniques were employed in this study : KNN (Altman, 1992), MLP (Simon, 1999), SVR (Simon, 1999), and DT (Jeffery et al., 2001), besides an heterogeneous ensemble consisting of the four ML techniques using three combiners: average, median, and inverse ranked weighted mean.

4 EMPIRICAL DESIGN

This section outlines the experimental design adopted to conduct the experiments presented in this paper. It begins by specifying the performance metrics and statistical tests used to assess the accuracy of the proposed predictive models. Next, it details the use of the grid search hyperparameter optimization technique to fine-tune the parameter settings of the predictive models. It then provides information on the dataset chosen for empirical analysis. Finally, it describes the methodology employed for building the predictive models.

4.1 Performance Metrics and Statistical Test

To evaluate the accuracy of the proposed techniques, we employed a set of eight widely used performance criteria commonly found in the SDEE literature. These criteria include Mean Absolute Error (MAE), Mean Balanced Relative Error (MBRE), Mean Inverted Balanced Relative Error (MIBRE), along with their respective median values, Logarithmic Standard Deviation (LSD), and Prediction at 25% (Pred(25)) (Miyazaki, 1991; Minku and Yao, 2013; Foss et al., 2003).

Additionally, to determine whether the investigated STEE techniques outperformed random guessing, we utilized standardized accuracy (SA) and effect size as additional evaluation measures (Shepperd and MacDonell, 2012). The mathematical formulas for these performance indicators are provided in Equations (1)-(8).

$$AE_i = |e_i - \widehat{e_i}| \tag{1}$$

$$Pred(0.25) = \frac{100}{n}\sum_{i=1}^{n}\begin{cases}1 & \text{if } \frac{AE_i}{e_i} \leqslant 0.25\\ 0 & \text{otherwise}\end{cases} \tag{2}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} AE_i \tag{3}$$

$$MBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{AE_i}{\min(e_i, \widehat{e_i})} \tag{4}$$

$$MIBRE = \frac{1}{n}\sum_{i=1}^{n}\frac{AE_i}{\max(e_i, \widehat{e_i})} \tag{5}$$

$$LSD = \sqrt{\frac{\sum_{i=1}^{n}(\lambda_i + \frac{s^2}{2})^2}{n-1}} \tag{6}$$

$$SA = 1 - \frac{MAE_{p_i}}{\overline{MAE}_{p_0}} \tag{7}$$

$$\triangle = \frac{MAE_{p_i} - \overline{MAE}_{p_0}}{S_{p_0}} \tag{8}$$

where:

- e_i and $\widehat{e_i}$ denote the actual and predicted effort, respectively, for the i^{th} project.
- The average mean absolute error from numerous random guessing trials is represented by $\overline{MAE}_{p_0}$. It is computed by randomly sampling (with equal probability) from the remaining $n-1$ cases and setting $\widehat{e_i} = e_r$, where r is a randomly selected value from 1 to n, excluding i. This randomization method is robust as it does not rely on any assumptions about the population.
- The mean of absolute errors for a given prediction technique i, denoted as MAE_{p_i}, corresponds to the standard deviation of the sample derived from the random guessing approach.
- λ_i is determined by taking the natural logarithm of e_i and subtracting the natural logarithm of $\widehat{e_i}$.
- The term s^2 is used as an estimator of the residual variance associated with λ_i.

The predictive models were built using the Leave-One-Out Cross-Validation (LOOCV) technique.

To assess the statistical significance of the proposed technique based on AE, the Scott-Knott (SK) test was employed. The SK test is a statistical method used to compare and rank different approaches or techniques based on their performance metrics. It helps determine whether there are significant differences in performance between the evaluated approaches.

4.2 Hyperparameters Optimization

Several papers in the SDEE literature have discussed hyperparameter settings in detail (Song et al., 2013; Hosni et al., 2018; Hosni, 2023). These studies have highlighted the importance of optimization techniques in enhancing the accuracy of predictive models. It has been observed that the performance of ML techniques in SDEE can vary significantly across different datasets. Consequently, using the same parameter settings for a given technique may result in an incorrect assessment of its predictive capability. To address this issue, we employ the grid search optimization method to determine the optimal parameter values for the selected models. Table 1 presents the predefined search space, specifying the range of optimal parameter values for each ML technique.

4.3 Datasets

The predictive analysis conducted in this paper utilized the dataset from the International Software Benchmarking Standards Group (ISBSG). This comprehensive dataset includes over 6,000 projects and more than 120 features covering aspects such as project size, effort, schedule, development type, and application environment. Prior to building the machine learning models, the dataset undergoes preprocessing. This process starts with selecting software projects with high data quality, adhering to the guidelines established by the ISBSG group. The selection criteria are based on the standards outlined in (Hosni et al., 2019a; Labidi and Sakhrawi, 2023).

Afterwards, we selected attributes that, according to the authors' knowledge, have a clear influence on the STE. As a result, we selected nine numerical features along with the target variable 'Effort Test'. The input features used for our predictive models are listed in Table 2. It is worth noting that any data rows with missing values were removed from the dataset.

Table 1: Range of parameters values for each ML technique.

Technique	Search space
KNN	'n_neighbors': [1,11], 'weights': ['uniform', 'distance'], 'metric': ['euclidean', 'manhattan', 'cityblock', 'minkowski']
SVR	'kernel': ['rbf', 'poly'], 'C': [5, 10, 20, 30, 40, 50, 100], 'epsilon': [0.0001, 0.001, 0.01, 0.1], 'degree': [2, 3, 4, 5, 6], 'gamma': [0.0001, 0.001, 0.01, 0.1]
MLP	'hidden_layer_sizes': [(8,), (8,16), (8, 16, 32), (8,16,32,64)], 'activation': ['relu', 'tanh', 'identity', 'logistic'], 'solver': ['adam', 'lbfgs', 'sgd'], 'learning_rate': ['constant', 'adaptive', 'invscaling'],
DT	'criterion': ['squared_error', 'friedman_mse', 'absolute_error', 'poisson'],'max_depth': [None] + [1, number of feature space],'max_features': [None, 'sqrt', 'log2']

Table 2: Selected features.

Feature	Importance score
Enquiry count	**0.131424**
File count	**0.12467**
Output count	**0.121829**
Adjusted function points	**0.12108**
Input count	**0.120946**
Max team size	**0.120207**
Interface count	**0.103466**
Value adjustment factor	**0.083748**
User base - locations	**0.07263**
Effort test	-

4.4 Evaluation Methodology

This subsection outlines the experimental design employed to develop and evaluate the proposed STE techniques in this paper.

- **Step 1:** Four ML algorithms: KNN, SVR, MLP and DT, were trained and optimized using grid search with 10-fold cross-validation to identify the best hyperparameters.
- **Step 2:** Optimal hyperparameter values were selected for each model based on the lowest Mean Absolute Error (MAE).
- **Step 3:** The models were then retrained using the identified optimal parameters and evaluated using LOOCV.
- **Step 4:** The validity of the optimized models was assessed through Standardized Accuracy (SA) and effect size analysis, comparing their performance against the 5% quantile of random guessing.
- **Step 5:** Performance was measured using a comprehensive set of indicators: Mean Absolute Error (MAE), Median Absolute Error (MdAE), Mean Inverted Balanced Relative Error (MIBRE), Median Inverted Balanced Relative Error (MdIBRE), Mean Balanced Relative Error (MBRE), Median Balanced Relative Error (MdBRE), Logarithmic Standard Deviation (LSD), and Prediction at 25% (Pred(25)).
- **Step 6:** A heterogeneous ensemble was created by integrating the four models using three combination methods: average (AVR), median (MED), and inverse rank-weighted mean (IRWM).
- **Step 7:** The ensemble's performance was evaluated using the same metrics outlined in Step 5.
- **Step 8:** The software effort estimation methods were ranked using the Borda count voting system, considering all eight performance metrics.
- **Step 9:** The Scott-Knott statistical test was applied to group the estimation techniques into statistically similar categories based on AE, identifying those with comparable predictive performance.

5 EMPIRICAL RESULTS

This section presents the empirical findings derived from the experiment conducted in this paper. The experiments were executed using various tools, with Python and its associated libraries being used to run the experiments. Additionally, the R programming language was utilized to perform the SK test.

5.1 Single Techniques Assessment

In this phase, the first step involves identifying the optimal parameters that yield improved estimates for each individual technique. To achieve this, multiple rounds of preliminary experiments were conducted using the grid search optimization technique. The hyperparameters were varied within the range values specified in Table 1 for the four selected ML techniques: KNN, SVR, MLP, and DT. The evaluation was performed using the 10-fold cross-validation technique. The objective function targeted for minimization was the MAE criterion. The rationale behind selecting MAE is its unbiased nature as a performance measure.

Subsequently, we constructed our predictive models using the optimal parameters identified in the previous step, employing the LOOCV technique for validation. This approach was selected for its ability to provide low bias and high variance estimates, enhancing the replicability of the study.

We then evaluated the reasonability of our STE techniques by comparing them to a baseline estimator suggested by Shepperd and MacDonell (Shepperd and MacDonell, 2012), which constructs an estimator through multiple runs of random guessing.

The evaluation was carried out using the Standardized Accuracy (SA) metric and effect size (Δ), as proposed by the authors. As shown in Table 3, all four ML techniques significantly outperformed random guessing, showing substantial improvement with effect sizes greater than 0.8 ($\Delta > 0.8$). Notably, all techniques exceeded the 5% quantile of random guessing. Among the techniques, KNN ranked highest in both SA and effect size improvement, while SVR ranked lowest.

Table 3: SA and effect size value of the constructed techniques.

Technique	**SA**	**Delta**
	$SA_{5\%}$= **0.2061**	
KNN	0.981245	-7.134
SVR	0.384077	-2.79237
MLP	0.548538	-3.98806
DT	0.554524	-4.03159

We then assessed the accuracy of the four ML techniques using the eight chosen performance metrics. The evaluation results are summarized in Table 4.

The KNN technique demonstrated the highest accuracy among the four ML techniques used in this study, consistently ranking first across all eight performance metrics. DT and MLP followed, frequently alternating between second and third positions across several indicators. SVR consistently ranked lowest across all performance measures.

These results suggest that the proposed approach provides satisfactory accuracy, with KNN standing out as the most effective technique for estimating STE among those evaluated.

5.2 Ensemble Methods

This step involves constructing the proposed heterogeneous ensemble using the four ML techniques. The ensemble produces the final estimation through three combiners: AVR, MED, and IRWM based on the MAE. This approach is grounded in SDEE literature, which indicates that ensembles typically achieve higher accuracy than individual estimation techniques.

Performance metrics of the constructed ensemble, based on the eight selected indicators, are presented in Table 5. The ensemble with the IRWM combiner (EIRWM) consistently outperformed the others, ranking first across all performance metrics. The ensembles with AVR (EAVR) and MED (EMED) combiners ranked second and third, respectively. The consistent rankings of the ensemble techniques across all performance indicators demonstrate their reliable and stable accuracy.

5.3 STE Techniques Comparison

In this step, we ranked all the proposed techniques using the eight accuracy measures. The final ranking was determined through the Borda count voting system, which considers all eight performance metrics. This approach was chosen because the accuracy of a technique can depend on the selected performance indicators, potentially leading to conflicting results as different metrics may produce varying rankings for each technique (Myrtveit et al., 2005; Mittas and Angelis, 2013). Table 6 presents the final rankings obtained through the Borda count system. As shown, the KNN technique achieved the top position, followed by the three heterogeneous ensembles, with SVR ranked last.

To validate these results, we conducted the SK statistical test to identify techniques with statistically similar predictive capabilities. The SK test was performed based on the AE of the proposed techniques. Table 6 shows the content of clusters identified by the SK test.

The first cluster contained only the KNN technique, while the second cluster included the proposed ensemble methods. The last cluster was comprised

Table 4: Performance metrics for the four ML techniques.

Technique	MAE	MdAE	MBRE	MdBRE	MIBRE	MdIBRE	PRED	LSD
KNN	17.66399	0	0.168228	0	0.019629	0	95.55556	0.320259
SVR	580.0858	336.8311	112995.8	1.602058	0.523648	0.615689	17.77778	3.649184
MLP	425.1941	313.952	106435.1	0.753829	0.456234	0.429819	24.44444	3.235486
DT	419.5556	225	75112.29	0.836956	0.453273	0.455621	24.44444	3.418981

Table 5: Accuracy performance of the ensemble methods.

Technique	MAE	MdAE	MBRE	MdBRE	MIBRE	MdIBRE	PRED	LSD
EAVR	306.2841	190.1564	73635.61	0.555469	0.374774	0.3571069	35.55556	3.103686
EMED	333.5098	207.3907	90773.51	0.710584	0.394308	0.4154042	35.55556	3.236895
EIRWM	244.7196	156.9293	55120.18	0.439571	0.327887	0.3053485	42.22222	2.971071

Table 6: Rank obtained by Borda Count Voting System, and identified Clusters.

Rank	Models	Cluster
1	KNN	1
2	EIRWM	2
3	EAVR	2
4	EMED	2
5	DT	3
6	MLP	3
8	SVR	4

solely of the SVR technique. Notably, the clusters identified by the SK test correspond closely with the rankings obtained through the Borda count method. This confirms that the KNN technique remains statistically the most superior, while the three proposed ensemble methods consistently outperform the other individual techniques.

5.4 Features Importance

An important aspect of our investigation was assessing feature importance in explaining the target variable, Effort Test. We employed the **ExtraTreesClassifier**, which uses multiple decision trees to evaluate and rank the significance of features within the dataset.

Table 2 shows the importance scores for each feature used in our predictive models. The results confirm that all features contribute to the target variable, aligning with our manual feature selection process on the original ISBSG dataset. Notably, the ISBSG dataset contains over 100 features, suggesting that incorporating additional relevant features could enhance the predictive models' accuracy.

It is important to note that there is currently no literature specifically addressing which software features are most effective for predicting software testing activities. Therefore, a more comprehensive analysis is required to identify the most impactful features for this purpose.

6 CONCLUSIONS AND FURTHER WORK

This empirical study explored the effectiveness of ML techniques in estimating the effort required for software testing activities within the SDLC. Four ML techniques and three heterogeneous ensembles were examined, with hyperparameters optimized using grid search. The evaluation employed the Leave-One-Out Cross-Validation (LOOCV) technique and eight unbiased performance metrics. The key findings related to each research question are summarized below:

- **(RQ1).** The KNN technique consistently outperformed the other three ML techniques across all eight performance metrics.
- **(RQ2).** Results indicated that the ensemble methods did surpass the accuracy of the individual techniques (SVR, DT, and MLP) and show less performance than KNN. This conclusion was supported by the SK test.
- **(RQ3).** All features used in training the ML techniques were identified as important; however, integrating additional features could further enhance the models' predictive capabilities.

Ongoing research is focused on exploring alternative ensemble methods, particularly homogeneous ensembles, which were not covered in this study. Efforts are also underway to improve the selection of ensemble components. Additionally, acquiring more relevant datasets for STE is a key priority, as this will contribute to the development of more robust and accurate STE models.

REFERENCES

Ajorloo, S., Jamarani, A., Kashfi, M., Kashani, M. H., and Najafizadeh, A. (2024). A systematic review of machine learning methods in software testing. *Applied Soft Computing*, page 111805.

Ali, A. and Gravino, C. (2019). A systematic literature review of software effort prediction using machine learning methods. *J. Softw. Evol. Process*, 31(10):1–25.

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *Am. Stat.*, 46(3):175–185.

Azzeh, M. and Nassif, A. B. (2013). Fuzzy model tree for early effort estimation. In *2013 12th International Conference on Machine Learning and Applications*, pages 117–121.

Azzeh, M., Nassif, A. B., and Minku, L. L. (2015). An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation. *J. Syst. Softw.*, 103:36–52.

Charette, R. N. (2005). Why software fails? *IEEE Spectr.*, 42(9):42–49.

d. A. Cabral, J. T. H., Oliveira, A. L. I., and da Silva, F. Q. B. (2023). Ensemble effort estimation: An updated and extended systematic literature review. *J. Syst. Softw.*, 195:111542.

Elish, M. O., Helmy, T., and Hussain, M. I. (2013). Empirical study of homogeneous and heterogeneous ensemble models for software development effort estimation. *Math. Probl. Eng.*, 2013.

Foss, T., Stensrud, E., Kitchenham, B., and Myrtveit, I. (2003). A simulation study of the model evaluation criterion mmre. *IEEE Trans. Softw. Eng.*, 29(11):985–995.

Hosni, M. (2023). Encoding techniques for handling categorical data in machine learning-based software development effort estimation. In *KDIR*, pages 460–467.

Hosni, M. and Idri, A. (2018). Software development effort estimation using feature selection techniques. In *Frontiers in Artificial Intelligence and Applications*, pages 439–452.

Hosni, M., Idri, A., and Abran, A. (2019a). Evaluating filter fuzzy analogy homogenous ensembles for software development effort estimation. *J. Softw. Evol. Process*, 31(2).

Hosni, M., Idri, A., and Abran, A. (2019b). Improved effort estimation of heterogeneous ensembles using filter feature selection. In *ICSOFT 2018 - Proceedings of the 13th International Conference on Software Technologies*, pages 405–412. SciTePress.

Hosni, M., Idri, A., Abran, A., and Nassif, A. B. (2018). On the value of parameter tuning in heterogeneous ensembles effort estimation. *Soft Comput.*, 22(18):5977–6010.

Idri, A., Hosni, M., and Abran, A. (2016). Systematic mapping study of ensemble effort estimation. In *Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering*, pages 132–139.

Jeffery, R., Ruhe, M., and Wieczorek, I. (2001). Using public domain metrics to estimate software development effort. In *Seventh International Software Metrics Symposium. METRICS 2001*, pages 16–27.

Labidi, T. and Sakhrawi, Z. (2023). On the value of parameter tuning in stacking ensemble model for software regression test effort estimation. *J. Supercomput.*, page 0123456789.

López-Martín, C. (2022). Machine learning techniques for software testing effort prediction. *Softw. Qual. J.*, 30(1):65–100.

Minku, L. L. and Yao, X. (2013). An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering - PROMISE '13*, pages 1–10.

Mittas, N. and Angelis, L. (2013). Ranking and clustering software cost estimation models through a multiple comparisons algorithm. *IEEE Trans. Softw. Eng.*, 39(4):537–551.

Miyazaki, Y. (1991). Method to estimate parameter values in software prediction models. *Inf. Softw. Technol.*, 33(3):239–243.

Myrtveit, I., Stensrud, E., and Shepperd, M. (2005). Reliability and validity in comparative studies of software prediction models. *IEEE Trans. Softw. Eng.*, 31(5):380–391.

Radliński, Ł. (2023). The impact of data quality on software testing effort prediction. *Electron.*, 12(7).

Shepperd, M. and MacDonell, S. (2012). Evaluating prediction systems in software project estimation. *Inf. Softw. Technol.*, 54(8):820–827.

Simon, H. (1999). *Neural networks: a comprehensive foundation*. MacMillan Publishing Company, 2nd edition.

Song, L., Minku, L. L., and Yao, X. (2013). The impact of parameter tuning on software effort estimation using learning machines. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*.

Wen, J., Li, S., Lin, Z., Hu, Y., and Huang, C. (2012). Systematic literature review of machine learning based software development effort estimation models. *Inf. Softw. Technol.*, 54(1):41–59.

Insights into the Potential of Fuzzy Systems for Medical AI Interpretability

Hafsaa Ouifak[1][a] and Ali Idri[1,2][b]
[1]*Faculty of Medical Sciences, Mohammed VI Polytechnic University, Ben Guerir, Morocco*
[2]*Software Project Management Research Team, ENSIAS, Mohammed V University, Rabat, Morocco*
{Hafsaa.ouifak, ali.idri}@um6p.ma

Keywords: Explainable AI, Interpretability, Black-Box, Machine Learning, Fuzzy Logic, Neuro-Fuzzy, Medicine.

Abstract: Machine Learning (ML) solutions have demonstrated significant improvements across various domains. However, the complete integration of ML solutions into critical fields such as medicine is facing one main challenge: interpretability. This study conducts a systematic mapping to investigate primary research focused on the application of fuzzy logic (FL) in enhancing the interpretability of ML black-box models in medical contexts. The mapping covers the period from 1994 to January 2024, resulting in 67 relevant publications from multiple digital libraries. The findings indicate that 60% of selected studies proposed new FL-based interpretability techniques, while 40% of them evaluated existing techniques. Breast cancer emerged as the most frequently studied disease using FL interpretability methods. Additionally, TSK neuro-fuzzy systems were identified as the most employed systems for enhancing interpretability. Future research should aim to address existing limitations, including the challenge of maintaining interpretability in ensemble methods

1 INTRODUCTION

With the emergence of social networks and the digital transformation of most of the aspects of our lives, data has become abundant (Yang et al., 2017). Based on this data, Machine Learning (ML) techniques can provide decision-makers with future insights and help them make informed decisions. ML techniques are now being used in various fields given engineering (Thai, 2022), industry (Bendaouia et al., 2024), medicine (Zizaan and Idri, 2023), etc.

ML techniques can be divided into two classes: white-box and black-box models. White-box models, like decision trees or linear classifiers, are transparent and easily interpretable, allowing for straightforward explanations of the knowledge they learn. On the other hand, black-box models, such as Support Vector Machines (SVMs), Random Forests, and Artificial Neural Networks (ANNs) (Loyola-Gonzalez, 2019), are not interpretable.

With the popularity of Deep Learning (DL), black box techniques have been extensively and successfully used: the more data these techniques are fed, the better their performance capabilities (Alom et al., 2019).

Despite their effectiveness, black box techniques lack an acceptable performance-interpretability tradeoff, and this represents a major obstacle to their acceptance in several domains where the cost of an error is very high and intolerable (Alom et al., 2019). For example, in the medical context, a “wrong” decision is likely to cost the life of a patient. Thus, interpretability in medicine can be used to argue the diagnosis or treatments given and makes the ML technique used trustworthy to physicians and patients.

Interpretability refers to how well humans can comprehend the reasons behind a decision made by a model (Christoph, 2020). The evaluation and assessment of interpretability techniques are challenging and sometimes left to subjectivity as it has no common interpretability measure.

A common technique to make black box techniques interpretable is to use fuzzy logic (FL). Works attempting to use FL to interpret ML black box models do so in two ways: 1) fuzzy rule extraction (Markowska-Kaczmar and Trelak, 2003), where FL is used to extract fuzzy rules explaining the behavior of the model; fuzzy rules are composed of linguistic variables that are more comprehensible to humans

[a] https://orcid.org/0000-0002-4611-6987
[b] https://orcid.org/0000-0002-4586-4158

Ouifak, H. and Idri, A.
Insights into the Potential of Fuzzy Systems for Medical AI Interpretability.

In *Proceedings of the 16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2024) - Volume 1: KDIR*, pages 525-532
ISBN: 978-989-758-716-0; ISSN: 2184-3228

(Zadeh, 1974). And 2) neuro-fuzzy systems which are used to add the interpretability aspect to ANNs while maintaining their learning and performance capabilities (de Campos Souza, 2020).

To the best of our knowledge, no Systematic Mapping Study (SMS) dealing with the use of FL in ML black box models' interpretability has been carried out for medical applications. However, there are some works related to this topic. For instance, Souza (de Campos Souza, 2020) reviewed the theory behind hybrid models, i.e., the models based on FL and ANNs, and concluded that such models present a certain degree of interpretability while maintaining a high level of performance. Similarly, Das and al. (Das et al., 2020) reviewed the improvements FL can bring to DNNs and the real-life applications of such models. Other recent studies have reviewed fuzzy interpretability to highlight its emerging trend and the promises of this field (Padrón-Tristán et al., 2021).

This study presents an SMS of the use of FL in ML interpretability for medical applications. We conducted a search on six digital libraries: IEEE Xplore, ScienceDirect, PubMed, ACM Digital Library, Wiley, and Google Scholar. The search was conducted in the period between 1994 and January 2014 and has identified 67 primary studies. The selected studies were analyzed according to four Mapping Questions (MQs):

- Publication channels and years of publications (MQ1).
- Type of presented contribution (MQ2).
- Identifying the studied medical diseases (MQ3).
- Discovering the FL categories and systems used the most by the selected papers (MQ4).

The structure of this paper is as follows: Section 2 provides an introduction to ML interpretability and FL. Section 3 outlines the research methodology used to carry out this SMS. Section 4 details the findings from the mapping study. Lastly, the conclusions are discussed in Section 5.

2 BACKGROUND

This section presents an overview of the concepts and techniques that will be referred to in this study.

2.1 Interpretability

Interpretability techniques (i.e., post-hoc or post-modeling interpretability techniques) are used to explain the behavior of certain ML models that are not intrinsically interpretable (i.e., black box) (Barredo Arrieta et al., 2020). These techniques can be classified based on their applicability and their scope. In terms of applicability, post-hoc interpretability techniques can be divided into two main groups: 1) model-agnostic methods which can be applied to any ML model (Barredo Arrieta et al., 2020). These methods work without accessing the model's internal architecture and are applied after the training (e.g. Fuzzy rule extraction (Markowska-Kaczmar and Trelak, 2003)). 2) Model-specific methods (Barredo Arrieta et al., 2020), on the other hand, rely on the internal structure of a particular model and can only explain that model (Carvalho et al., 2019) (e.g., feature relevance, visualization).

Another type of interpretability techniques classification can be done using the scope of the explanations they generate. 1) Global interpretability techniques which try to explain the whole behavior of a model; and 2) Local interpretability techniques which are only concerned with explaining the process that led the model to a particular decision (Doshi-Velez and Kim, 2017). Examples of global interpretability techniques are permuted feature importance (Fisher et al., 2018) and global surrogates (Christoph, 2020). Local interpretable model-agnostic explanations (LIME) (Barredo Arrieta et al., 2020) and SHapley Additive exPlanations (SHAP) (Lundberg et al., 2017) are two of the popular local interpretability techniques). Moreover, methods that combine a white-box and a black-box to achieve a tradeoff between performance and interpretability are referred to as hybrid architectures (e.g., neuro-fuzzy systems (Ouifak and Idri, 2023a)).

2.2 Fuzzy Inference Systems

Fuzzy inference systems (FIS) use a set of fuzzy rules to map inputs to outputs (Jang, 1993). There are two primary types of FIS: Mamdani and Takagi-Sugeno-Kang (TSK). The difference between these types occurs in the consequent part of their fuzzy rules (Zhang et al., 2020).

Mamdani FIS (Mamdani and Assilian, 1975): Developed by Mamdani for controlling a steam engine and boiler system, the Mamdani FIS follows four steps: 1) Fuzzifying the inputs, 2) Evaluating the rules (inference), 3) Aggregating the results of the rules, and 4) Defuzzifying the output. This type of FIS is often used in Linguistic Fuzzy Modeling (LFM) because of its interpretable and intuitive rule bases. For example, in a system with one input and one output, a Mamdani fuzzy rule might be structured as:

$$If\ x\ is\ A\ Then\ y\ is\ B \quad (1)$$

where x and y are linguistic variables, A and B are fuzzy sets.

TSK (Takagi-Sugeno-Kang) FIS (Sugeno and Kang, 1988): This type of FIS was introduced by Takagi, Sugeno, and Kang. It also uses fuzzy rules but differs in that the consequent part is a mathematical function of the input variables rather than a fuzzy set. For example, in a system with two inputs, a TSK fuzzy rule might be structured as:

$$If\ x\ is\ A\ and\ y\ is\ B\ then\ z\ is\ f(x,y) \tag{2}$$

where x and y are linguistic variables, A and B are fuzzy sets, and $f(x,y)$ is a linear function.

3 METHODOLOGY

Kitchenham and Charters (Kitchenham and Charters, 2007) proposed a mapping and review process consisting of six steps as shown in Figure 1. The present mapping study follows their process.

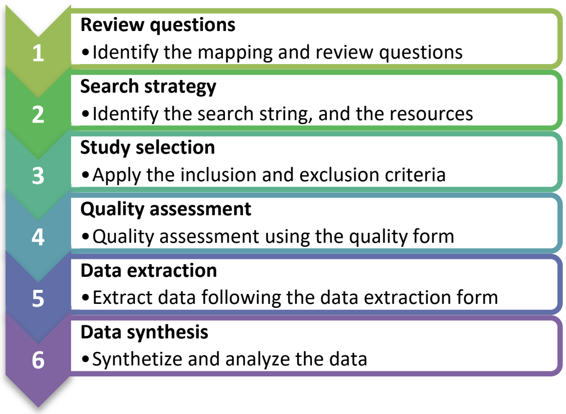

1
Review questions
•Identify the mapping and review questions
2
Search strategy
•Identify the search string, and the resources
3
Study selection
•Apply the inclusion and exclusion criteria
4
Quality assessment
•Quality assessment using the quality form
5
Data extraction
•Extract data following the data extraction form
6
Data synthesis
•Synthetize and analyze the data

Figure 1: Mapping methodology steps (Kitchenham and Charters, 2007).

3.1 Mapping Questions

The purpose of this SMS is to select and organize research works focused on using fuzzy systems to interpret ML models for medical applications. The proposed MQs for this study are outlined in Table 1.

Table 1: Mapping questions of the study.

ID	Question	Motivation
MQ1	What are the publication channels and years of publications?	To determine if there is a dedicated publication channel and to identify the number of articles discussing the use of FL in enhancing the interpretability of ML black box models for medicine over the years
MQ2	What are the types of contributions presented in the literature?	To identify the different types of studies dealing with the use of FL for ML black box models' interpretability
MQ3	What are the most studied diseases?	To find out the diseases and the medical applications that were mostly studied using the fuzzy systems to make ML decisions interpretable
MQ4	What are is the type of fuzzy systems most evaluated?	To discover the FL technique category claimed to have a better chance of enhancing the interpretability of ML black box models

3.2 Search Strategy

To address the suggested MQs, we initially created a search string and then selected six digital libraries: IEEE Xplore, ScienceDirect, ACM Digital Library, PubMed, Wiley, and Google Scholar. These libraries were frequently used in previous reviews in the field of medicine (Ouifak and Idri, 2023b; Zizaan and Idri, 2023).

3.2.1 Search String

To ensure comprehensive coverage, the search string included key terms related to the study questions along with their synonyms. Synonyms were connected using the OR Boolean operator, while the main terms were linked with the AND Boolean operator. The full search string was constructed as follows:

("black box" OR "neural networks" OR "support vector machine" OR "random forest" OR "ensemble") AND (fuzz*) AND (interpretab* OR explainab* OR “rule extraction” AND (medic* OR health*).

3.2.2 Search Process

The search process of the present SMS was based on titles, abstracts, and keywords of the primary retreived studies indexed by the six digital libraries.

3.3 Study Selection

At this point, the searches carried out returned a set of candidate studies. To further filter the candidate studies, we used a set of ICs and ECs, described in Table 2, and evaluated each one of the candidate papers based on the titles and abstracts. In case no final decision can be made based on the abstract and/or title, the full paper was reviewed.

3.4 Quality Assessment

The quality assessment (QA) phase is used to further filter high-quality papers and limit the selection. To do this, we created a questionnaire with six questions aimed at evaluating the quality of the relevant papers, as shown in Table 3.

Table 2: Inclusion and exclusion criteria.

Inclusion criteria	Exclusion criteria
Paper proposing/improving a new/existing FL-based ML interpretability technique for a medical application	Papers not written in English
Paper providing an overview of FL-based ML interpretability techniques	Unavailability of the full-text
Paper evaluating/comparing FL-based ML interpretability techniques of ML black box models	Paper using FL for any purpose other than increasing the interpretability of ML black box models
	Paper attempting to improve the interpretability of ML black box models without the use of FL

Table 3: Quality assessment form.

	Question	Possible answers
QA1	Is the FL-based ML interpretability method presented in detail?	“Yes”, “Partially” or “No”
QA2	Does the study evaluate the performance of the proposed FL-based ML interpretability technique?	“Yes”, “Partially” or “No”
QA3	Was the assessment done quantitatively or qualitatively?	“Quantitatively” or “Qualitatively”
QA4	Does the study compare the proposed technique with other techniques?	“Yes” or “No”
QA5	Does the study discuss the benefits and limitations of the proposed technique?	“Yes”, “Partially” or “No”
QA6	Is the Journal/Conference recognized?	**Conferences**: Core A: +1.5 Core B: +1 Core C: +0.5 Not ranked: +0 **Journals** : Q1 : +2 Q2 : +1.5 Q3 or Q4: +1 Not ranked: +0

3.5 Data Extraction

A data extraction form was utilized for each selected paper to answer the MQs. The extraction process was divided into two phases: initially, the first author reviewed the full texts of the studies to collect relevant data, followed by a verification step where the co-author ensured the accuracy of the extracted information.

3.6 Data Synthesis

During the data synthesis stage, the extracted data is consolidated and reported for each MQ. To simplify this process, we used the vote-counting method, and narrative synthesis to interpret the results. Then, visualization tools such as bar and pie charts, created using MS Excel were used for a better presentation.

3.7 Threats to Validity

Highlighting the study's limitations is as important as presenting its findings, enhancing reliability. Some main threats to validity in this study can be:

Study selection bias: A search string using the search string may miss some studies due to the broad scope. To address this, we set minimum criteria in the QA for objective decisions and included three possible answers to minimize disagreement (“Yes”, “Partially” and “No”).

To ensure accuracy during the data extraction phase, the results were reviewed consecutively by two authors.

4 MAPPING RESULTS

This section gives a summary of the selected articles, addresses the MQs listed in Table 1, and discusses the results of the synthesis.

4.1 Selection Process

The searches across the six selected digital libraries returned a total of 2,561 potential articles. By applying IC/EC and performing a quality assessment, we identified the papers relevant to our SMS, resulting in 67 pertinent studies, as depicted in Figure 2.

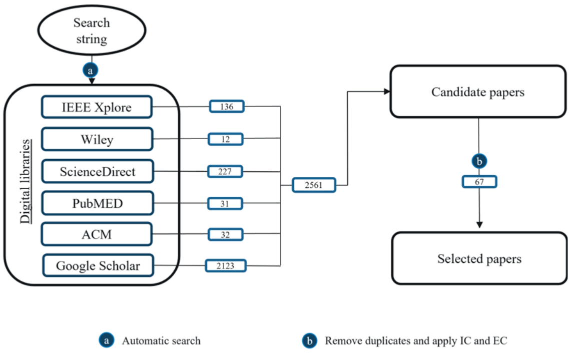


Figure 2: Papers selection steps.

4.2 MQ1: Publication Channels and Years

The 67 selected studies were distributed across journals and conferences, as depicted in Figure 3. Specifically, 67% of these papers were published in journals, and 33% in conference proceedings.

The selected papers were published in the journals IEEE Transactions on Fuzzy Systems, Expert Systems with Applications, and Applied Soft Computing, each featuring six publications. The International Conference on Fuzzy Systems (FUZZ-IEEE) was the most common conference, appearing three times among the selected papers, whereas other conferences were cited only once or twice.

The bar chart in Figure 3 shows the distribution of papers published each year from 1999 to 2023. There are several years with low numbers of publications, mostly between 2 to 4 papers, 1999 (4 papers), 2005 (3 papers), and 2006 (3 papers). A significant increase is observed starting in 2020, with 6 papers, followed by 14 papers in 2021, and peaking at 15 papers in 2022. In 2023, the number of publications decreased to 4.

The observed increase in studies focusing on the interpretability of ML black-box models using FL in 2022 may be related to the increased interest in transparency and trustworthiness in ML models. The necessity for explainable AI (XAI) has become particularly pressing in critical domains such as medicine (Chaddad et al., 2023). Consequently, researchers have been exploring various XAI approaches, with fuzzy systems being one notable avenue of investigation.

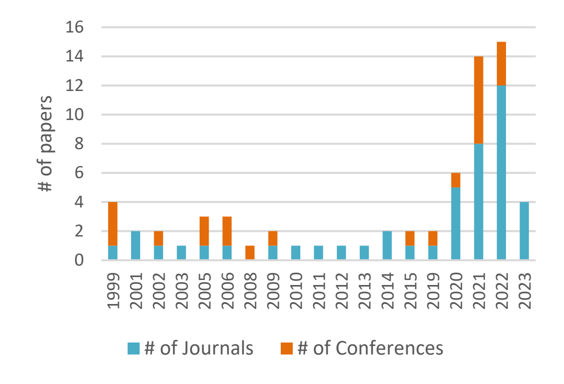


Figure 3: Distribution of the qualified studies per year and channels.

The decrease in the number of papers in 2023 can be attributed to several challenges, such as the complexity involved in training neuro-fuzzy systems for high-dimensional datasets (Ouifak and Idri, 2023a). As the rule bases expand, the rules themselves can become lengthy and difficult to interpret (Ouifak and Idri, 2023b, 2023a). Another factor may be the transparency these models offer when dealing with tabular data, where linguistic rules are more easily understood. However, many ML applications in medicine are related to medical imagery, where this clarity is less apparent. Additionally, it remains unclear to many medical professionals how FL can be integrated into their daily work. For instance, during diagnosis, patients often describe symptoms with some degree of ambiguity (e.g., 'a not strong pain,' 'a medium pain,' 'a little bit of pain'). These degrees of truth should be considered by doctors, but managing numerous symptoms with varying degrees of truth can be very complicated. A system capable of handling such fuzziness would be effective in these cases. Furthermore, there is a limited number of high quality open-source medical datasets, whether tabular or image-based, available for research (Chrimes and Kim, 2022). The lack of open data in this field can also pose a significant barrier to the evaluation of new techniques.

Contributions to FL and related systems are still evolving, but there is a need to showcase more practical applications and simplified models across different domains to maximize the potential and fully leverage the benefits of this research area.

4.3 MQ2: Type of Contributions

As shown in Figure 4, two types of contribution are identified: Solution Proposal (SP), and Evaluation Research (ER).

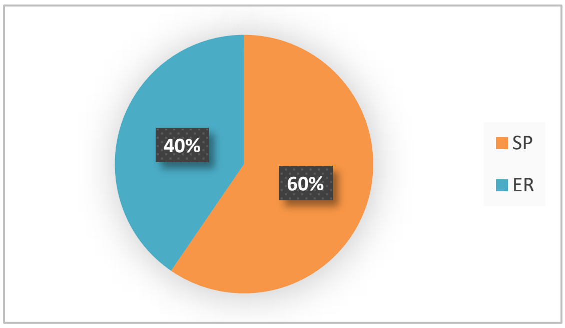


Figure 4: Type of contribution in the selected studies.

As illustrated in Figure 5, ER and SP are more prevalent compared to other types of contributions such as reviews or opinions. This indicates a significant interest in proposing and evaluating new FL-based interpretability techniques for medicine. Moreover, the prevalence of SP over evaluating existing FL techniques indicates that the field is still immature and requires further development. It's important to note that even when papers introduce a new approach, they still conduct evaluations using at least one dataset.

4.4 MQ3: Studied Diseases

The chart in Figure 5 displays the number of papers addressing different diseases. The distribution indicates a significant research focus on breast cancer and diabetes compared to other diseases. Breast cancer has the highest representation with 18 papers, followed by diabetes with 15 papers, and heart disease with 13 papers. Liver cancer and hepatitis each have 5 papers, while sleep disorder and mammography are addressed in 4 papers each. EEG signals related to bipolar disorder are discussed in 3 papers. Hypothyroid, mental health disorders, and bipolar disorder each have 2 papers. Additionally, there are 2 papers focusing on hepatobiliary disorders, Wisconsin, and Parkinson's.

Breast cancer is a significant health issue and is the leading cause of death among women worldwide (Zerouaoui and Idri, 2021). It has become a major focus in the field of ML for diagnosis, prognosis, and treatment. The importance of this topic and the availability of open-source data have contributed to its prominence in research, explaining why it is frequently studied in the selected papers.

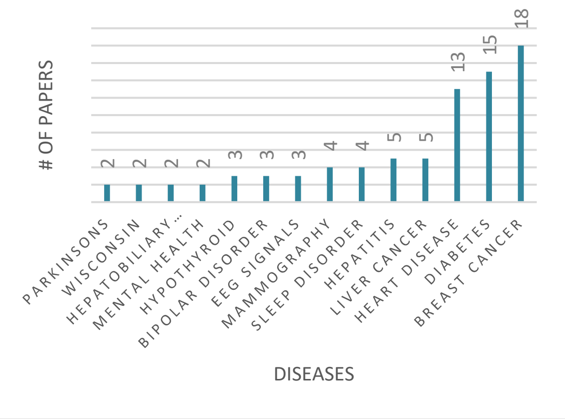


Figure 5: Most Studied Diseases.

4.5 MQ4: Types of FL Techniques

The selected studies have mainly either trained: (1) an FL-based ML model to leverage the interpretability features of FL (e.g. neuro-fuzzy systems for cancer diagnosis (Nguyen et al., 2022) or association rules for medical diagnosis based on medical records (Fernandez-Basso et al., 2022)), or (2) an ML model and then extracted FL rules from it to explain its decisions (e.g. rule extraction from SVM on lung cancer (Fung et al., 2005) or liver cancer (Chaves et al., 2005)). 14 of the selected studies used TSK fuzzy systems (e.g. (Shen et al., 2020; Zhou et al., 2021)), 9 of them specified the Mamdani category fuzzy system (e.g. (Ahmed et al., 2021; Liu et al., 2006)), while others didn't specify. Also, 36 of the papers mentioned using type-1 fuzzy systems.

32 papers used neuro-fuzzy systems and fuzzy linguistic rules (Nguyen et al., 2022) for a performance-interpretability tradeoff, while others used other techniques like the visualization (Sabol et al., 2019).

The research community has tended towards the use of the neuro-fuzzy framework. This can be explained by the fact that neuro-fuzzy networks combine both the powerful performance capabilities of ANNs and the interpretability that FL provides (Ouifak and Idri, 2023a). For example, (Nguyen et al., 2022) used the adaptive neuro-fuzzy system (ANFIS) (Jang, 1993), which is a popular model used across domains (Ouifak and Idri, 2023b). They combine fuzzy inference in a hierarchical architecture with attention to select the important rules to interpret the results of medical diagnosis. Others also used neuro-fuzzy systems for different tasks and diseases like sleep disorders (Juang et al., 2021), heart diseases (Bahani et al., 2021), and ovarian cancer (Tan et al., 2005) and showed the potential of FL system in interpreting ML rules, especially in the form of rules (Bahani et al., 2021; Chaves et al., 2005; Fung et al., 2005; Nguyen et al., 2022; Ouifak and Idri, 2023a).

5 CONCLUSION

This paper aimed to perform an SMS on the use of FL in the interpretability of ML black boxes in medicine. First, using a search string, a search was conducted in six different digital libraries. Second, a study selection process was performed, it started with identifying the papers within the scope of our SMS, and then the quality scores were computed to get only relevant papers. The study selection and quality assessment phases returned 67 relevant papers which were used to answer the MQs of this study. The main findings of each MQ are summarized below:

- MQ1. The data extracted to answer this MQ revealed that the interest in using FL to tackle the black box ML models is a hot research topic that is attracting attention once more. This was especially the case in 2022 with 15 papers. Moreover, two publication avenues were identified: journals and conferences.
- MQ2. Evaluation Research and Solution Proposal were the two main types of contributions made by the selected papers. Most of the selected papers conducted experiments and compared existing or new FL-based ML interpretability techniques.
- MQ3. Breast cancer and diabetes diseases were the most studied using FL techniques for ML interpretability.
- MQ4. Neuro-fuzzy systems specifically type-1 TSK systems are the most evaluated and studied to generate ML explanations.

Future work aims to delve deeper into neuro-fuzzy systems, which show great promise despite some limitations. One key issue is the loss of interpretability when using ensembles. To address this, we plan to develop a single rule base model that effectively represents the ensemble and maintains interpretability.

REFERENCES

Ahmed, U., Lin, J.C.W., Srivastava, G., 2021. Fuzzy Explainable Attention-based Deep Active Learning on Mental-Health Data. IEEE International Conference on Fuzzy Systems 2021-July. https://doi.org/10.1109/FUZZ45933.2021.9494423

Alom, M.Z., Taha, T.M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M.S., Hasan, M., Van Essen, B.C., Awwal, A.A.S., Asari, V.K., 2019. A state-of-the-art survey on deep learning theory and architectures. Electronics (Switzerland) 8, 292. https://doi.org/10.3390/electronics8030292

Bahani, K., Moujabbir, M., Ramdani, M., 2021. An accurate fuzzy rule-based classification systems for heart disease diagnosis. Sci Afr 14, e01019. https://doi.org/10.1016/J.SCIAF.2021.E01019

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., Herrera, F., 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion 58, 82–115. https://doi.org/10.1016/J.INFFUS.2019.12.012

Bendaouia, A., Abdelwahed, E.H., Qassimi, S., Boussetta, A., Benzakour, I., Benhayoun, A., Amar, O., Bourzeix, F., Baïna, K., Cherkaoui, M., Hasidi, O., 2024. Hybrid features extraction for the online mineral grades determination in the flotation froth using Deep Learning. Eng Appl Artif Intell 129, 107680. https://doi.org/10.1016/J.ENGAPPAI.2023.107680

Carvalho, D. V., Pereira, E.M., Cardoso, J.S., 2019. Machine Learning Interpretability: A Survey on Methods and Metrics. Electronics 2019, Vol. 8, Page 832 8, 832. https://doi.org/10.3390/ELECTRONICS8080832

Chaddad, A., Lu, Q., Li, J., Katib, Y., Kateb, R., Tanougast, C., Bouridane, A., Abdulkadir, A., 2023. Explainable, Domain-Adaptive, and Federated Artificial Intelligence in Medicine. IEEE/CAA Journal of Automatica Sinica 10, 859–876. https://doi.org/10.1109/JAS.2023.123123

Chaves, A.D.C.F., Vellasco, M.M.B.R., Tanscheit, R., 2005. Fuzzy rule extraction from support vector machines. Proceedings - HIS 2005: Fifth International Conference on Hybrid Intelligent Systems 2005, 335–340. https://doi.org/10.1109/ICHIS.2005.51

Chrimes, D., Kim, C., 2022. Review of Publically Available Health Big Data Sets. 2022 IEEE International Conference on Big Data (Big Data) 6625–6627. https://doi.org/10.1109/BIGDATA55660.2022.10020258

Christoph, M., 2020. Interpretable Machine Learning A Guide for Making Black Box Models Explainable., Book.

Das, R., Sen, S., Maulik, U., 2020. A Survey on Fuzzy Deep Neural Networks. ACM Computing Surveys (CSUR) 53. https://doi.org/10.1145/3369798

de Campos Souza, P.V., 2020. Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature. Appl Soft Comput 92, 106275. https://doi.org/10.1016/J.ASOC.2020.106275

Doshi-Velez, F., Kim, B., 2017. Towards A Rigorous Science of Interpretable Machine Learning.

Fernandez-Basso, C., Gutiérrez-Batista, K., Morcillo-Jiménez, R., Vila, M.A., Martin-Bautista, M.J., 2022. A fuzzy-based medical system for pattern mining in a distributed environment: Application to diagnostic and co-morbidity. Appl Soft Comput 122, 108870. https://doi.org/10.1016/J.ASOC.2022.108870

Fisher, A., Rudin, C., Dominici, F., 2018. Model Class Reliance: Variable Importance Measures for any Machine Learning Model Class, from the.

Fung, G., Sandilya, S., Bharat Rao, R., 2005. Rule extraction from linear support vector machines. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining 32–40. https://doi.org/10.1145/1081870.1081878

Jang, J.S.R., 1993. ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE Trans Syst Man Cybern 23, 665–685. https://doi.org/10.1109/21.256541

Juang, C.F., Wen, C.Y., Chang, K.M., Chen, Y.H., Wu, M.F., Huang, W.C., 2021. Explainable fuzzy neural network with easy-to-obtain physiological features for screening obstructive sleep apnea-hypopnea syndrome. Sleep Med 85, 280–290. https://doi.org/10.1016/J.SLEEP.2021.07.012

Kitchenham, B., Charters, S., 2007. Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, School of Computer Science and Mathematics, Keele University.

Liu, F., Ng, G.S., Quek, C., Loh, T.F., 2006. Artificial ventilation modeling using neuro-fuzzy hybrid system. IEEE International Conference on Neural Networks - Conference Proceedings 2859–2864. https://doi.org/10.1109/IJCNN.2006.247215

Loyola-Gonzalez, O., 2019. Black-box vs. White-Box: Understanding their advantages and weaknesses from a practical point of view. IEEE Access 7, 154096–154113. https://doi.org/10.1109/ACCESS.2019.2949286

Lundberg, S.M., Allen, P.G., Lee, S.-I., 2017. A Unified Approach to Interpreting Model Predictions. Adv Neural Inf Process Syst 30.

Mamdani, E.H., Assilian, S., 1975. An experiment in linguistic synthesis with a fuzzy logic controller. Int J Man Mach Stud 7, 1–13. https://doi.org/10.1016/S0020-7373(75)80002-2

Markowska-Kaczmar, U., Trelak, W., 2003. Extraction of Fuzzy Rules from Trained Neural Network Using Evolutionary Algorithm *. European Symposium on Artificial Neural Networks.

Nguyen, T.L., Kavuri, S., Park, S.Y., Lee, M., 2022. Attentive Hierarchical ANFIS with interpretability for cancer diagnostic. Expert Syst Appl 201, 117099. https://doi.org/10.1016/J.ESWA.2022.117099

Ouifak, H., Idri, A., 2023a. On the performance and interpretability of Mamdani and Takagi-Sugeno-Kang based neuro-fuzzy systems for medical diagnosis. Sci Afr e01610. https://doi.org/10.1016/J.SCIAF.2023.E01610

Ouifak, H., Idri, A., 2023b. Application of neuro-fuzzy ensembles across domains: A systematic review of the two last decades (2000–2022). Eng Appl Artif Intell 124, 106582. https://doi.org/10.1016/J.ENGAPPAI.2023.106582

Padrón-Tristán, J.F., Cruz-Reyes, L., Espín-Andrade, R.A., Llorente-Peralta, C.E., 2021. A Brief Review of Performance and Interpretability in Fuzzy Inference Systems. Studies in Computational Intelligence 966, 237–266. https://doi.org/10.1007/978-3-030-71115-3_11/TABLES/6

Sabol, P., Sincak, P., Ogawa, K., Hartono, P., 2019. Explainable Classifier Supporting Decision-making for Breast Cancer Diagnosis from Histopathological Images. Proceedings of the International Joint Conference on Neural Networks 2019-July. https://doi.org/10.1109/IJCNN.2019.8852070

Shen, T., Wang, J., Gou, C., Wang, F.Y., 2020. Hierarchical Fused Model with Deep Learning and Type-2 Fuzzy Learning for Breast Cancer Diagnosis. IEEE Transactions on Fuzzy Systems 28, 3204–3218. https://doi.org/10.1109/TFUZZ.2020.3013681

Sugeno, M., Kang, G.T., 1988. Structure identification of fuzzy model. Fuzzy Sets Syst 28, 15–33. https://doi.org/10.1016/0165-0114(88)90113-3

Tan, T.Z., Quek, C., Ng, G.S., 2005. Ovarian cancer diagnosis by hippocampus and neocortex-inspired learning memory structures. Neural Netw 18, 818–825. https://doi.org/10.1016/J.NEUNET.2005.06.027

Thai, H.T., 2022. Machine learning for structural engineering: A state-of-the-art review. Structures 38, 448–491. https://doi.org/10.1016/J.ISTRUC.2022.02.003

Yang, C., Huang, Q., Li, Z., Liu, K., Hu, F., 2017. Big Data and cloud computing: innovation opportunities and challenges. Int J Digit Earth 10, 13–53. https://doi.org/10.1080/17538947.2016.1239771

Zadeh, L.A., 1974. The Concept of a Linguistic Variable and its Application to Approximate Reasoning. Learning Systems and Intelligent Robots 1–10. https://doi.org/10.1007/978-1-4684-2106-4_1

Zerouaoui, H., Idri, A., 2021. Reviewing Machine Learning and Image Processing Based Decision-Making Systems for Breast Cancer Imaging. J Med Syst 45, 1–20. https://doi.org/10.1007/S10916-020-01689-1/FIGURES/19

Zhang, S., Sakulyeva, T.N., Pitukhin, E.A., Doguchaeva, S.M., Zhang, S., Sakulyeva, T.N., Pitukhin, E.A., Doguchaeva, S.M., 2020. Neuro-Fuzzy and Soft Computing - A Computational Approach to Learning and Artificial Intelligence. International Review of Automatic Control (IREACO) 13, 191–199. https://doi.org/10.15866/IREACO.V13I4.19179

Zhou, T., Zhou, Y., Gao, S., 2021. Quantitative-integration-based TSK fuzzy classification through improving the consistency of multi-hierarchical structure. Appl Soft Comput 106, 107350. https://doi.org/10.1016/J.ASOC.2021.107350

Zizaan, A., Idri, A., 2023. Machine learning based Breast Cancer screening: trends, challenges, and opportunities. Comput Methods Biomech Biomed Eng Imaging Vis 11, 976–996. https://doi.org/10.1080/21681163.2023.2172615

AUTHOR INDEX

Proceedings of IC3K 2024 - Volume 1: KDIR

16th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management

https://ic3k.scitevents.org

EVENT MANAGEMENT SYSTEM:

LOGISTICS:

IN COOPERATION WITH:

PROCEEDINGS WILL BE SUBMITTED FOR EVALUATION FOR INDEXING BY:

ISSN: 2184-3228
ISBN: 978-989-758-716-0