UNIVERSITY OF L'AQUILA
DEPARTMENT OF INFORMATION ENGINEERING,
COMPUTER SCIENCE AND MATHEMATICS

DOCTORAL THESIS

Trustworthy AI Through Principled Advances in Machine Unlearning, Algorithmic Bias Mitigation, and Explainability

---

Author:
Andrea D'ANGELO

Advisor:
Prof. Giovanni STILO

Co-Advisor:
Prof. Antinisca DI MARCO

Doctoral Program Coordinator:
Prof. Davide Di Ruscio

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in

Information and Communication Technology
Emerging computing models, software architectures, and intelligent systems
XXXVIII Cycle
SSD INF/01
A.Y. 2024/2025

"Although many believe that technology automatically enables more realistic expression, I believe that is just not correct."

Satoru Iwata

UNIVERSITY OF L'AQUILA
DEPARTMENT OF INFORMATION ENGINEERING, COMPUTER SCIENCE
AND MATHEMATICS

Abstract

Information and Communication Technology
Emerging computing models, software architectures,
and intelligent systems
XXXVIII Cycle
SSD INF/01
A.Y. 2024/2025

Doctor of Philosophy

Trustworthy AI Through Principled Advances in Machine Unlearning, Algorithmic Bias Mitigation, and Explainability

by Andrea D'ANGELO

In an era of unprecedented advances in Machine Learning and Artificial Intelligence, concerns are rising about the *trustworthiness* of these systems. **Trustworthy AI** is a multidimensional concept, which the EU High-Level Expert Group on Artificial Intelligence characterizes through three key pillars: **Lawfulness**, referring to the legal compliance and regulatory alignment of AI models; **Ethics**, concerning their adherence to ethical principles such as fairness and accountability; and **Robustness**, denoting their reliability, resilience, and safety under real-world conditions, including distributional shifts and evolving deployment environments.

In this thesis, we introduce principled advances across all three dimensions of Trustworthy AI. For Lawfulness, we rationalize and extend the growing literature on Machine Unlearning, proposing a unified and easily extensible framework, called ERASURE, from which we derive a more coherent theoretical and empirical framing, alongside a state of the art advancement through the study of an orthogonal problem called Forget Set Identification. For Ethics, we conduct an empirical investigation into data fairness in Software engineering, identifying structural sources of bias, and we introduce a state of the art debiasing method that substantially mitigates bias at the data level. Finally, for Robustness, we examine dense information retrieval systems powered by large language models, proposing a technique, called DbU-Cloud, that enhances their explainability while preserving or improving retrieval performance.

Collectively, these contributions represent a significant advancement toward building legally compliant, privacy-preserving, and trustworthy AI systems.

Contents

List of Figures

List of Tables

To Vincenzo

Chapter 1

Introduction

The growing adoption of machine learning (ML) models across various industries has raised several concerns about their everyday use, especially in sensitive applications. In particular, the literature recognizes several problematic scenarios, including the use of potentially sensitive data in model training [113] and the consequent challenge of ensuring regulatory compliance [91, 174], as well as issues related to implicit bias [7, 25], and explainability [6].

In an era marked by rapid advancements in artificial intelligence (AI), the **trustworthiness** of these systems has not always kept pace with their development and widespread deployment. In response, both the research community and policymakers have intensified their efforts to address the technical challenges associated with the responsible design, implementation, and governance of these technologies.

A prominent attempt to formalize this notion is offered by the European Commission's High-Level Expert Group on Artificial Intelligence (AI HLEG), which, in its Ethics Guidelines for Trustworthy AI [89], defines **trustworthy AI** via three trustworthiness dimensions (TDs):

- *Lawfulness* — compliance with all applicable laws and regulations;
- *Ethics* — adherence to ethical principles and values; and
- *Robustness* — technical and social robustness to drifts and deployment contexts;

This definition serves as the foundation for the recent European AI Act [91], which translates these principles into a binding regulatory framework for the development of AI. Comparable initiatives have also emerged globally, including the OECD's Principles on Artificial Intelligence [198], and UNESCO's Recommendation on the Ethics of Artificial Intelligence [249]. Moreover, independent research has also formulated similar principles that a trustworthy AI should follow [102]. Although these frameworks differ in terminology, their core values can easily be mapped to the three TDs defined by the AI HLEG: lawfulness, ethics, and robustness.

Despite these efforts to define and regulate trustworthy AI, implementing its principles in practice remains an open challenge. As [186] claims, principles are not enough to guide trustworthy AI: concrete methods are needed to translate aspirations into systems that can be trusted in real-world contexts.

In this thesis, we adopt the AI HLEG's definition of trustworthiness and analyze all three of its dimensions, addressing open challenges within each. In particular, for ***(i). Lawfulness***, we analyze how ML systems comply with the *right to be forgotten*, as established by the AI Act [91] and the General Data Protection Regulation (GDPR) [255], which mandates that an individual's data must be removed upon request [174]. To address this issue, the field of **machine unlearning** has emerged,

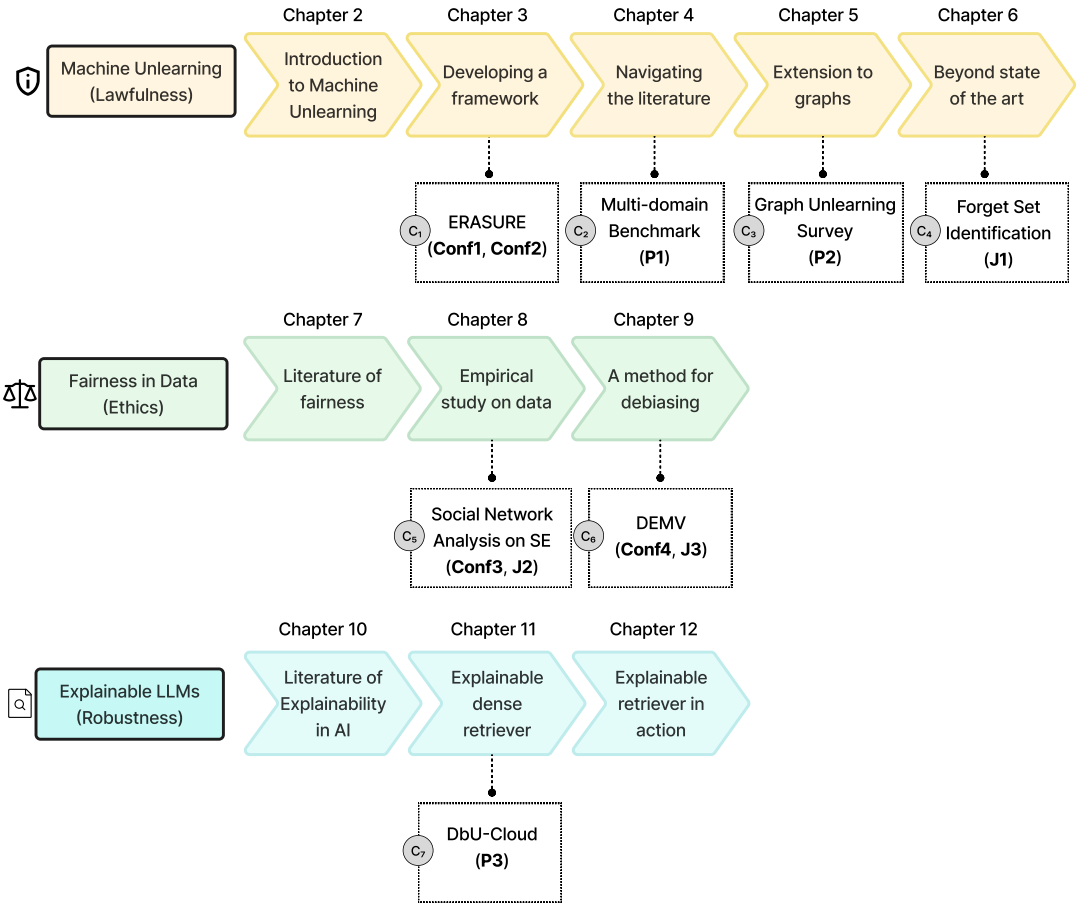


FIGURE 1.1: Structure of this thesis. We present principled advances across all TDs of trustworthy AI. The research process for each dimension is depicted in the arrow steps, one for each Chapter. Where applicable, we denote our contribution, enumerated with C_i. Each contribution is linked to one or more publications (in bold) among those listed in Section 1.2.

with the aim of efficiently removing the influence of specific data points from trained models. For ***(ii). Ethics***, we tackle the issue of *bias and fairness* within ML systems from a data-centric perspective: we study how structured and unstructured data can both present hidden biases, and propose a strategy for identifying and mitigating their effects. Lastly, for ***(iii). Robustness***, we discuss how *explainability* is cardinal to trust blackbox systems and we propose a more explainable method for Large Language Model (LLM)-based dense retrieval.

1.1 Thesis Contributions

Figure 1.1 shows how each TD is addressed through a dedicated line of investigation, outlining the methodological steps taken. Each step is described in the corresponding Chapter, and linked to the corresponding contribution (C_i) achieved, where applicable. Moreover, each contribution refers to one or more publications among those listed in Section 1.2.

Compared to other TDs, existing research on the *Lawfulness* dimension is still at an early stage. The most active and promising line of work within this dimension is *machine unlearning*, which has a rapid surge in popularity in recent years. Because of

its novelty, however, most existing works remain fragmented, with reproducibility packages often duplicated, inconsistent, or incomplete. Recognizing this gap, our first contribution in this area is **ERASURE (C_1)**, a unified and extensible framework for machine unlearning evaluation, described in Chapter 3. Exploiting this framework, we are able to rationalize, evaluate and organize the methods in the literature through a comprehensive **Multi-domain Benchmark (C_2)**, reported in Chapter 4. The benchmark covers the domains of Image, Text, Tabular, and graph-level classification. However, the extension to the node/edge-level classification scenario is more complex. To address this complexity, we survey all machine unlearning methods on transductive graphs and describe the results in our **Graph Unlearning Survey (C_3)** in Chapter 5. Lastly, we go beyond the state of the art with a novel problem formulation, aimed at finding the best set of samples to remove in order to forget a certain model behavior. We call this the **Forget Set Identification** (C_4) problem and we formalize it, prove its NP-Hardness, propose a linear programming solution, and show results in Section 6.

On the other hand, work on fairness and explainability in AI has reached a much higher level of maturity. They represent the research directions we follow for the TDs of Ethics and Robustness, respectively.

There are many definitions of fairness in the literature, so we first establish our ground terminology before moving on to an empirical study on how bias can be inherent in structured data like graph. Our **Social Network Analysis on the Software Engineering Community** (C_5) is described in detail in Chapter 8. Following on this line of research, we present a model-agnostic method to remove biases at the data level, namely **DEMV** (C_6), in Chapter 9.

For Robustness, we first analyze the literature of explainability in AI. We identify a GAP for explainable dense retrievers in the context of Information Retrieval and Retrieval-Augmented Generation (RAG). To fill this gap, we propose **DbU-Cloud** (C_7), a novel dense retriever that is more explainable than the state of the art. We demonstrate its superior effectiveness in terms of retrieval metrics and with anecdotal examples in Chapter 11 .

Next, we present the detailed contributions of the thesis.

1.1.1 List of Contributions

Dashed boxes in Figure 1.1 represent the contributions of this thesis in the three dimensions of trustworthy AI. In the following, we outline each contribution and provide a brief overview of its scope and goals.

Our research is strongly focused on reproducibility. For this reason, where available, contributions are annotated with repositories containing all code and experiments.

C_1. *ERASURE: A unified framework for machine unlearning*[72, 73]

By studying the literature on machine unlearning (MU), we identified a critical gap: despite its growing relevance, the field lacks a widely accepted reference framework. Researchers and practitioners rely on ad-hoc implementations, hindering the reproducibility and comparability of results [103, 110]. The absence of standardized tools and benchmarks makes consistently evaluating unlearning methods hard [279]. To address this gap, we introduce **ERASURE**, a flexible and modular framework designed to support research in MU.

ERASURE simplifies experimentation by organizing all settings within a single JSON configuration file: dataset, model, unlearning method, and evaluation measures. ERASURE offers built-in implementations of many state of the art unlearning techniques, ranging from retraining-based approaches to more efficient approximate methods. In this thesis, we demonstrate how ERASURE is both easily extensible and customizable, and we present the design patterns that enable this flexibility.

Code: https://github.com/aiim-research/ERASURE

C_2. *A multi-domain Benchmark of machine unlearning for classification tasks*

Building on the ERASURE framework, we continue the effort of rationalizing the MU literature by presenting a benchmark on machine unlearning methods applied to classification tasks. Specifically, our contributions are the following: **(i).** We introduce the Laplacian Unlearning Multidimensional Assessment (LUMA), a unified metric that jointly captures the three key aspects of MU: utility, efficacy, and efficiency. **(ii).** We validate LUMA on a benchmark of 12 machine unlearning methods on 8 classification datasets, including the first benchmarks for tabular and graph data. **(iii).** We propose a taxonomy of the 12 benchmarked methods to aid in structuring and understanding the field. **(iv).** The benchmark we provide publicly is reproducible, easily extensible, and intended to serve as a foundation for future research in MU.

C_3. *A Survey of Graph Unlearning*

The graph-level scenario of graph classification falls under the benchmark presented as C_2. In contrast, the node/edge-level classification scenario is inherently more complex. The vast majority of the literature on graph unlearning focuses on the latter scenario [43], with a plethora of methods claiming to define their own unlearning framework [262, 48, 263, 82]. To assess this complexity, we survey all GU methods in the literature, rationalizing their approaches under a common framework and formalizing a taxonomy of GU methods. We highlight duplicate efforts, weaknesses, and future directions for the field.

C_4. *The Forget Set identification problem* [74]

After surveying and rationalizing the literature, proposing a unified framework, and extending such framework to graph data, our fourth and last contribution to the field of machine unlearning is to go beyond the state of the art and propose a novel problem on finding the best *forget set* to remove a certain behavior from the model. The forget set, in the context of MU, is the set of samples whose influence must be removed, and MU methods assume to have it as input. However, the requirement of having the forget set as input is so restrictive that it might prevent MU from being profitably applied in many real-world scenarios. For instance, individuals requesting data deletion may lack sufficient technical expertise to define the forget set by themselves [255]. In this work, we fill this important gap, and define the Forget Set identification problem: find the forget set such that its removal from the original training set ensures unlearning of a model behavior, specified by the user. This problem is an important complementary problem to MU, to be used coupled with (and not in substitution for) MU methods. In this thesis, we formalize the problem and prove its NP-Hardness. We propose an algorithm based on the Red-blue set cover, and we provide extensive experimentation on different settings and datasets.

Code: https://github.com/dangeloandrea14/Forget_Set_Identification

C_5. *Social Network Analysis on the Software Engineering Community* [9]

Our first contribution in the *ethics* dimension through fairness in data is an empirical study through co-authorship graphs in the Software Engineering community, uncovering hidden biases in gender distribution and behaviour. The relevance of this topic is also highlighted by international programs such as the 2030 United Nations Sustainability Development Goals [250] or European Union programs such as EUGAIN, which aims to improve gender balance in Informatics at all levels [30]. To achieve this goal, we collect all the needed data from several open repositories and process them with data mining techniques to make them suitable for analysis. We conduct a comprehensive analysis with a dual focus, comparing the Italian SE community with the worldwide SE community and the Italian Informatics community, analyzing both co-authorship relationships and the scenarios of academic promotions.

Code: https://github.com/dangeloandrea14/public_SE_fairness

C_6. *Debiaser for Multiple Variables* [67, 66]

After analyzing how bias can be hidden in structured data in C_5, we move our efforts to the development of a method to remove the influence of bias from data. To this aim, we present the Debiaser for Multiple Variables (**DEMV**), an approach able to mitigate unbalanced groups bias (i.e., bias caused by an unequal distribution of instances in the population) in both binary and multiclass classification problems with multiple sensitive variables. The proposed method is compared with a set of well-established baselines. At first, we conduct a specific study to understand which is the best generation strategies and their impact on DEMV's ability to improve fairness. Then, we evaluate our method on a heterogeneous set of datasets and we show how it overcomes the established algorithms of the literature.

Code: https://github.com/giordanoDaloisio/demv

C_7. *DbU-Cloud* [71]

Our last contribution relates to the third TD, robustness. In the context of explainable AI, we propose a novel method for dense retrieval, that is more explainable than the baselines methods in the literature. Employing LLMs for Document Ranking involves the pooling of term embeddings into a single sentence embedding, which can represent either a document or a query. With this approach, the scoring function is determined by measuring the similarity between the mean dense embeddings of the query and a document, commonly using metrics like cosine similarity or the inner dot product. However, collapsing the term embeddings by pooling them into a single sentence embedding might result in undesirable ranking results and degraded effectiveness. To address these issues, we propose DbU-Cloud, a novel unsupervised method for Document Ranking with LLMs. This method does not employ a pooling layer for the embeddings, opting instead to compute a density-based Query-Document Relevance Score on the respective sets of embeddings. DbU-Cloud rewards not only the proximity of the two sets but their relative density as well. Moreover, as an unsupervised method, it does not require further fine-tuning of the employed LLMs.

1.2 Publications

All the contributions presented in this thesis are based on work published or submitted to peer-reviewed conferences and journals. The complete list of journal, conference, and workshop publications is given below in chronological order. Where available, publications are annotated with the respective journal rank (from Scimago Journal Rank (SJR)[1]) or conference rank (from CORE [2]).

Journal Papers

J1. **[Q1] D'Angelo, A.**, Gullo, F., & Stilo, G. (2025). The forget-set identification problem. Machine Learning. [74]

This paper presents the Forget Set problem presented in Chapter 6.

J2. **[Q1] D'Angelo, A.**, d'Aloisio, G., Marzi, F., Marco, A.D., & Stilo, G. (2024). Uncovering gender gap in academia: A comprehensive analysis within the software engineering community. J. Syst. Softw., 217, 112162. [9]

This paper presents the empirical study of the Software Engineering community of Chapter 8.

J3. **[Q1]** d'Aloisio, G., **D'Angelo, A.**, Marco, A.D., & Stilo, G. (2023). Debiaser for Multiple Variables to enhance fairness in classification tasks. Inf. Process. Manag., 60, 103226. [66]

This paper presents the debiasing method of Chapter 9.

Conference Papers

Conf1. **[A*] D'Angelo, A.**, Savelli, C., Tagliente, G., Giobergia, F., Baralis, E., & Stilo, G. (2024). How to Make Reproducible Research in Machine Unlearning with ERASURE. International Joint Conference on Artificial Intelligence. [73]

This paper describes the technical implementation of ERASURE (Chapter 3).

Conf2. **[A] D'Angelo, A.**, Savelli, C., Tagliente, G., Giobergia, F., Baralis, E., & Stilo, G. (2025). ERASURE: a Modular and Extensible Framework for Machine Unlearning. Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM 2025), 2025. [72]

This paper presents the design patterns and rationale behind ERASURE (Chapter 3).

Conf3. **[A]** d'Aloisio, G., **D'Angelo, A.**, Marzi, F., Di Marco, D., Stilo, G., & Di Marco, A. (2023). Data-driven analysis of gender fairness in the software engineering academic landscape. In Proceedings of the European Conference on Software Architecture (ECSA 2023). [65]

This paper presents an early result of the bigger social network analysis presented in Chapter 8.

Conf4. d'Aloisio, G., Stilo, G., Marco, A.D., & **D'Angelo, A.** (2022). Enhancing Fairness in Classification Tasks with Multiple Variables: A Data- and Model-Agnostic Approach. International Workshop on Algorithmic Bias in Search and Recommendation. [67]

[1] https://www.scimagojr.com/

[2] https://portal.core.edu.au/conf-ranks/

This paper presents an introductory study and early implementation of DEMV, described in Chapter 9.

Preprints

P1. **D'Angelo, A.**, Savelli, C., Giobergia, F., Baralis, E., & Stilo, G. (2025), A Multi-domain Benchmark for Machine Unlearning in Classification Tasks.

This paper presents the MU benchmark of Chapter 4.

P2. **D'Angelo, A.**, Mottin, D., Gullo, F., Stilo, G. (2025), A survey on Graph Unlearning.

This paper presents the Graph Unlearning survey of Chapter 5.

P3. **D'Angelo, A.**, Stilo, G., & Di Marco, A. (2024). A novel relevance score for unsupervised retrieval with large language models. Paper presented at the KDD 2024 Conference Ph.D. Consortium. [71]

In this paper, we introduced DbU-Cloud for the first time, as illustrated in Chapter 11.

Authored papers not included in this thesis

J4. **[Q1]**Bianchi, A., Zelli, V., **D'Angelo, A**., Di Matteo, A., Scoccia, G., Cannita, K., Dimas, A.S., Glentis, S., Zazzeroni, F., Alesse, E., Di Marco, A., & Tessitore, A. (2024). A method to comprehensively identify germline SNVs, INDELs and CNVs from whole exome sequencing data of BRCA1/2 negative breast cancer patients. NAR Genomics and Bioinformatics, 6. [20]

J5. **[Q1]** Della Penna, G., Orefice, S., & **D'Angelo, A.** (2023). Exploiting spatial relations for grammar-based specification of multidimensional languages. Knowledge and Information Systems, 65, 3995-4020. [77]

Conf5. Bianchi, A., d'Aloisio, G., **D'Angelo, A.**, Di Marco, A., Di Matteo, A., Leone, J., Scoccia, G., Stilo, G., & Traini, L. (2022, September 20). DIORAMA: Digital twin for sustainable territorial management. In Proceedings of itaDATA 2022 Conference.

Conf6. **[A] D'Angelo, A.**, Di Sipio, C., Politowski, C., & Rubei, R. (2024). PlayMyData: a curated dataset of multi-platform video games. 2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR), 525-529.

Conf7. **[B] D'Angelo, A.**, & d'Aloisio, G. (2024). Grammar-Based Anomaly Detection of Microservice Systems Execution Traces. Companion of the 15th ACM/SPEC International Conference on Performance Engineering.

Conf8. **[C]** Di Sipio, C., **D'Angelo, A.**, Rubei, R., & Politowski, C. (2025, August 26). On the need for reproducibility guidelines for open-source games: An itch.io case study. In Proceedings of the 2025 IEEE Conference on Games (CoG) (pp. 1–8).

1.3 Thesis Structure

This thesis is divided into three parts, following the partitioning of Trustworthy AI into three corresponding TDs. Part 1 is dedicated to our contributions in the field of machine unlearning for *lawfulness*. Part 2 follows our contributions to the TD of *ethics* through our works in the inherent biases of data and how they impact model fairness. Lastly, Part 3 details our proposed contribution to explainable dense retrievers for the TD of *robustness*.

Part I

Machine Unlearning for Legal Compliance and Privacy Preservation

Chapter 2

Introduction to Machine Unlearning

As machine learning (ML) models become more widely integrated across industries, concerns have emerged about the use of sensitive data in their training [113].

These concerns have implications across a wide range of applications and technologies, including privacy-preserving Machine Learning, legal compliance, and data poisoning. The field of *Machine Unlearning* (MU) has recently emerged as a solution to many of these concerns. Simply put, we sometimes want our trained ML model to *unlearn* some of the knowledge they have acquired as part of the training set. This can be done by retraining the model from scratch, but this solution incurs huge expenses, both in terms of time and money, for huge models. Consequently, the goal of MU methods is to obtain a model similar to the one we would obtain by retraining, more efficiently.

In this Section, we first describe the motivations behind Machine Unlearning, answering the often asked question: *why would I want my model to unlearn something?* Then, we formally define the task of Machine Unlearning and its workflow. We explore related works in the rapidly evolving literature, and outline the contribution of this thesis to the topic.

2.1 Motivation

Although the task of MU is counter-intuitive with respect to Machine Learning, it draws motivations from several areas and for different reasons. While the main goal for most MU methods at this stage is legal compliance, several use cases are also found in privacy preservation and data poisoning. In general, a trustworthy AI system needs to be able to unlearn sensitive or outdated data, and be adaptive with respect to evolving knowledge bases.

Legal compliance. Recent regulations like the AI Act [91] and the General Data Protection Regulation (GDPR) [255] established the *right to be forgotten*, mandating that an individual's data must be removed upon request [174].

In such cases, the model's owner must release a new version of the model itself, specifically excluding those targeted samples from the training set. However, retraining ML models from scratch upon every request is often too expensive in terms of time, money, and environmental costs [59]. *Machine unlearning* (MU) offers a promising alternative: instead of (re)training the model from scratch, MU aims to efficiently remove the influence of specific data points from an already trained model [265, 155].

For large models, this might be the only option to stay legally compliant with the most recent regulations, given the impossibility of retraining from scratch after each unlearning request.

A second point in the context of legal compliance is the misuse of copyrighted material. Recently, several AI systems have been trained on copyrighted artworks, resulting in generated images that are stylistically very similar to intellectual properties [74, 131]. The most popular example of this phenomenon was OpenAI's ChatGPT generating images similar to Studio Ghibli's signature art style at the user's request [1]. However, this issue can be generalized for any copyrighted property or piece of art executed by a human artist and used as training data for an ML system.

This issue can be tackled via MU in a similar way to the unlearning requests produced via the right to be forgotten: we can remove the influence of an image or a set of images from the training set without retraining the model from scratch.

Privacy-preserving ML. Preserving privacy in ML systems has gained significant importance in recent years. The textbook example to understand the significance of this field is the Enron dataset [141], a large collection of e-mails sent between Enron employees. Because of legal actions moved towards the company, the dataset was released without anonymization.

An empirical study [124] found that LLMs that are trained or fine-tuned on the Enron dataset can reveal information about those names when prompted correctly. This shows that data leakage from generative models is possibly exploitable from attackers to gain knowledge on the training set.

Machine Unlearning helps maintaining privacy by inducing the model to unlearn particularly sensitive data. For instance, by running MU methods on features or specific samples, we can make the model unlearn specific information (as shown in our benchmark, detailed in Section 4).

The intersection between the fields of MU and privacy preservation are vast. For instance, a subset of MU methods, called "Certified Machine Unlearning" methods, derive a formulation of model proximity that is very close, syntactically, to differential privacy [85]. We characterize them more in detail in Section 5, where we propose a taxonomy and rationalization of MU methods for graphs.

Moreover, both privacy preserving and MU are being extensively researched for their application on Federated Learning systems. While this is out of scope for the thesis, we acknowledge it as an interesting avenue of research.

In general, preserving privacy through multiple mechanisms like differential privacy and machine unlearning is necessary to build trustworthy AI systems that evolve over time and are resilient to attacks like Membership inference attacks [230].

Data poisoning. Data poisoning is a malicious attack aimed at hindering the utility of a model by injecting malicious samples in the training set. It is extensively studied [232] and it has also been shown that, in order to compromise the utility of an LLM, we only require a very limited amount of samples (in constant rate) [233]. If the data poisoning attack succeeds, the model's utility will be destroyed because of a few samples.

This effect can possibly be reverted with a MU method without having to retrain from scratch, by simply unlearning the influence of the poisoned samples. Clearly, this approach assumes that the poisoned samples are known beforehand. All MU methods in the literature hold this same assumption. We challenge it and go beyond the state of the art, describing the problem of defining a forget set, in Section 6, where we present the Forget Set Identification problem.

[1] https://www.nytimes.com/2025/03/27/style/ai-chatgpt-studio-ghibli.html

2.2 Definition and Workflow

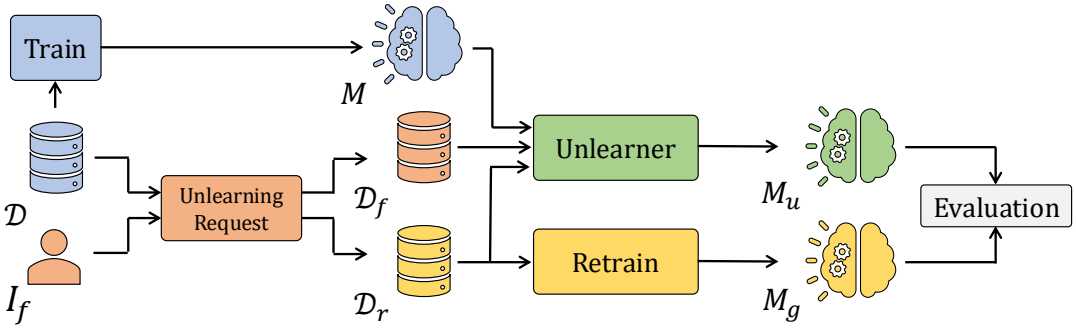


FIGURE 2.1: Machine Unlearning Workflow.

Now that we are aware of the motivation behind the rise in popularity of the field of Machine Unlearning, we proceed to a formal definition and an overview of its workflow.

Let M be a ML model parametrized on the parameter space Θ, and let $\mathcal{H}$ be the hypothesis space $\mathcal{H} : \{M_\theta : \theta \in \Theta\}$, where M_θ is the ML model parametrized by θ.

Let D be a dataset, $D_{train} \subset D$ the training set. Then, let A be a training algorithm:

$$A : (D_{train}) \rightarrow M_{\theta^*} \in \mathcal{H}$$

that learns the optimal model parameters $\theta^* \in \Theta$ minimizing a given empirical loss over the set D_{train}.

Machine Unlearning

We formally define a Machine Unlearning method as a function

$$U : \mathcal{H} \times D \rightarrow \mathcal{H}$$

that, given as input the trained model M_θ and the forget set D_f (and the corresponding retain set $D_r = D \setminus D_f$), outputs an updated model M_{θ_u} such that:

$$U(M_\theta, D_f) = M_{\theta_u} \approx M_G,$$

where $M_G = A(D_r)$ denotes the Gold Model retrained from scratch on the retain set.

The aim of a MU method is always to approximate the *Gold Model* as close as possible. The Gold Model represents the ideal solution: the model retrained from scratch on the training set without the data from the forget set. Clearly, while this is the optimal solution of the task, it is also too expensive to obtain. As such, a MU task approximates the Gold Model while being as efficient as possible.

An important note is that the aim of a MU method is not to destroy the model's utility on the forget set. This would open the model to Membership inference attacks [37], and privacy issues in general. Instead, the goal is to match the utility of the Gold Model on the forget set.

Fig. 2.1 illustrates a typical workflow in MU. In this setting, an unlearning request is issued by an entity I_f, which triggers the partitioning of the original dataset $\mathcal{D}$ into a Forget Set $\mathcal{D}_f$ and a Retain Set $\mathcal{D}_r$, containing respectively the data samples

to be removed and those to be preserved. An unlearning algorithm (Unlearner) operates on the original model M, which was trained on $\mathcal{D}$. Using both $\mathcal{D}_f$ and $\mathcal{D}_r$, it produces an updated model M_u in which the influence of $\mathcal{D}_f$ should be removed. The objective is to make M_u as similar as possible to the Gold Model M_g, which is trained from scratch using only $\mathcal{D}_r$. The similarity between M_u and M_g is typically assessed using a set of measures that capture utility, efficacy, and efficiency, as detailed in [121]: *utility* – how well the model performs after unlearning – *efficacy* – the degree to which the target data is removed – and *efficiency* – the computational cost of unlearning w.r.t. retraining the model from scratch.

2.3 Related Work on Machine Unlearning

Since its introduction [35], MU has been a very active area of research. A plethora of different MU approaches have been devised, either exact or approximate, and adhering to various principles, including data reorganization and model manipulation [265]. There are also recent works that investigate how the properties of the forget set make the MU process more or less difficult [94, 279].

Variants of MU include the *selective forgetting* problem [229], whose goal is to forget one or more entire classes from a trained model, rather than individual samples. This is a goal that is also at the basis of other slightly different, but still class-level forgetting problems [153, 242, 270]. *Model editing* (or *knowledge editing*) has recently emerged in the large language model (LLM) landscape as the problem of modifying an LLM to incorporate specific knowledge, without negatively influencing other irrelevant knowledge [257].

Part of the contribution of this thesis for the task of MU is to rationalize the field through a unified framework, a benchmark, and a taxonomy on graph unlearning methods. For this reason, a deeper dive on the methods currently available in the literature is presented in Sections 4 and 5.

Frameworks. MU has received an increasing amount of research attention in recent years, leading to fast development of the field. Because of this, however, researchers have been coding their own MU methods in individual and separate implementations, duplicating efforts and not aligning to a single reference framework.

ERASURE, which we present in Section 3, is a unified and extensible framework that aims to give researchers and practitioners a standardized environment to test MU methods and develop new ones. At the time of release, ERASURE was one of the first frameworks for MU. At the same time or later, other frameworks were also released, like MUBox [158], which also provides a large number of MU methods out-of-the-box, similarly to ERASURE. However, ERASURE's strength lies in its extensibility and adaptability: via a single JSON configuration file, users can implement any unlearning scenario, with any dataset of choice, and even implement their own custom preprocessing functions by simply adapting their logic to ERASURE's interfaces.

Several other frameworks, like OFMU [12] specialize in a subset of MU (in OFMU's case, optimization-driven unlearning for LLMs). To the best of our knowledge, ERASURE and MUBox are the only general-purpose frameworks available to date.

Surveys and Benchmarks. Despite increasing interest in MU, most existing codebases are limited to reproducing methods introduced in individual research papers. These implementations are often related to a particular experimental setup and lack modularity, making it difficult to adapt them to new datasets, models, or evaluation protocols.

Recent benchmark works have released partial code demonstrating the proposed unlearning strategies. For instance, [132], [173], and [228] focus on unlearning in the context of LLMs, providing code that supports basic experimentation with textual data. However, their implementations are highly specialized and not intended for reuse outside the original context. Similarly, [112], [32], and [50] target unlearning in computer vision: as such, their codebases are scoped to specific datasets and do not support other data modalities.

Broader efforts, like [46], examine multiple models and data types, but do not offer a general-purpose framework: the code is focused on evaluating a fixed set of scenarios and is not structured for extensibility or long-term reuse. In all cases, measures implementations, data loading practices, and evaluation pipelines are developed independently, resulting in duplicated effort and low reproducibility.

With respect to **Graph Unlearning** (GU), both [157] and [203] omit GU entirely, presenting instead broader MU conceptual aspects. Likewise, [265] only identifies graphs as a potential avenue for future research. The more recent survey by [163] does tangentially include graphs, but specific graph-tailored methods are not discussed nor evaluated. Similarly, most MU frameworks do not include GU [83] or only list it as future work [158].

2.4 Our Contribution

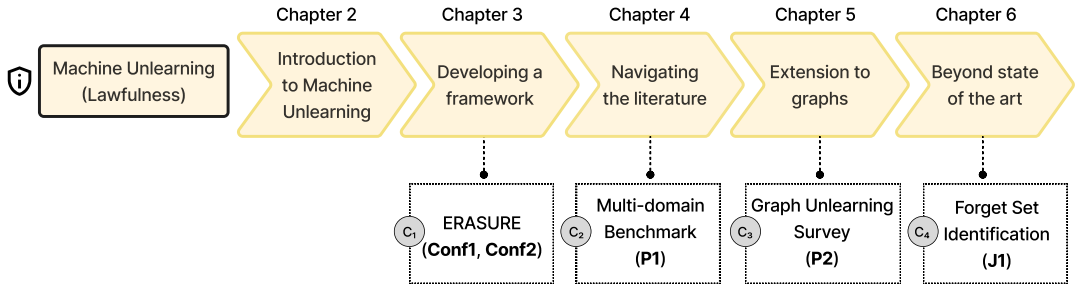


FIGURE 2.2: The research process we followed for the topic of Machine Unlearning, related to the TD of Lawfulness. Each step is annotated with the respective contribution of this thesis.

Figure 2.2 shows our contributions to the field of Machine Unlearning. Given the lack of reproducibility-focused frameworks in the literature, we developed ERASURE, a unified extensible framework for MU research, that we illustrate in Chapter 3. ERASURE faithfully implements the workflow of Figure 2.1 and is built to be easily extensible in all of its components.

By leveraging ERASURE, we then propose a multi-domain benchmark in Chapter 4. In this work, we establish coherence in this fragmented landscape by studying MU methods on diverse domains under a single, unified evaluation benchmark. We go beyond single-domain implementations that are present in the literature and evaluate the adaptability of MU techniques to different domains. Moreover, we introduce the *Laplacian Unlearning Multidimensional Assessment* (LUMA), a unified metric for quick and easy comparison of Unlearning methods.

Then, in Chapter 5, we address a critical gap in the literature: the absence of a comprehensive survey on graph unlearning methods. Given the importance of this field, we provide a formalization and conceptual clarification of its core components, along with a structured taxonomy of existing approaches. Our survey highlights recent advances and outlines promising directions for future research.

Lastly, in Chapter 6, we go beyond the state of the art: as seen in Figure 2.1, all MU methods assume to have as input a fully defined Forget Set D_f. However, the requirement of having the forget set as input is so restrictive that it might prevent MU from being profitably applied in many real-world scenarios. In this work, we fill this important gap, and study for the first time what we call the FORGET SET IDENTIFICATION problem (for short, FORSID): identify the samples to be removed by analyzing model's predictions for both unwanted and wanted examples.

As such, FORSID is an important complementary problem to MU, to be used coupled with (and not in substitution for) MU methods.

2.5 Future work

The field of MU is new, and despite these advances, most of it remains open to exploration. While most MU methods are tested on a single instance of forget set, we ideally want to reach trustworthy AI systems that evolve over time, that learn and unlearn continuously on evolving knowledge bases. This is the unlearning desiderata of several industry-leading companies like Google [279].

Moreover, the application of MU to federated learning is present, although still limited, and open to improvements. Several ML applications are only possible through federated learning, which also helps preserving user's data privacy, a common goal with Machine Unlearning.

Expanding MU is a necessary step toward building AI systems that respect privacy, remain trustworthy over time, and align with real-world deployment needs.

Chapter 3

ERASURE: Building a Unified Framework for Machine Unlearning

Machine unlearning is a growing field that currently lacks a widely accepted reference framework for collecting methods, metrics, and datasets.

Some studies in the literature have attempted to address this limitation. For instance, in [50], the authors provide the implementation of a limited number of existing unlearning techniques and metrics. However, the repository has not been designed to be an openly available library: the codebase does not allow easy extensions or usage in scenarios other than the replication of the experimental results of the paper. A further attempt at aggregating resources within machine unlearning is presented in [194], where references to code repositories of the primary unlearning techniques presented in the literature are collected. However, this collection remains fragmented, highlighting the pressing need for a unified framework that provides a standardized interface for seamlessly adopting diverse unlearning techniques and scenarios.

To address this gap, we present **ERASURE**[1], a flexible and modular framework designed to support research in MU. ERASURE simplifies experimentation by organizing all settings within a single JSON configuration file: dataset, model, unlearning method, and evaluation measures, as shown in Fig. 3.1, which provides an overview of the experimental workflow. The framework is built upon specialized design patterns tailored to requirements of MU.

ERASURE offers implementations of many state of the art unlearning techniques, ranging from retraining-based approaches to more efficient approximate methods. It supports easy access to widely used data sources such as Hugging Face [126], TorchVision [245], the UCI Repository [138], Torch Geometric [101], and supports multiple data types (image, text, tabular, graph). To assess the performance of unlearning techniques, ERASURE supports a set of evaluation measures that capture their *efficacy*, *utility*, and *efficiency* [121, 145]. It is also designed for easy customization, allowing users to add new methods, datasets, or measures with minimal setup. By bringing these components together, ERASURE aims to simplify experimentation, support reproducibility, and help the community address the growing societal need for responsible AI development.

As illustrated in Figure 3.1, ERASURE simplifies experimentation by organizing all settings within a single JSON configuration file: dataset, model, unlearning method, and evaluation metrics.

[1] Full code, and tutorial available at `https://github.com/aiim-research/ERASURE`

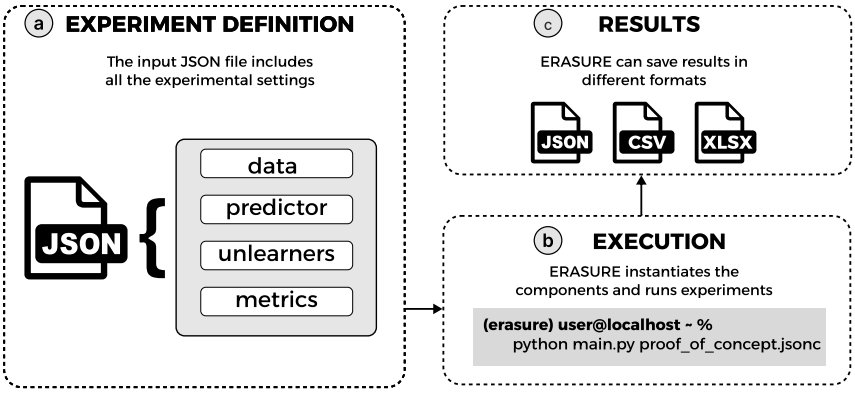


FIGURE 3.1: ERASURE in action – The experiment is defined via a single JSON configuration file that specifies required components and is executed by the main script.

These JSON objects provide an overview of the experimental workflow. Then, simply running the main file of ERASURE, passing the main configuration JSON as argument, is enough to run the entire pipeline and obtain the desired results. The framework is built upon specialized design patterns tailored to requirements of MU.

To showcase ERASURE's flexibility and robustness we present use case scenarios through a main configuration, enabling reproducible experimental evaluations of various unlearning techniques.

3.1 Machine Unlearning Workflow

ERASURE implements all steps of the workflow of Figure 2.1 by instantiating all the necessary objects. Specifically, the **Data Management** module of ERASURE is designed to meet the unique requirements of Machine Unlearning. All operations occur within the `DataManager` class, which orchestrates three main steps: *(i)* data loading, *(ii)* preprocessing, and *(iii)* partitioning. Partitioning the data is critical for Machine Unlearning, as it often involves more sophisticated strategies than conventional train/test splits, as shown in Fig. 2.1.

To support all the possible unlearning scenarios, ERASURE introduces a modular `DataSplitter` strategy that allows users to configure complex dataset partitions. Each `DataSplitter` defines a specific logic, such as selecting a percentage of the data, extracting a particular class, or filtering by a secondary label (e.g., a specific identity or topic to be forgotten). These actions can be executed sequentially, with each one optionally referencing the output of previous ones, enabling chained and hierarchical dataset splits. This design defines unlearning requests at various granularities, such as removing all samples from a particular user or forgetting only a subset of examples tied to a specific semantic label.

The **Predictor** component M is instantiated directly from the main configuration file. It is either a model that external libraries provide or a custom implementation. The configuration interface allows users to define all essential hyperparameters, such as the number of hidden layers, training epochs, and the optimizer. The optimizer is instantiated as a separate, fully configurable entity.

The **Unlearner** module represents the core abstraction for all unlearning methods within ERASURE. Each unlearning method extends this base class, encapsulating its algorithmic logic while maintaining modularity. To preserve the integrity of the original model, every Unlearner instance operates on an isolated copy of M.

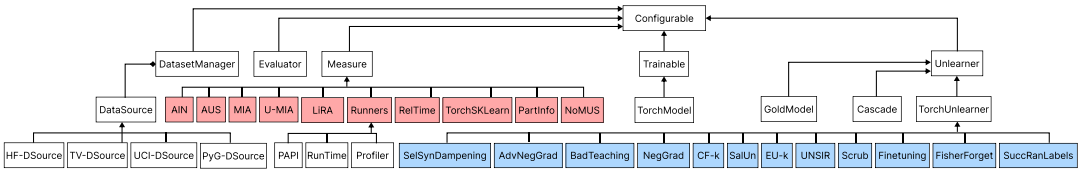


FIGURE 3.2: Overview of the main classes of the ERASURE Unlearning Framework and their hierarchical relations.

Finally, the **Evaluator** module integrates all essential components for assessing the performance of unlearning techniques. To facilitate large-scale experimentation, the `Evaluator Manager` automates the setup, memory management, and sequential execution of all configured `Measures`. Results are collected through a centralized `Evaluation` object, ensuring consistency and traceability across experiments. They can be saved in a variety of file formats.

3.2 Design Principles

Inversion of Control (IoC) [170] is a software design pattern that promotes modularity and ease of maintenance by decoupling object creation and control flow from the client logic. ERASURE adopts IoC to enable dynamic and extensible dependency management, allowing components to be instantiated from standard libraries (e.g., PyTorch or scikit-learn) and custom modules, including datasets, models, unlearning methods, or evaluation measures.

Factory Pattern [184] is another core design strategy adopted by ERASURE. It handles object creation through a centralized factory, making building different types of objects easier, especially when they have many parameters or dependencies. In ERASURE, most components are implemented as `Configurable` classes (see Fig. 3.2, showing the classes' hierarchy, for reference), meaning their parameters can themselves be instantiated via configuration. For example, a `Predictor` may include an `Optimizer` as one of its parameters, which can be further customized (e.g., specifying the learning rate or weight decay) directly through the configuration file. This nested configuration capability is made possible by ERASURE's implementation of the Factory Pattern, enabling highly flexible and declarative experiment setups.

Beyond Reproducibility. ERASURE provides a wide selection of ready-to-use data sources, models, unlearning methods, and evaluation measures (a complete list is available on the GitHub Repository, or in Fig. 3.2). At the same time, ERASURE is easily extensible: developers can integrate their components by subclassing the appropriate base classes and following the interface structure.

For example, adding a custom unlearning method only requires extending the base `Unlearner` class and defining the logic for model preparation, unlearning, and optional post-processing steps. Once implemented, the custom component can be used in experiments simply by referencing its import path and configuration parameters in the Main Configuration file. Moreover, multiple Unlearners can be concatenated using ERASURE's `compose` feature, enabling exploration of method combinations beyond those in the literature.

3.3 Core Components

ERASURE includes components (see Fig. 3.2) designed to develop and evaluate all the aspects of the unlearning process. Hereafter, we present the three main modules: data management, unlearning methods, and evaluation with measures.

Data Management - The *DatasetManager* orchestrates data handling by creating the *DataSource*, which is responsible for loading data from a specific source (e.g., files or repository). ERASURE provides built-in support for all the datasets available through the widely-used libraries HuggingFace [126], TorchVision [245], and UCI Repository [138]. Once loaded, the data undergoes a series of – built-in or custom – preprocessing steps that are applied on the fly as batches are loaded. Developers can seamlessly integrate their own preprocessing logic by porting their existing code into ERASURE. ERASURE's vision is that the data can be partitioned through a cascade of configurable *DataSplitters*. ERASURE already provides the splitters that solve the typical selection scenarios, e.g., by percentages, based on class groupings, or by a fixed sample count. This versatility enables users to define data partitions (such as training, testing, forget, or retain sets) to suit their specific experimental protocols without coding.

Unlearning Methods - The *Unlearner* class encompasses the common logic of each unlearning strategy. In ERASURE, we specifically included the most adopted and widely recognized Unlearners from the literature to evaluate the key dynamics of the machine unlearning field, ranging from retraining-based methods to efficient approximate techniques such as selective gradient dampening and synaptic decay: ***Gold-Model***, ***Fine-Tuning***, ***Successive Random Labels***, ***CF-k*** [109], ***EU-k*** [109], ***NegGrad*** [110], ***Advanced NegGrad*** [50], ***UNSIR*** [242], ***Bad Teaching*** [51], ***SCRUB*** [151], ***Fisher Forgetting*** [110], ***Selective Synaptic Dampening*** [103], and ***Saliency Unlearning*** [95]. Moreover, the 13 available Unlearners have been implemented and refactored, with the possibility of chaining (i.e., by adopting the *Cascade* class) them together or applying different combinations of them, e.g., using any unlearning method by exploiting the saliency maps generated by SalUn.

Evaluation - The *Evaluator* module incorporates all the necessary components for assessing the different kinds of performance of unlearning techniques, such as efficiency and efficacy. To facilitate large-scale experiments, the *Evaluator Manager* automates the setup and evaluation process by initializing all the required *Measures* (see its abstract class) and by executing them sequentially. A shared *Evaluation* object collects the results of the evaluated measures. The evaluation starting point (see the *Runners* measure class) might include **efficiency** performance metrics such as the running time, the FLOPS (floating point operations), and memory usage (implemented through PAPI [128] and torch Profiler, because these metrics must be measured during the running of each Unlearning method. With respect to **efficacy**, we implemented UMIA (i.e., Unlearning-specific MIA) [121], Relearning Time [242, 111, 110], Anamnesis Index [52] and Adaptive Unlearning Score (AUS) [57]. For **accuracy**, all metrics in scikit-learn [199] are available.

3.4 Example of Main Configuration

The structure of the main configuration, illustrated in Listing 3.1, is based on four core components: `data`, `predictor`, `unlearners`, and `evaluator`. These directly correspond to the elements outlined in the Machine Unlearning workflow of Fig. 2.1.

```
1  "data": {"class":"erasure.data.<NS>.DatasetManager",
2         "parameters": {
3             "DataSource": { ... },
4             "partitions": [ "p_1", "p_2", ... , "p_n" ] }},
5  "predictor": {"class": "erasure.model.<NS>.TorchModel",
6    "parameters": {
7      "optimizer": {"class": "torch.optim.Adam"},
8      "loss_fn": {"class": "torch.nn.CrossEntropyLoss"},
9      "model": {"class": "erasure.<NS>.BERTClassifier"}}},
10 "unlearners":[
11     {"compose_gold" : "configs/snippets/u_gold.json"},..,
12     {"class":"erasure.<NS>.AdvancedNegGrad",
13         "parameters": {
14               "epochs": 1,
15               "ref_data_retain": "retain",
16               "ref_data_forget": "forget",
17               "optimizer": {"class": "torch.optim.Adam",
                      "parameters": {"lr": 0.0001}}} }],
18 "evaluator":{
19   "class": "erasure.evaluations.<NS>.Evaluator",
20   "parameters": { "measures":[ ... ] } }
```

Listing n. 3.1: Structure of the main configuration snippet for an ERASURE experiment. $\langle NS \rangle$ compacts sub-modules namespace.

Each module requires a `class` field specifying its implementation and a `parameters` field defining the initialization configuration.

The `data` section (Lines 1-4) defines the dataset and how to partition it. This includes selecting a data source and applying preprocessing (e.g., filtering) via modular splitting components.

The `predictor` section (Lines 5-9) specifies the model and all related settings. The example shown uses a PyTorch-based model, where the optimizer and loss function can be selected directly from existing libraries, enabling easy reuse without additional code.

The `unlearners` (Lines 10-17) field is a list that defines the unlearning methods to be applied and then evaluated. Each entry includes its configuration, such as the relevant data partitions and optimization settings. All `unlearners` are executed independently to ensure isolation. The `compose_` feature, shown in this section, enables loading JSON snippets from external files for any configuration component, allowing users to easily reuse predefined setups across data sources, models, unlearners, and evaluation measures.

Lastly, the `evaluator` section (Lines 18-20) defines the measures. Although the specific measures are omitted for brevity, they are fully customizable and follow the same configuration logic.

3.5 ERASURE in action

To demonstrate the practical utility and flexibility of ERASURE, we performed proof-of-concept experiments with several state of the art evaluation measures (red boxes in Fig. 3.2) on four datasets spanning the supported domains (i.e., Tabular, Images, Text, and Graph). In Table 3.1 we report the results for two unlearning methods selected from the 12 included in ERASURE (blue boxes in Fig. 3.2), which represent

DS	Unlearner	$\Delta Acc_t \downarrow$	$\Delta Acc_f \downarrow$	$\Delta Acc_r \downarrow$	$\Delta AUS \downarrow$	$\Delta AIN \downarrow$	$\Delta UMIA \downarrow$	NoMUS ↑	Time (Speedup) ↓
Tabular	`Orig.`	.001 ± .000	.041 ± .002	.006 ± .003	.014 ± .004	.998 ± .000	.009 ± .003	.916 ± .001	-
	`Gold`	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.924 ± .002	461.688 ± .1 (1.0×)
	`SSD` [103]	.001 ± .000	.041 ± .002	.006 ± .003	.014 ± .004	.998 ± .000	.009 ± .004	.915 ± .001	11.889 ± .1 (38.8×)
	`SalUn` [95]	.006 ± .003	.023 ± .003	.022 ± .002	.012 ± .001	.594 ± .089	.003 ± .002	.928 ± .001	2.824 ± .0 (163.5×)
Images	`Orig.`	.003 ± .001	.034 ± .002	.003 ± .002	.035 ± .004	.885 ± .156	.031 ± .007	.918 ± .007	-
	`Gold`	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.949 ± .005	1598.795 ± 13.6 (1.0×)
	`SSD` [103]	.007 ± .003	.032 ± .005	.007 ± .005	.041 ± .005	.885 ± .156	.031 ± .006	.916 ± .001	102.026 ± .8 (15.7×)
	`SalUn` [95]	.002 ± .001	.007 ± .000	.014 ± .002	.011 ± .004	1.770 ± .951	.007 ± .004	.949 ± .003	42.110 ± .8 (38.0×)
Text	`Orig.`	.042 ± .008	.035 ± .000	.052 ± .010	.104 ± .016	.859 ± .104	.090 ± .014	.796 ± .008	-
	`Gold`	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.907 ± .006	8454.617 ± 48.6 (1.0×)
	`SSD` [103]	.036 ± .009	.033 ± .003	.045 ± .012	.093 ± .016	.568 ± .304	.071 ± .019	.817 ± .008	1736.043 ± .1 (4.9×)
	`SalUn` [95]	.004 ± .003	.020 ± .005	.011 ± .002	.037 ± .020	.221 ± .157	.019 ± .009	.887 ± .011	2713.718 ± 4.9 (3.1×)
Graph	`Orig.`	.007 ± .001	.002 ± .001	.008 ± .004	.002 ± .001	.067 ± .094	.008 ± .004	.896 ± .003	-
	`Gold`	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.000 ± .000	.897 ± .004	12.689 ± .2 (1.0×)
	`SSD` [103]	.007 ± .001	.002 ± .001	.008 ± .004	.002 ± .001	.067 ± .094	.006 ± .001	.900 ± .003	1.808 ± .0 (7.0×)
	`SalUn` [95]	.005 ± .003	.014 ± .005	.006 ± .009	.014 ± .005	.067 ± .094	.007 ± .008	.896 ± .008	1.957 ± .0 (6.5×)

TABLE 3.1: Experimental comparison of different unlearning methods across various datasets (tabular, image, textual, and graph), with 3 runs for each experiment. For each metric and dataset, the best result is in bold. `Orig.` and `Gold` models are highlighted.

two of the most popular unlearning paradigms. Full configurations are included in the repository[1].

Methods. Selective Synaptic Dampening (`SSD`) [103] modifies model weights directly, demonstrating the framework's support for model editing methods. In contrast, `SalUn` [95] leverages a saliency map to identify and apply unlearning only on the weights most activated by the forget set. This exemplifies how ERASURE supports unlearning on specific model subcomponents, while its `compose` feature allows the integration of multiple unlearning methods in cascade.

Datasets and models. We selected a widely used and representative dataset for each domain: Adult [19] for Tabular, CelebA [165] for Images, AG News [115] for Text, and BBBP [216] for Graph. We employed a two-layer network for Tabular, a ResNet18 for Images, a BERT model for Text, and a two-layer GCN for the Graph domain.

Measures. We employed several state of the art measures for MU evaluation. For *utility*, we report the difference in Accuracy on the test, forget, and retain sets (ΔAcc_t, $\Delta\ Acc_f$, ΔAcc_r) with respect to the `Gold` baseline. Similarly, for *efficacy*, we compute the change in the Adaptive Unlearning Score [57] (ΔAUS), Anamnesis Index [52] (ΔAIN), and Membership Inference Attack [50] (ΔUMIA) effectiveness compared to the `Gold` model. We use the Normalized Machine Unlearning Score [50] (NoMUS) as a comprehensive metric combining *utility* and *efficacy*, and adopt it to select the best unlearning configuration for each method and domain. Lastly, for *efficiency*, we report the Time and the Speedup with respect to complete model retraining (referred to as the `Gold` model in Table 3.1).

Outcomes. In terms of *utility*, both `SSD` and `SalUn` maintain high model performance, with accuracy degradation that is often negligible across the different domains. For *efficacy*, `SalUn` performs generally better across all domains except Graph, where the compound NoMUS score is slightly lower. The Text domain exhibits the most significant performance gap: `SalUn` severely outperforms `SSD` on Text across all *efficacy* measures. It must be noted that, since both methods are state of the art, they still achieve high NoMUS scores on all domains. However, the Text domain seems the hardest to unlearn, possibly due to bigger, more complex models. Lastly, in terms of *efficiency*, `SalUn` is faster than `SSD` on all domains except Text. This is expected,

as fine-tuning a large model is more time-consuming than modifying its weights directly. Nonetheless, both methods have substantial Speedups across all domains relative to the `Gold` model. This improvement is particularly critical in the context of MU, where the practicality of a technique is undermined if its computational cost exceeds that of retraining from scratch.

Highlights. While the results cover only a subset of what ERASURE supports, they provide meaningful insights. Methods like `SSD` and `SalUn` show that high predictive performance can be maintained after unlearning. However, no single approach consistently excels across all evaluation criteria, highlighting the inherent trade-offs between *utility*, *efficacy*, and *efficiency*. However, the significant Speedup obtained reinforces the core motivation behind MU itself.

Conclusion. ERASURE introduces a unified, extensible platform for machine unlearning, addressing a critical need for reproducible research. Using results from two representative methods across four domains as a proof of concept, we demonstrate the framework's ability to support diverse experimental setups at scale. ERASURE's modular design enables seamless integration of custom datasets, unlearning techniques, and evaluation measures with minimal overhead. We believe this platform will accelerate progress in the field by equipping researchers with the tools needed to build, compare, and refine unlearning methods efficiently.

Chapter 4

Benchmarking Machine Unlearning methods

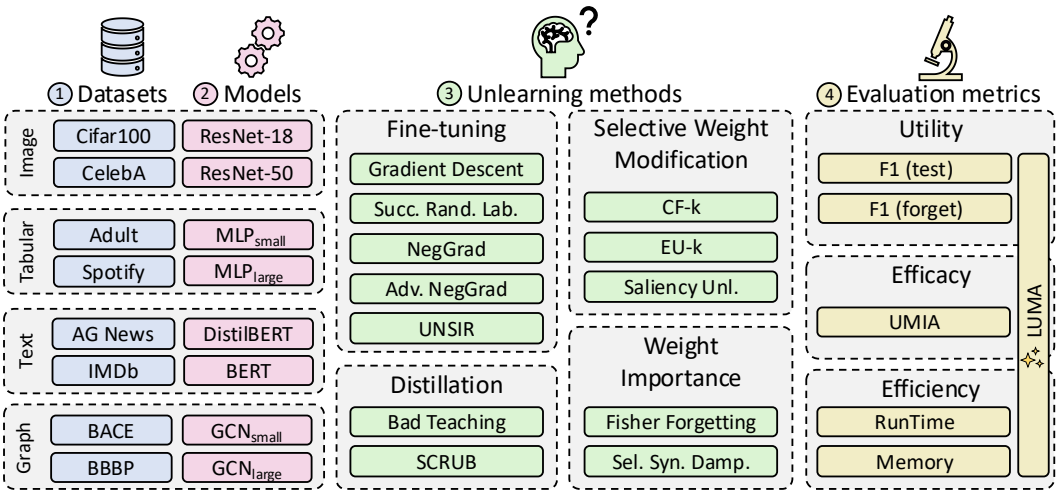


FIGURE 4.1: Experimental workflow of our benchmark. We evaluate 8 classification datasets across 4 domains, training 2 models per domain. We test 12 unlearning methods and assess them across 3 evaluation dimensions. We introduce a unified metric, LUMA, to facilitate comparison.

Building on the ERASURE framework we presented, we establish coherence in the fragmented landscape of MU by studying methods on diverse domains under a single, unified evaluation benchmark. As illustrated in Figure 4.1, the proposed unlearning benchmark consists of three steps: (1) selecting datasets from one of the four supported domains, (2) training the appropriate model (between two sizes), and (3) applying unlearning methods directly to the model without requiring domain-specific adjustments. After unlearning, step (4) is the collection and analysis of the values of the evaluation metrics.

Other benchmarks. As an emerging field, machine unlearning has still seen limited systematic evaluation of the extensive research developed in recent years. In addition, most existing benchmarks remain restricted to a single domain. We list these studies in Table 4.1, providing details on their coverage (in terms of datasets and methods), the domains they include, and the number of aspects of MU they capture with their metrics.

Several works only focus on one domain: [112, 50, 32] evaluate and compare unlearning approaches exclusively on image datasets. Similarly, [145] focuses solely on spoken language understanding datasets.

	Coverage		Modalities						Metrics		
	# datasets	# methods	Image	Text	Tabular	Graph	Speech	Video	Utility	Efficacy	Efficiency
Our	8	12	✓	✓	✓	✓	✗	✗	✓	✓	✓
[46]	6	5	✓	✓	✗	✗	✓	✓	✓	✗	✓
[50]	2	7	✓	✗	✗	✗	✗	✗	✓	✓	✗
[32]	5	11+7[1]	✓	✗	✗	✗	✗	✗	✓	✓	✓
[112]	1	6	✓	✗	✗	✗	✗	✗	✓	✗	✗
[145]	5	8	✗	✗	✗	✗	✓	✗	✓	✓	✓

TABLE 4.1: Comparison of classification benchmarks present in the literature.

While valuable, no method was tested on multiple domains jointly, an omission that critically limits our understanding of the generalizability of MU methods. Moreover, most benchmarks either do not consider all the evaluation dimensions or are very limited in the number of datasets and methods they employ.

To the best of our knowledge, the benchmark introduced by [46] remains the only existing effort to evaluate MU methods across multiple domains. However, it lacks a crucial component of MU evaluation: the incorporation of metrics that directly assess the efficacy of the unlearning process. In their absence, the reported results lack rigorous quantification of unlearning effectiveness, undermining the ability to address the core privacy and security goals that MU is meant to achieve [265]. In addition, as shown in Table 4.1, their study covers a far narrower set of methods (5 versus our 12), which restricts both the scope and generalizability of their findings.

Moreover, no benchmark in the literature has assessed MU methods in the Tabular and Graph domains. Conversely, our benchmark fills this gap by considering 8 datasets (2 per domain, more than any other benchmark) and 12 methods across four different data domains: Image, Text, Tabular, and Graph. Moreover, we incorporate all key dimensions of MU evaluation (utility, efficacy, and efficiency) within a single framework, introducing LUMA (see Sec. 4.2), the first unified multidimensional metric. By providing the most extensive and systematic benchmark to date, we bring coherence to the fragmented landscape of MU in classification tasks and establish an extensible reference point for validating future methods.

Contributions. Our contributions in this Section are the following: ***(i).*** We introduce LUMA, a unified metric that jointly captures the three key aspects of MU: utility, efficacy, and efficiency. ***(ii).*** We validate LUMA on a benchmark of 12 machine unlearning methods on 8 classification datasets spanning four domains (the most comprehensive evaluation of MU methods to date), including the first benchmarks for tabular and graph data. ***(iii).*** We propose a taxonomy of the 12 benchmarked methods to aid in structuring and understanding the field. ***(iv).*** The benchmark we provide publicly is reproducible, easily extensible, and intended to serve as a foundation for future research in the MU field of study.

4.1 Benchmark Design

We formalize the standard MU workflow as follows: a model M is first trained on a dataset D. Upon receiving an unlearning request, a subset of samples to be removed is specified as the forget set D_f, with the retain set defined as $D_r = D \setminus D_f$. The goal of an unlearning method (Unlearner) is to transform M into an updated model

[1] 7 of the methods reported in this survey were taken from a Machine Unlearning competition and, as such, were not formally peer-reviewed.

M' that closely approximates a retrained model `Gold` (Gold Model), i.e., the one obtained by retraining from scratch on D_r.

We designed our benchmark to faithfully implement this workflow uniformly across datasets. For each method, we perform hyperparameter selection over the learning rate in the range 10^{-6} to 10^{-3}, and additionally tune method-specific parameters where relevant (e.g., the dampening constant in Selective Synaptic Dampening [103]). All experiments were run three times with different seeds to account for statistical variation.

4.1.1 Datasets and models

We evaluate all unlearning methods across four domains – image, text, tabular, and graph – using two publicly available datasets per domain. For domains with established MU benchmarks, we adopt datasets widely used in prior work [51, 110, 103, 242, 95, 109, 40, 46]. For domains less explored in this context, we select representative and widely studied datasets [155, 264].

The forget set D_f for each dataset is constructed following one of two strategies: (i) selecting training samples containing predefined named entities (e.g., person or organization names) or identity-related attributes **(NE)**, simulating realistic deletion requests; or (ii) sampling 20% of the training set uniformly at random **(SA)** when identity-based filtering is not feasible. In both cases, the forget set size is approximately 20% of the training data, consistent with prior MU literature.

For the **image** domain we selected CIFAR-100 ([148]) ***(SA)*** and CelebA ([165]) ***(NE)***. CelebA was used for multilabel classification. For **text** we selected IMDB ([108]) ***(NE)*** and AG News ([115]) ***(NE)*** for **tabular** data, we selected the datasets of Adult ([19]) ***(SA)*** and Spotify Tracks ([172]) ***(SA)***, and for **graphs** we selected BBBP ([216]) ***(SA)*** and BACE ([264]) ***(SA)***.

For each domain, we adopt model architectures widely used in the MU literature, selecting both smaller and larger variants. In the **image** domain, we use ResNet-18 and ResNet-50 [123], consistent with prior MU studies [103, 95, 110, 242]. For **text**, we fine-tune DistilBERT [220] and BERT [79] (110M+ parameters), each augmented with a classification head. In the **tabular** domain, we employ two fully connected networks with one and three hidden layers, respectively, each layer containing 100 units. Finally, for **graphs**, we use two GCN-based classifiers: one with a backbone of one GCN layer followed by one dense layer, and another with two layers followed by one dense layer.

4.1.2 Unlearners

For this benchmark, we include 12 different state of the art unlearning methods. To facilitate analysis and discussion, we categorize these into four groups based on their unlearning strategy:

Fine Tuning (`FT`): Gradient Descent (`GD`), Successive Random Labels (`SRL`), NegGrad (`NG`) ([110]), Advanced NegGrad (`ANG`) ([50]), UNSIR (`UNSIR`) ([242]). These methods train the model further according to specific strategies.

Selective Weight Modification (`SWM`): CF-k (`CF`k) ([109]), EU-k (`EU`k) ([109]), Saliency Unlearning (`SalUn`) ([95]). These methods only operate on a subset of the model parameters, typically the last layers or weights identified via saliency masking.

Distillation (`DIS`): Bad Teaching (`BT`) ([51]), SCRUB (`SCRUB`) ([151]). These methods rely on teacher–student setups to alter the target model's behavior.

Weight Importance (`WI`): Fisher Forgetting (`FF`) ([110]), Selective Synaptic Dampening (`SSD`) ([103]). These directly modify model weights, leveraging the Fisher Information Matrix to estimate the importance of parameters with respect to D_f.

For reference, the metrics related to the ***Original*** (`Orig.`) model and the ***Gold Model*** (`Gold`) are also reported for each dataset. In a few cases, certain Unlearners required minor modifications (e.g., change of loss function) to work consistently across settings.

4.1.3 Metrics

We evaluate unlearning methods along three key dimensions:

Utility: the predictive performance of the model after unlearning to evaluate the unintended degradation on non-forgotten samples. We compute the F1 score on both the test set and the forget set.

Efficacy: the extent to which the influence of the forget set is removed. We adopt the Unlearning Membership Inference Attack (UMIA) [121], which tests whether the model can distinguish forgotten samples from previously unseen ones. Effective unlearning yields UMIA values close to those of the Gold model, avoiding both *over*- and *under*-unlearning [228].

Efficiency: the computational cost of unlearning. We measure the relative training speedup compared to full retraining, as well as the peak GPU memory usage during execution.

In the following, we refer to Utility, Efficacy, and Efficiency as Evaluation Dimensions (*ED*). While we refer to single benchmarking scores chosen for the *ED*s (e.g., the UMIA for Efficacy) as *measures*. Compound metrics will be referred to simply as *metrics*.

4.2 LUMA: A Unified Metric

Although examining the three *ED*s separately offers a detailed view of an Unlearner's quality, a unified MU metric is essential for comprehensively comparing methods, whether to discard underperforming hyperparameters or to identify the best performer in real-world applications.

Despite its importance, a unified metric remains a critical gap in the MU literature. [145] introduces GUM, a Global Unlearning Metric, but it faces limitations on scalability to multiple measures and resilience to edge cases. GUM reduces each dimension to a single proxy measure (e.g., UMIA for Efficacy, F1 score for Utility, RunTime for Efficiency), which fails to capture the richness of MU performance when multiple measures per *ED* are available: in practice, MU performance is evaluated by several measures (e.g., F1 scores on forget and test sets, RunTime and GPU memory usage), and considering all these aspects jointly provides a more faithful assessment than relying on single-value proxies. [279] proposed Tug of War (ToW), defined as the product of the relative differences between the Gold and retrained models on accuracies over the test, retain, and forget sets. However, ToW excludes Efficiency, leading to a partial evaluation. We provide empirical evidence of these shortcomings in Appendix 4.2.1.

To address these limitations, we introduce the **Laplacian Unlearning Multidimensional Assessment (LUMA)**. Unlike GUM and ToW, LUMA is multidimensional by design, incorporating vectors of measures within each *ED*, and flexible, allowing users to add any measure deemed relevant. LUMA computes the distance

of an unlearned model from the retrained Gold model across utility, efficacy, and efficiency, while penalizing large deviations in any single dimension.

All *ED*s are referenced against the Gold Model (`Gold`), which represents the ideal target of MU methods ([265]) as the model retrained from scratch only on the retain set. Let `Gold` and M' be the Gold Model and the model obtained via the application of the MU method to be measured, respectively.

In our benchmark, the efficacy dimension is measured by UMIA, the utility dimension by F1 score on the test and forget sets, and the Efficiency dimension by runtime and GPU memory usage. Each *ED* can therefore be represented as a vector of the values computed by the chosen measures on the considered model $\bullet \in \{\texttt{Gold}, M'\}$

$$\mathbf{e}_\bullet = [\,\text{UMIA}_\bullet\,]^\top,\ \mathbf{u}_\bullet = \left[F1_\bullet^{test},\ F1_\bullet^{forget}\right]^\top,\ \mathbf{t}_\bullet = \left[\text{RunTime}_\bullet,\ \text{Memory}_\bullet\right]^\top$$

To compare the Unlearners against the Gold Model, we map efficacy and utility into similarity factors using a γ-parametrized Laplacian kernel. This choice penalizes *under*- or *over*-performance relative to the Gold Model, while amplifying large deviations in any individual measure.

$$M_U(\mathbf{u}_{\texttt{Gold}}, \mathbf{u}_{M'}) = \exp((-\gamma\|\mathbf{u}_{\texttt{Gold}} - \mathbf{u}_{M'}\|_1))\,,$$
$$M_E(\mathbf{e}_{\texttt{Gold}}, \mathbf{e}_{M'}) = \exp((-\gamma\|\mathbf{e}_{\texttt{Gold}} - \mathbf{e}_{M'}\|_1))\,.$$

While their closeness to the Gold Model evaluates utility and efficacy, efficiency follows a different principle: the less time and memory a method consumes, the better; the Gold Model provides an upper bound reference point. To capture this, we first normalize each efficiency measure by the corresponding value of the Gold Model. We then apply a logarithmic transformation to smooth extreme differences and emphasize relative improvements over absolute ones. Finally, we apply an exponential decay so that larger deviations from the Gold baseline are penalized more strongly, and aggregate the resulting values through a weighted average to obtain a single efficiency score:

$$M_T(\mathbf{t}_{\texttt{Gold}}, \mathbf{t}_{M'}) = \sum_i \mathbf{w}_i \exp\left[-\left(\frac{\log(1+\mathbf{t}_{M',i})}{\log(1+\mathbf{t}_{\texttt{Gold},i})}\right)^3\right],$$

where $\mathbf{w}_i$ represents the weight assigned to the i^{th} efficiency measure.

Finally, we define the unified metric as:

Definition 1 (LUMA)**.** Let M_U, M_E, M_T be the similarity scores for utility, efficacy, and efficiency relative to the Gold model (as defined above). Then

$$\text{LUMA} = \frac{3}{\frac{1}{M_U} + \frac{1}{M_E} + \frac{1}{M_T}}.$$

In other words, LUMA is the harmonic mean of M_U, M_E, and M_T. LUMA is parametrized by the γ of the Laplacian kernels and the weight vector $\mathbf{w}$ assigned to the efficiency measures. In our setting, we set $\gamma = 3$ and $\mathbf{w} = (0.9, 0.1)$, assigning 90% of the weight to RunTime and 10% to memory efficiency. We detail parameter tuning of LUMA in Section 4.2.1.

Note that since the Gold Model is also scored (with a value of 1 for both Utility and Efficacy, as the difference w.r.t. itself is, by definition, 0), its LUMA is fixed. Consequently, any Unlearner that takes longer than the Gold Model will inevitably be penalized by receiving a lower LUMA score, even if it achieves perfect similarity with the Gold Model. This property makes it straightforward to identify cases where an Unlearner is not suitable for application.

LUMA offers several advantages: *(i).* It is designed to range from 0 to 1, where 1 is the ideal MU algorithm that returns a model identical to the Gold Model with no additional cost in time or memory usage. *(ii).* It strongly penalizes any significant deviation on any measure, enabling fast pruning of ill-defined (or ill-parameterized) Unlearners. *(iii).* It is extensible, allowing the integration of any task-specific measure. For example, ROC can replace F1 in the Efficacy *ED*, or be added alongside it without further modifications. This flexibility is unique to LUMA and ensures seamless integration with future evaluation protocols.

4.2.1 LUMA against other metrics

In this section, we show shortcomings and issues of the previous two unified unlearning metrics, GUM([145]) and ToW ([279]). Then, we detail how LUMA handles edge cases and its sensitivity to hyperparameters.

Shortcomings of other metrics

Indicating the Gold Model with g and the resulting Unlearned model with u, GUM is defined as:

$$\text{GUM} = \frac{(1+\alpha+\beta)UET}{\alpha ET + \beta UT + UE}$$

where $U = 1 - \left|F1_T^{(g)} - F1_T^{(u)}\right|$, $\quad E = 1 - \left(\frac{\text{MIA}'(u)-\text{MIA}'(g)}{\text{MIA}(o)-\text{MIA}'(g)}\right)^2$, with

$$\text{MIA}'(u) = \min\{\text{MIA}(u), \text{MIA}(o)\}, \quad \text{MIA}'(g) = \min\left\{\text{MIA}(g), \frac{\text{MIA}'(u)+\text{MIA}(o)}{2}\right\}.$$

and lastly $T = 1 - \frac{\log\left(T^{(u)}+1\right)}{\log\left(T^{(g)}+1\right)}$.

While GUM includes all the three key evaluation dimensions (*ED*s) for MU methods (efficacy, utility, efficiency) it suffers from two core issues: *(i).* Only using one measure per dimension: the F1 score on the test set is not enough to assess the model's utility, and the RunTime is not enough to assess the MU method's efficiency. The F1 score on the forget set and the memory efficiency are also important. *(ii).* When $MIA'(u) - MIA(o) \leq \epsilon$ with a reasonably small ϵ, E grows without bound, effectively dominating GUM to unusable values. Worse, the metric is not computable at all for a division by 0 given two conditions:

- $MIA(g) \geq MIA(o)$
- $MIA'(u) \geq MIA(o)$

Consider the definition of E, specifically the denominator $MIA(o) - MIA'(g)$. This will be 0 when $MIA(o) = MIA'(g)$. Given the definition of $MIA'(g)$, this can only happen when $MIA(g) > MIA(o)$ (first condition) and $MIA'(u) = MIA(o)$. The latter is true when $MIA(u) \geq MIA(o)$ (second condition). We show an example in Section 4.2.1.

Tug of War (ToW) is instead defined as:

$$\text{ToW}(\theta_u, \theta_g, S, R, D_{\text{test}}) = \big(1 - d_a(\theta_u, \theta_g, S)\big)\,\big(1 - d_a(\theta_u, \theta_g, R)\big)\,\big(1 - d_a(\theta_u, \theta_g, D_{\text{test}})\big),$$

where

$$a(\theta, D) = \frac{1}{|D|} \sum_{(x,y)\in D} \mathbf{1}[f(x;\theta) = y]$$

is the accuracy of a model f parameterized by θ on dataset D, and

$$d_a(\theta_u, \theta_r, D) = \big|\, a(\theta_u, D) - a(\theta_r, D) \,\big|$$

is the absolute difference between the accuracies of models θ_u (the unlearned model) and θ_g (the Gold Model) on the dataset D.

ToW includes multiple metrics for the Utility dimension, but misses the Efficiency dimension entirely. Accordingly, the Gold Model will always obtain the optimal ToW score of 1 (regardless of how expensive it is to train) and two unlearners that output the same retrained model with largely varying RunTimes will obtain the same ToW score. Moreover, by omitting MIA, ToW doesn't capture the information leakage of any MU method, which is a known problem in the literature ([265, 155]).

Empirical evidence

TABLE 4.2: Extract of results on the Adult dataset (Table 4.4).

Group	Method	adult3mlp				
		F1 test	UMIA	Runtime	LUMA	GUM
–	Orig.	.793 ± .002	.498 ± .001	135.7 ± 7.9	.631 ± .000	X
–	Gold	.791 ± .002	.499 ± .001	135.7 ± 7.9	.636 ± .000	X
FT	GD	.791 ± .003	.498 ± .002	88.3 ± 4.9	.710 ± .001	X

Table 4.2 is an extract from the bigger Table 4.4 shown in Section 4.3. This is empirical evidence of the proven problem of GUM in Section 4.2.1: we have that $MIA(g) \geq MIA(o)$ as $0.499 \geq 0.498$ (first condition) and that $MIA'(u) \geq MIA(o)$ as $0.498 \geq 0.498$. For this reason, GUM cannot be calculated and will fail because of a statistical variation.

GUM is very useful because it combines measures across all *ED*s, but suffer from numerical instability under some conditions. LUMA fixes this by employing Laplacian kernels.

ToW suffers from the opposite problem: ignoring Efficiency entirely. Consider a toy Unlearner that simply retrains the model from scratch, but deliberately taking double the time to do so. Despite the wasted resources, it would still achieve a perfect ToW score. This illustrates that, to serve as a truly unified metric for unlearning, the *ED*dimension of Efficiency must be incorporated.

LUMA's behavior and hyperparameters

In contrast, LUMA fixes all shortcomings by considering of all the three dimensions, possibly with more than one metric each. By employing Laplacian kernels, LUMA is easily extendable with future MU measures.

In this Section, we analyze the impact of each *ED* on LUMA, to shed light on its sensitivity and robustness.

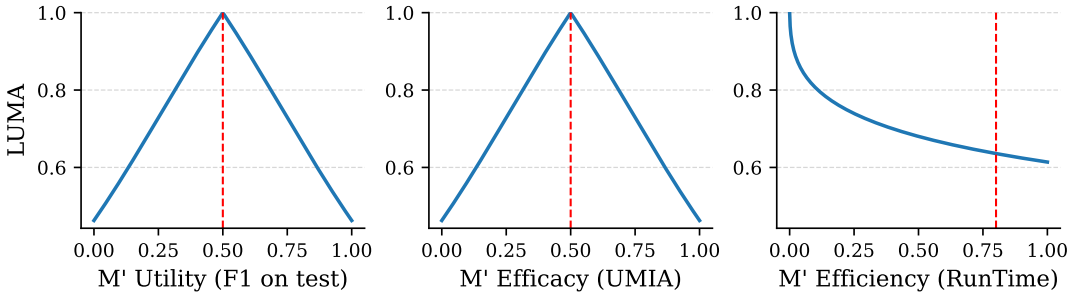


FIGURE 4.2: Sensitivity of the proposed LUMA metric with respect to each *ED*.

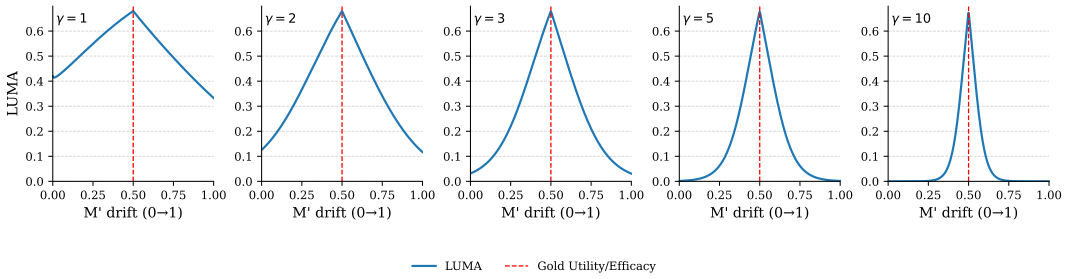


FIGURE 4.3: Effect of the Laplacian kernel parameter γ on the LUMA metric as a model drifts from an all-zero representation to an all-one representation of the *ED*s of Utility and Efficacy.

By construction, the Laplacian kernels ensure that both positive and negative drifts in an evaluation dimension are penalized symmetrically. Figure 4.2 illustrates this property, showing the response of LUMA with respect to each *ED* while assuming perfect performance on the remaining *ED*s. For Utility and Efficacy, LUMA reaches its maximum value of 1 when the varying *ED* matches the performance of `Gold`, and decreases smoothly as the model drifts away in either direction. For Efficiency, LUMA decreases inversely with runtime, lowering as runtime increases.

The Laplacian kernels are parametrized by γ. In this work, we chose $\gamma = 3$ to severely punish even single-measure drifts between the Gold Model (`Gold`) and the unlearned Model (M').

Figure 4.3 illustrates the impact of the kernel parameter γ on LUMA as M' drifts from 0 to 1 across all measures. For reference, the Gold model is fixed at 0.5 on all measures. In this work, we chose $\gamma = 3$ as it provided the best smoothness.

Moreover, LUMA is customizable in terms of weights assigned to measures in the Efficiency *ED*. The vast majority of works in the Machine Unlearning literature only consider RunTime when evaluating Efficiency, discarding peak memory usage. In this work, we introduced memory usage as a measure with weight = 0.1, to keep RunTime as the most important efficiency metric. This weight vector can be customized according to available resources and the task at hand. Figure 4.4 shows the sensitivy of LUMA to the weight vector w when the runtime scales w.r.t. to the Gold Model, assuming equality on the Utility and Efficacy *ED*s. The figure corroborates the correctness of LUMA: if we assign weight 1 to RunTime, and the RunTime is 0, LUMA will be 1, although this is basically unachievable in practice. The vector w, weighting Efficiency measures, is customizable. In this study we set $w = [0.9, 0.1]$, as runtime is the dominant Efficiency factor in unlearning. Figure 4.4 shows that this choice gives runtime decisive influence over the score while still ensuring memory

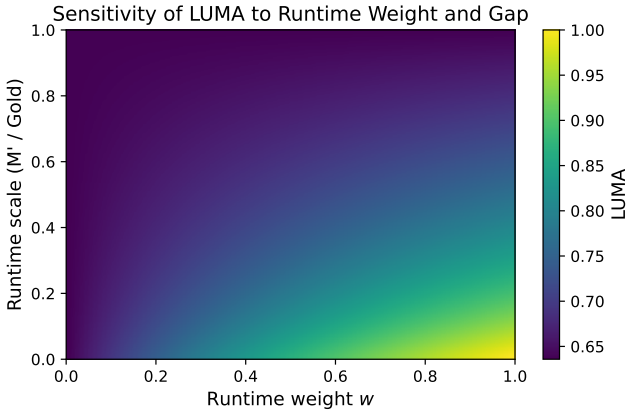


FIGURE 4.4: Heatmap of LUMA values as a function of runtime weight w (x-axis) and runtime scale of M' relative to the Gold model (y-axis). The weight assigned to memory is $1 - w$.

is not ignored.

Summing up, *(i).* By employing Laplacian kernels, LUMA fixes the shortcomings of GUM for edge cases, and surpasses both GUM and ToW in considering more than one measure per *ED*and being easily extensible with other metrics. *(ii.)* By incorporating all *ED*s, LUMA surpasses ToW in completeness, as the latter did not consider Efficiency at all.

4.3 Results

In this Section, we summarize the main findings from our benchmark study. In Section 4.3.1, we start by studying hyperparameters and we show that the size of the model has little impact on the performance of Unlearners. In Section 4.3.2, we report the main results from our experimentation on the four domains (Tabular, Image, Textual, Graphs). Finally, Section 4.3.3 summarizes takeaways.

4.3.1 Hyperparameter tuning and model selection

Most of the Unlearners are only parametrized by their Learning Rate (LR) of either the fine-tuning to be applied, the distillation, or any semi-Newton step they employ (refer to Section 4.1.2 for a taxonomy). In our benchmark, we tuned these parameters by employing LRs one order of magnitude lower, one order of magnitude higher, and equal to the training LR for each domain.

Figure 4.5 shows the LUMA of each unlearner across datasets and models as the learning rate varies. We use marker size to represent the underlying model dimension. Figure 4.5 shows that the Negative Gradient unlearners (`NG` and `ANG`) are the most sensitive to changes in the LR parameter, often severely degrading performance unless the unlearning LR is set lower than the training LR. In general, using a larger LR than that of training leads to poor LUMA, while the best performance is usually achieved with a rate close to the training value—particularly evident in the Image (CIFAR-100, CelebA) and Text (IMDB, AG News) domains. This makes the tuning of Unlearner dependent on LR quite simple, as the most reliable choice is always to select an LR comparable to the one used for training.

The only Unlearners that take as input a parameter different from the LR are the ones in the group of Weight Importance (WI): Fisher Forgetting (`FF`) and Selective

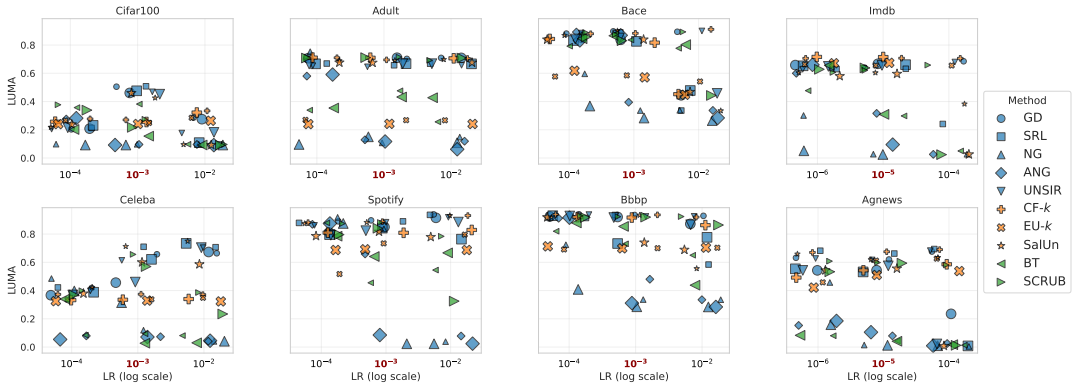


FIGURE 4.5: Performance of unlearning methods across datasets under varying learning rates. Each subplot represents a dataset. The marker size denotes the underlying model dimension (small vs. large), while the color indicates the unlearner family (FT, SWM, DIS, WI). The LR used to train the original model is highlighted in dark red.

TABLE 4.3: Comparison of Unlearners on the Image datasets trained on ResNet-50.

Group	Method	Cifar 100				Celeba			
		F1 test	UMIA	Runtime	LUMA	F1 test	UMIA	Runtime	LUMA
–	Orig.	.466 ± .007	.706 ± .028	3087.0 ± 442.3	.266 ± .046	.688 ± .009	.770 ± .011	8967.0 ± 744.0	.303 ± .027
–	Gold	.429 ± .003	.501 ± .003	3087.0 ± 442.3	.636 ± .000	.670 ± .009	.539 ± .008	8967.0 ± 744.0	.636 ± .000
FT	GD	.437 ± .001	.656 ± .034	112.3 ± 22.2	.461 ± .077	.578 ± .039	.535 ± .006	291.3 ± 31.8	.673 ± .117
	SRL	.438 ± .009	.649 ± .035	141.5 ± 21.8	.476 ± .072	.607 ± .011	.529 ± .005	291.8 ± 32.7	.734 ± .011
	NG	.001 ± .000	.499 ± .001	38.3 ± 6.1	.092 ± .004	.687 ± .004	.733 ± .006	1.5 ± .1	.397 ± .023
	ANG	.154 ± .011	.536 ± .016	224.1 ± 32.7	.282 ± .027	.216 ± .055	.750 ± .099	693.4 ± 26.1	.071 ± .019
	UNSIR	.436 ± .005	.652 ± .033	317.6 ± 29.6	.452 ± .071	.589 ± .044	.529 ± .008	294.1 ± 19.2	.704 ± .091
SWM	CF*k*	.462 ± .009	.693 ± .024	89.9 ± 18.7	.322 ± .029	.697 ± .001	.772 ± .008	250.1 ± 24.6	.340 ± .024
	EU*k*	.476 ± .004	.697 ± .027	1733.0 ± 313.3	.264 ± .031	.697 ± .002	.768 ± .006	1269.0 ± 78.4	.329 ± .013
	SalUn	.434 ± .021	.659 ± .048	153.2 ± 1.6	.460 ± .147	.682 ± .005	.620 ± .020	430.3 ± 199.6	.600 ± .059
DIS	BT	.074 ± .014	.651 ± .013	286.7 ± 5.2	.202 ± .042	.318 ± .091	.618 ± .021	744.4 ± 69.6	.340 ± .128
	SCRUB	.460 ± .025	.679 ± .038	143.2 ± 15.6	.339 ± .100	.648 ± .016	.638 ± .008	418.9 ± 126.4	.570 ± .019
WI	FF	.186 ± .161	.065 ± .922	2340.0 ± 121.8	.341 ± .294	.617 ± .033	.655 ± .024	205.6 ± 26.5	.575 ± .071
	SSD	.466 ± .007	.681 ± .030	120.9 ± .8	.316 ± .066	.688 ± .009	.773 ± .013	280.2 ± 40.3	.350 ± .040

Synaptic Dampening (SSD). In this case, the tuning was applied to their respective parameters. In the remainder of the paper, we only report the best results for each Unlearner.

The second key takeaway from Figure 4.5 is that the size of the model has little effect on the performance of the Unlearners, as they yield close scores across all *ED*s and, as a result, they are close in terms of LUMA. Unlearners perform similarly on models trained on a specific dataset, regardless of their sizes. This result is consistent across all domains.

As a result, in the remainder of the paper, we will only report results from the bigger of the two models.

4.3.2 Main Results

In Tables 4.3, 4.4, 4.5, and 4.6 we report the main results of our benchmark. All experiments were conducted three times on three different seeds, and the tables report the mean results along with their standard deviation. While LUMA is computed using the full set of measures described in Section 4.1, for the sake of space we only report one representative measure per *ED* (i.e., test F1 score, UMIA, and RunTime),

TABLE 4.4: Comparison of Unlearners on the Tabular datasets trained on MLP.

Group	Method	Adult				Spotify			
		F1 test	UMIA	Runtime	LUMA	F1 test	UMIA	Runtime	LUMA
–	Orig.	.793 ± .002	.498 ± .001	135.7 ± 7.9	.631 ± .000	.635 ± .009	.528 ± .005	119.6 ± 1.7	.573 ± .007
–	Gold	.791 ± .002	.499 ± .001	135.7 ± 7.9	.636 ± .000	.629 ± .002	.497 ± .003	119.6 ± 1.7	.636 ± .000
FT	GD	.791 ± .003	.498 ± .002	88.3 ± 4.9	.710 ± .001	.604 ± .007	.498 ± .004	4.7 ± .1	.915 ± .018
	SRL	.790 ± .002	.500 ± .000	11.2 ± 4.7	.668 ± .002	.634 ± .007	.521 ± .002	7.8 ± .1	.842 ± .012
	NG	.434 ± .000	.500 ± .002	16.2 ± .4	.150 ± .001	.604 ± .016	.522 ± .008	1.5 ± .0	.875 ± .006
	ANG	.767 ± .004	.499 ± .000	140.7 ± 1.5	.591 ± .006	.608 ± .009	.503 ± .003	11.9 ± .1	.875 ± .002
	UNSIR	.792 ± .003	.498 ± .001	105.2 ± .6	.668 ± .008	.603 ± .007	.499 ± .001	8.5 ± .2	.885 ± .012
SWM	CFk	.791 ± .003	.499 ± .000	85.5 ± 1.1	.711 ± .007	.642 ± .007	.519 ± .007	5.3 ± .1	.829 ± .011
	EUk	.793 ± .002	.498 ± .001	848.4 ± 15.6	.242 ± .010	.642 ± .007	.514 ± .007	59.6 ± .3	.692 ± .000
	SalUn	.786 ± .003	.500 ± .001	100.8 ± 11.9	.676 ± .014	.635 ± .009	.526 ± .006	8.3 ± .6	.817 ± .021
DIS	BT	.660 ± .106	.504 ± .005	156.1 ± .9	.432 ± .154	.583 ± .027	.531 ± .003	13.2 ± 1.0	.785 ± .029
	SCRUB	.789 ± .003	.498 ± .002	81.4 ± 1.0	.711 ± .007	.611 ± .013	.515 ± .007	7.6 ± .1	.841 ± .004
WI	FF	.786 ± .007	.500 ± .001	8.0 ± .8	.936 ± .013	.600 ± .028	.517 ± .008	19.4 ± .2	.812 ± .017
	SSD	.793 ± .002	.499 ± .001	91.5 ± .9	.697 ± .016	.635 ± .009	.518 ± .007	8.1 ± .1	.827 ± .017

together with the aggregated LUMA for all experiments. From these tables, we draw intra- (Section 4.3.2) and inter-domain (Section 4.3.2) observations.

Intra-domain observations

Image domain. Table 4.3 shows that the best-performing Unlearners in the image domain, according to LUMA, are SRL, GD, SalUn, and UNSIR, which achieve comparable results. This holds across both datasets, despite their different setups (CIFAR-100 for multiclass and CelebA for multilabel classification), indicating a degree of stability in image-domain Unlearners. This is not surprising, as the image domain is by far the most extensively studied. However, all these methods substantially increase UMIA relative to Gold (up to .659 and .623, compared to .501 and .539), which results in relatively low LUMA values (below 0.5 and lower than Gold). This highlights that the problem remains unsolved: current Unlearners have yet to match gold-level performance without compromising UMIA.

Tabular domain. Table 4.4 shows that FF is the clear winner on the Adult dataset, followed by GD and SSD. These methods also achieve strong performance on the Spotify dataset. In general, WI methods perform very well in this domain, due to the simpler architecture of MLPs. Both FF and SSD compute a Fisher Information Matrix over the network, which is reasonably accurate on an MLP ([134]). This is corroborated by the fact that Unlearners in the tabular domain obtain substantially higher LUMA values, reflecting the relative simplicity of this setting. Fine-tuning (FT) methods are also effective, except for NG, which consistently stays behind.

Graph domain. FT and DIS Unlearners perform best in the Graph domain, as shown in Table 4.5. All methods exhibit extremely low running times and relatively minor deviations from the F1 test, as GCNs are notoriously able to generalize better even with fewer samples ([268]), so they aren't impacted as much by the removal of the Forget Set, and the LUMA scores are very high across the board. This holds across both datasets, although it also highlights the difficulty of completely erasing information from the Gold model itself. Overall, unlearning for graph classification remains underexplored and warrants further investigation.

Textual domain. Textual domain is where SWM methods shine, as reported in Table 4.6. This is because BERT (or LLMs in general) usually fine-tune the last classification layer, while the bulk of knowledge is encoded within the deeper transformer architecture. As such, methods that selectively modify weights (CFk EUk,

TABLE 4.5: Comparison of the Graph datasets. Each entry reports mean ± std.

Group	Method	BACE				BBBP			
		F1 test	UMIA	Runtime	LUMA	F1 test	UMIA	Runtime	LUMA
–	Orig.	0.630 ± 0.004	0.525 ± 0.025	57.3 ± 27.8	0.564 ± 0.048	0.702 ± 0.003	0.501 ± 0.007	55.1 ± 0.2	0.617 ± 0.002
–	Gold	0.559 ± 0.044	0.528 ± 0.009	57.3 ± 27.8	0.636 ± 0.000	0.712 ± 0.007	0.498 ± 0.005	55.1 ± 0.2	0.636 ± 0.000
FT	GD	0.612 ± 0.021	0.519 ± 0.017	3.1 ± 4.0	0.851 ± 0.102	0.705 ± 0.005	0.499 ± 0.006	1.0 ± 0.0	0.924 ± 0.014
	SRL	0.609 ± 0.015	0.536 ± 0.017	4.0 ± 5.1	0.832 ± 0.089	0.710 ± 0.005	0.496 ± 0.004	1.4 ± 0.0	0.930 ± 0.013
	NG	0.385 ± 0.026	0.488 ± 0.011	0.2 ± 0.0	0.368 ± 0.033	0.485 ± 0.046	0.495 ± 0.001	0.3 ± 0.0	0.408 ± 0.126
	ANG	0.534 ± 0.028	0.496 ± 0.017	0.7 ± 0.0	0.869 ± 0.031	0.688 ± 0.012	0.498 ± 0.011	0.9 ± 0.0	0.916 ± 0.022
	UNSIR	0.617 ± 0.009	0.528 ± 0.006	1.0 ± 0.0	0.850 ± 0.097	0.705 ± 0.005	0.494 ± 0.006	1.3 ± 0.0	0.928 ± 0.009
SWM	CFk	0.612 ± 0.021	0.522 ± 0.017	3.2 ± 4.1	0.843 ± 0.112	0.705 ± 0.005	0.488 ± 0.002	1.1 ± 0.0	0.920 ± 0.008
	EUk	0.616 ± 0.021	0.537 ± 0.015	42.9 ± 0.2	0.618 ± 0.110	0.709 ± 0.001	0.492 ± 0.006	34.0 ± 0.0	0.713 ± 0.005
	SalUn	0.609 ± 0.015	0.536 ± 0.023	1.3 ± 0.0	0.839 ± 0.101	0.711 ± 0.007	0.497 ± 0.006	1.7 ± 0.1	0.919 ± 0.012
DIS	BT	0.570 ± 0.045	0.529 ± 0.041	2.2 ± 0.1	0.866 ± 0.011	0.659 ± 0.010	0.494 ± 0.008	2.9 ± 0.0	0.879 ± 0.028
	SCRUB	0.612 ± 0.021	0.523 ± 0.013	1.4 ± 0.0	0.857 ± 0.126	0.717 ± 0.008	0.498 ± 0.004	1.9 ± 0.0	0.920 ± 0.032
WI	FF	0.607 ± 0.020	0.526 ± 0.017	6.9 ± 0.1	0.830 ± 0.108	0.669 ± 0.066	0.505 ± 0.008	7.0 ± 0.0	0.793 ± 0.147
	SSD	0.630 ± 0.004	0.527 ± 0.009	1.2 ± 0.0	0.823 ± 0.099	0.702 ± 0.003	0.497 ± 0.003	1.6 ± 0.1	0.935 ± 0.006

TABLE 4.6: Comparison of Unlearners on the Textual datasets trained on BERT.

Group	Method	IMDB				AG news			
		F1 test	UMIA	Runtime	LUMA	F1 test	UMIA	Runtime	LUMA
–	Orig.	.937 ± .005	.549 ± .004	4914.0 ± 2965.0	.565 ± .006	.880 ± .012	.641 ± .030	5200.0 ± 54.7	.399 ± .056
–	Gold	.939 ± .006	.501 ± .000	4914.0 ± 2965.0	.636 ± .000	.906 ± .005	.525 ± .024	5200.0 ± 54.7	.636 ± .000
FT	GD	.941 ± .002	.539 ± .001	1329.0 ± 4.1	.667 ± .045	.908 ± .003	.568 ± .009	1655.0 ± 24.9	.544 ± .063
	SRL	.939 ± .000	.542 ± .003	1402.0 ± .2	.661 ± .043	.913 ± .001	.545 ± .006	1748.0 ± .4	.554 ± .061
	NG	.408 ± .090	.497 ± .000	75.9 ± 4.8	.051 ± .031	.570 ± .004	.852 ± .000	65.3 ± .1	.158 ± .028
	ANG	.935 ± .003	.502 ± .010	271.0 ± 8.4	.660 ± .056	.893 ± .002	.975 ± .001	3414.0 ± 8.4	.185 ± .006
	UNSIR	.933 ± .010	.538 ± .004	1478.0 ± 6.1	.656 ± .036	.910 ± .002	.554 ± .015	1814.0 ± .7	.575 ± .095
SWM	CFk	.938 ± .005	.549 ± .006	512.7 ± 2.9	.716 ± .038	.905 ± .005	.555 ± .017	599.1 ± 5.9	.586 ± .076
	EUk	.939 ± .004	.551 ± .002	1025.0 ± 2.5	.674 ± .042	.907 ± .004	.556 ± .014	1785.0 ± 17.2	.538 ± .059
	SalUn	.926 ± .000	.540 ± .000	1988.0 ± 0.0	.596 ± .000	.912 ± .001	.552 ± .006	2154.0 ± 1.8	.556 ± .047
DIS	BT	.890 ± .000	.540 ± .000	2427.0 ± 0.0	.656 ± .000	.289 ± .044	.784 ± .140	299.0 ± 78.3	.083 ± .027
	SCRUB	.935 ± .009	.543 ± .001	1904.0 ± 132.5	.638 ± .055	.888 ± .011	.563 ± .003	2258.0 ± 13.4	.594 ± .087
WI	FF	.500 ± .011	.498 ± .002	89.2 ± 3.3	.086 ± .000	.142 ± .058	.854 ± .007	.3 ± .0	.009 ± .001
	SSD	.941 ± .000	.555 ± .015	1492.0 ± 83.8	.642 ± .065	.879 ± .010	.654 ± .012	1793.0 ± 46.3	.440 ± .065

SalUn) remove task-specific information without harming the general representations captured by the transformer layers. Interestingly, WI Unlearners are unable to leverage the Fisher Information Matrix to correctly identify which weights to modify on LLMs, as shown by the low LUMA score for FF and SSD on AG news. Lastly, FT and DIS methods achieve good performance (with a few exceptions) but are held back by the non-trivial cost of further training a model of this size in its entirety.

Inter-domain observations

No single Unlearner (or group of Unlearners) dominates across all modalities, which reflects both the early stage of MU research and the ongoing search for more reliable methods. Still, some consistent patterns can be observed across domains.

Fine-tuning is often a safe strategy, so FT Unlearners are generally the most reliable, with the notable exception of NG, which often performs worst in terms of LUMA. Because the Forget Sets we define are heterogeneous (spanning multiple classes), a single epoch of negative gradient updates tends to collapse the model, leading to poor F1 scores on the Test Set. ANG, by contrast, mitigates this issue by also taking the Retain Set into account, and should therefore always be preferred.

Similarly, SWM Unlearners perform consistently well, but they rarely achieve the best performance. Among these, EUk is consistently the worst in terms of LUMA,

while SalUn is the most reliable across all domains. This aligns with its widespread adoption in the literature, where it often serves as a reference baseline.

On the other hand, both DIS and WI Unlearners show inconsistent performance. The former group performs best with smaller models (e.g., the tabular and graph domains). Still, it fails to scale to large pretrained architectures such as ResNet-50 and BERT, as distilling knowledge from such complex models is considerably more challenging, which is consistent with the literature ([176, 96]). The latter, which directly modify model parameters, are particularly effective given the simpler architecture of MLPs compared to CNNs (e.g., ResNet50) or LLMs (e.g., BERT), where kernels and attention mechanisms may introduce unintended side effects. Among these, SSD is generally more reliable.

4.3.3 Main takeaways

The main results of our benchmark can be summarized as follows: ***(i).*** Unlearners parameterized by learning rate for a semi-Newton step should adopt a value close to that used during training. Setting the learning rate too high leads to severe degradation of utility, effectively rendering the model unusable. ***(ii).*** Scaling up model size does not help: large models do not mitigate unlearning weaknesses. ***(iii).*** Unlearning remains fundamentally domain-dependent, as no method succeeds across all domains, although Fine-tuning Unlearners are the most reliable. ***(iv).*** A utility–efficacy trade-off is unavoidable with current methods (especially so in the Image domain), showing the field lacks methods that can forget without leaking information. ***(v).*** Distillation Unlearning is not scalable: it fails on large pretrained models, while selective weight methods are more reliable for these models. ***(vi).*** LUMA exposes hidden weaknesses: prior benchmarks overstated progress by ignoring efficiency and efficacy simultaneously.

4.4 Conclusion

In this work, we bring order to the landscape of Machine Unlearning methods for classification by providing: *(i).* the most comprehensive benchmark to date, covering 4 data modalities, 12 Unlearners, 8 datasets, and 8 models; *(ii).* the first systematic evaluation on the tabular and graph modalities; *(iii).* a taxonomy of Unlearner methods in the literature, *(iv).* a novel unified metric, LUMA, which quantifies performance as a single value to support hyperparameter tuning and unlearner selection in practice; and *(v).* a publicly available, fully reproducible, and extensible benchmark to facilitate fair comparison in future work.

Our results lead to clear guidelines for the design and deployment of Unlearners, and expose critical shortcomings that can only be revealed through cross-modality analysis. Together, these contributions establish a common ground for evaluating and comparing Machine Unlearning approaches for classification. By ensuring reproducibility and extensibility, our work provides a solid basis for future methods to be rigorously evaluated.

Chapter 5

Machine Unlearning for Graphs

While most existing MU methods focus on models trained on image or text, an increasingly important question is how they translate to structured domains such as graphs. Graphs are a natural representation for many privacy-sensitive systems, and Graph Neural Networks (GNNs) are often trained on sensitive graph data [227]. Social networks, for instance, offer an empirical example of unlearning: when a user requests the removal of their data from the system, they must comply ensuring that the user's information no longer influences models trained on that data.

> An important clarification concerns the distinction between graph-level and node/edge-level classification. Graph-level classification, discussed previously in Sections 3 and 4 as part of our benchmark, treats the entire graph as a single input instance and assigns a label to the whole structure (e.g., predicting a molecular property from a molecular graph).
> In contrast, this survey focuses on node- and edge-level classification, where the full graph is given as input, but the goal is to predict labels for individual nodes or edges. Although related, this setting poses different modeling challenges and evaluation considerations, making it a different but crucial problem. The bulk of the literature on Graph Neural Networks and Machine Unlearning for graphs relates to the latter problem.

Several approaches to Machine Unlearning for Graphs, i.e. Graph Unlearning (GU), have been proposed, leveraging the properties of structured data. The work by [43] was the first to address GU through shard-based retraining, while more recent advancements such as [48] introduced certified graph unlearning, providing formal guarantees for unlearning effectiveness.

However, the literature on GU is scattered and incomplete, suffering from three main problems: ***(i).*** The application of traditional MU methods to graphs remains unclear; ***(ii).*** The same small datasets (e.g., Cora [177] and Citeseer [107]) are often used to improperly infer generalizability; and ***(iii).*** the evaluation of GU methods is often incomplete and unreliable.

To address these gaps, in this survey, we first formalize a taxonomy of GU methods, encompassing both traditional MU methods and graph-tailored ones. We apply traditional MU methods to graphs and show weaknesses and applicability. Then, we show the problems that arise when results are generalized from small datasets, and how GU methods are often poorly evaluated, highlighting current problems in the literature and unexplored directions.

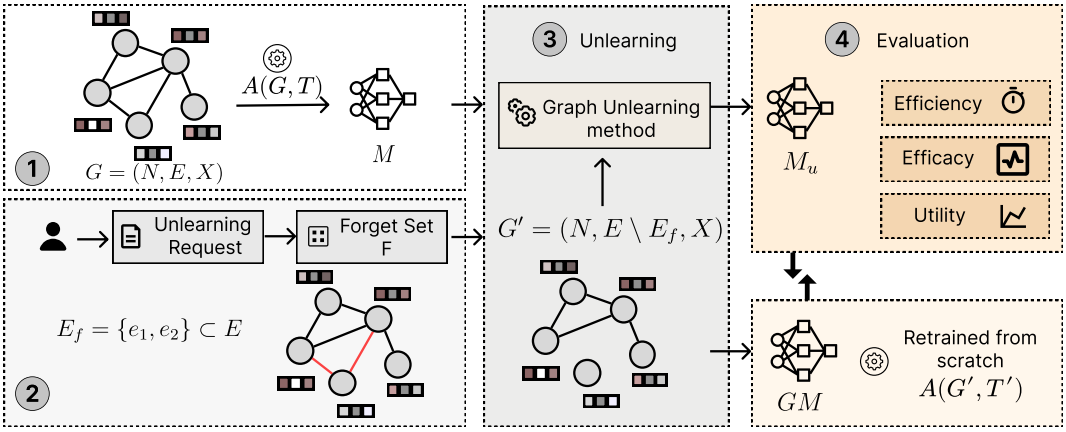


FIGURE 5.1: Graph Unlearning process. In Step 1, a model M is trained on a Graph $G = (N, E, X)$. Then, in Step 2, a user (or legal entity) files an Unlearning Request, formalized into a Forget Set $\mathcal{F}$ to remove from G, resulting in G' (in the example of the figure, $\mathcal{F}$ is a set of edges $E_f \subset E$). In Step 3, the Graph Unlearning method is applied and returns a model M_u. For evaluation (Step 4), a Gold Model M_G is retrained from scratch on G' and is compared against M_u on three evaluation criteria.

5.1 Related Work

Although MU is a relatively new topic, it has witnessed a surge of research activity in recent years. Several surveys have followed; however, most of them focus primarily on image and text modalities, with limited attention to graphs. The surveys of [157], [203] and [259] omit GU entirely, presenting instead broader MU conceptual aspects. Likewise, [265] only identifies graphs as a potential avenue for future research. The more recent survey by [163] does include graphs in a limited manner, but specific graph-tailored methods are not discussed nor evaluated. Similarly, most MU frameworks do not include GU [83] or only list it as future work [158].

To the best of our knowledge, the only survey on GU [215] is *unpublished* and includes several methods that, while applicable to graph-structured data, originate from different research communities (e.g., recommender systems). However, it does not provide a detailed taxonomy that captures key distinctions among GU methods, such as certified versus non-certified approaches. While their work offers an overview of a broad range of published (as well as non–peer-reviewed) methods, we are the first to: **(i)** introduce a more fine-grained and formally grounded taxonomy of GU methods, **(ii)** provide a unified mathematical notation that establishes common theoretical foundations, and **(iii)** systematically analyze and critique both the proposed methods and the evaluation protocols adopted in the GU literature.

5.2 Preliminaries

This survey, consistent with the included papers and the majority of the literature, focuses on node- and edge-level classification. Extensions to graph-level classification are straightforward but omitted for ease of notation. In this section, we introduce a unified notation that will be used throughout the Chapter to provide a common framework for the proposed taxonomy and for the field of GU in general.

Notation. We will refer to the parameter space as Θ, and the model hypothesis space as $\mathcal{H} := \{M_\theta : \theta \in \Theta\}$, where M_θ is the Graph Neural Network (GNN) parametrized by θ.

Let $\mathcal{G}$ be the graph space, $G = (N, E, X) \in \mathcal{G}$ a directed graph with node set N, edge set E, and node features X, and T a training set $T = \{(o, y_o) : o \in \mathcal{O}_T\}$ defined over G, where o denotes either a node $i \in N$ or an edge $(i, j) \in E$. Let also A be a randomized training algorithm that takes as input a graph $G \in \mathcal{G}$ and a training set T, and outputs the optimal model parameters $\theta^* \in \Theta$ by minimizing a given empirical loss function over the labeled objects in T:

$$\theta^* = \arg\min_\theta \mathcal{L}(\theta, G, T) = \arg\min_\theta \sum_{(o,y_o)\in T} \ell(M_\theta(o, G), y_o),$$

where $\ell(\hat{y}, y_o)$ is the loss function (often cross-entropy) computed on a single sample prediction $\hat{y} = M_\theta(o, G)$, and $\mathcal{L}(\theta, G, T)$ will be the empirical risk on all samples. For ease of notation, we will refer to M_{θ^*} simply as M .

Graph Unlearning. Figure 5.1 shows the GU process. In particular, Step 1 is about the training procedure as previously formalized. In Step 2, at any time after M was trained, a user or legal entity can file an unlearning request. The unlearning request is specified by a *forget set* $\mathcal{F}$, which may correspond either to a subset of nodes $N_f \subseteq N$ (*node unlearning*), a subset of edges $E_f \subseteq E$ (*edge unlearning*), or a subset of features $X_f \subseteq X$ (*feature unlearning*). In the example of Figure 5.1, $\mathcal{F}$ is subset of edges $E_f \subset E$.

The removal of $\mathcal{F}$ from G induces the *retain graph* G':

$$G' = \begin{cases} (N \setminus N_f,\ E \setminus E(N_f), X_{N\setminus N_f}), & \text{if } \mathcal{F} = N_f \text{ (node unlearning)}, \\ (N,\ E \setminus E_f, X), & \text{if } \mathcal{F} = E_f \text{ (edge unlearning)}, \\ (N, E, X \setminus X_f), & \text{if } \mathcal{F} = X_f \text{ (feature unlearning)}. \end{cases}$$

where $E(N_f)$ are the edges adjacent to nodes in N_f. Step 3 carries out the unlearning process. In the example, G' is obtained from a single edge unlearning request. By definition, the training set T will also change accordingly to G'. We refer to the altered training set as T', the training set induced by G': for node and edge unlearning, T' excludes removed objects; for feature unlearning, T' contains updated features.

Definition 2 (Gold model)**.** Given a randomized training algorithm A (used to train M) and a retain graph G', the *gold model* $M_G = A(G', T')$ is the model trained from scratch by running A on the retain graph.

The ultimate goal of a GU method $\mathcal{U}(M, G', \mathcal{F})$ can be formalized as follows:

Definition 3 (Graph Unlearning (GU))**.** Given a trained model $M = A(G, T)$, the retain graph G', and the forget set $\mathcal{F}$, find an updated model M_{θ_u} that closely approximates M_G. A GU method can be viewed as a function $\mathcal{U}$ that takes a trained model M, the retain graph G', and the forget set $\mathcal{F}$ as input, and outputs an updated model M_{θ_u}.

In Definition 3, although the retain graph is uniquely determined by the forget set, we explicitly include it for notational clarity and to facilitate the taxonomy.

In the following, we will refer to the unlearned model M_{θ_u} as simply M_u. Clearly, the gold model M_G is the reference output model: retraining the model from scratch on G' ensures that the influence of $\mathcal{F}$ has been unlearned, while maintaining the remaining knowledge from G. However, retraining the model after every (or batched)

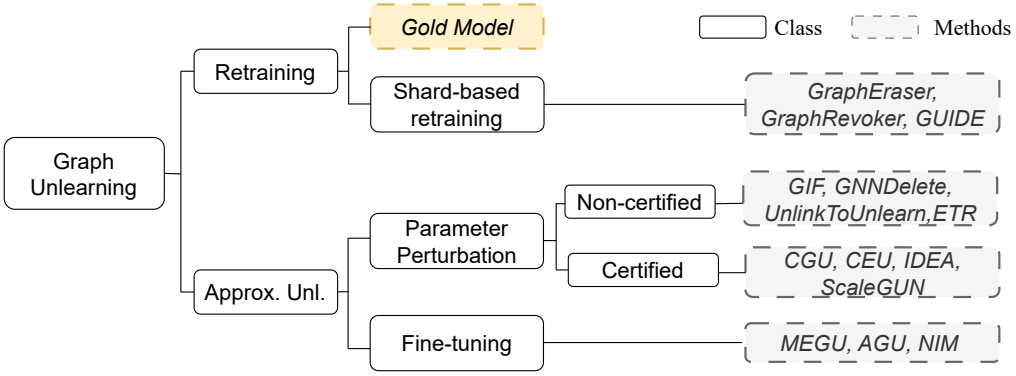


FIGURE 5.2: Proposed taxonomy of Graph Unlearning methods.

unlearning request is computationally expensive. For this reason, a GU method aims to approximate M_G as closely as possible while significantly reducing the computational burden.

Evaluating unlearning. The requirement that M_u *closely approximates* M_G can be formalized in terms of their functional similarity over the retain graph G'. While there is no consensus in the literature, most works employ proxy measures like the F1-score over the forget set and the test set to measure model utility after Unlearning and compare it to M_G.

In general, as per Step 4 of Figure 5.1, MU methods should be evaluated across three dimensions [121]:

- *Efficacy*: how well M_u unlearns $\mathcal{F}$,
- *Utility*: the utility of M_u on the remaining data,
- *Efficiency*: the computational cost of executing the method.

GU, as a subset of MU, follows the same evaluation principles, supplemented by graph-structured analyses and adversarial link inference attacks. Note that M_u is always evaluated against the gold model M_G.

Certified Graph Unlearning. Certified methods employ a definition similar to the one of differential privacy [85] to ensure formal guarantees on the closeness of M_u to M_G. Consider a randomized learning algorithm $A(G, T)$ on a graph G. It will output model parameters θ from a parameter space Θ (or equivalently, the induced model $M_\theta \in \mathcal{H}$). Similarly, a GU method $\mathcal{U}(M, G', \mathcal{F})$ will output model parameters in the same space Θ. We say that $\mathcal{U}(M, G', \mathcal{F})$ is (ϵ, δ)-certified if, for all forget sets $\mathcal{F} \subseteq G$ and all possible subsets $\Theta' \subseteq \Theta$:

$$(i) \quad P[\mathcal{U}(M, G', \mathcal{F}) \in \Theta'] \leq e^{\epsilon} P[A(G', T') \in \Theta'] + \delta$$

$$(ii) \quad P[A(G', T') \in \Theta'] \leq e^{\epsilon} P[\mathcal{U}(M, G', \mathcal{F}) \in \Theta'] + \delta$$

Conditions (i) and (ii) ensure distributional indistinguishability in both directions. This definition, adapted from [82, 48, 263], bounds the difference between the distribution of the parameters θ_u, obtained from $\mathcal{U}$, and the Gold Model M_G, obtained by retraining, by a small ϵ and relaxed by δ.

5.3 Method taxonomy

In this section, we describe, categorize, and discuss existing GU methods. Table 5.1 shows the methods for GU that are considered in this survey. The scope of this study is restricted to general-purpose GU and methods proposed in *top-tier venues*, providing an exhaustive coverage of the field. We exclude methods like UltraRE [161] that are restricted to recommender systems (e.g., bipartite graphs), and are generally treated as separate literature. Figure 5.2 presents the proposed taxonomy of GU methods. At the first level, methods are categorized by their unlearning strategy; at the second level, by the guarantees of their output. We now formally map each GU method to this taxonomy.

5.3.1 Unlearning by retraining

Traditionally, in the MU literature, *Exact Unlearning* refers to methods that guarantee that their output model is the same as the gold model M_G, i.e., the model retrained from scratch on the *retain graph*. Formally, $\mathcal{U}$ is an exact GU method if:

$$\mathcal{U}(M, G', \mathcal{F}) = M_G = A(G', T')$$

Clearly, obtaining M_G by retraining from scratch is an Exact Unlearning method by definition. However, guaranteeing exact unlearning while achieving computational efficiency is challenging, as such methods often depend on ad-hoc optimization strategies. Therefore, existing methods compromise exactness to enable efficiency.

Shard-based retraining. In classical MU, the SISA framework [27] defines the paradigm of training several models on an equal number of partitions of the original training set, and aggregating their individual predictions into a single combined output. The rationale for shard-based unlearning is that only the sub-model trained on the partition that contained the forget set needs to be retrained, saving computational time.

To formally adapt this framework to graphs, we consider a starting graph G that is partitioned into k disjoint shards $G_0, G_1, \dots G_k$. Then, k GNNs $M_0 = A(G_0, T_0), M_1 = A(G_1, T_1) \dots M_k = A(G_k, T_k)$ are trained on each individual shard, where $T_i = T \cap (N(G_i) \cap E(G_i))$. At inference time, we aggregate the predictions of the k models (with any aggregation function, e.g., majority voting): $M(x) = \mathcal{AGG}\big(M_1(x) \dots M_k(x)\big)$. Then, when a sample o (node or edge) is requested to be unlearned, only the model trained on the graph shard containing o is retrained. Formally, a shard-based retraining Unlearning method is defined as:

$$\mathcal{U}(M, G', \mathcal{F}) = \Big\{M_1, \dots, M_{j-1}, A(G'_j, T'_j), M_{j+1}, \dots, M_k\Big\}$$

where G_j is the shard containing o, and the updated ensemble predictor becomes:

$$M_u(x) = \mathcal{AGG}\big(M_1(x), \dots, A(G'_j, T'_j)(x), \dots, M_k(x)\big),$$

In the literature of GU, GraphEraser [43] and GraphRevoker [274] have adapted this approach to graphs, outlining three common steps: ***(i).*** Balanced Graph Partition, ***(ii).*** Shard model training, and ***(iii).*** Shard model aggregation. However, partitioning a graph into shards is challenging: inter-shards edges are dropped to ensure that each node's influence is confined to a single sub-model, and consequently preserving the structural properties of the graph is a non-trivial task.

GraphEraser only considers structural properties like node neighborhood for the sharding policies, which can lead to shards with highly skewed label distributions [274] and, as a result, biased sub-models. GraphRevoker fixes this problem by designing a loss function that preserves label semantics and results in more balanced graph shards. Together with losses minimizing unlearning time and inter-shards edges, GraphRevoker creates better shards for step ***(i)***. Lastly, GUIDE [256] argues that GraphEraser's partitioning mechanism is too computationally extensive and only refers to the transductive setting (where test nodes and edges are visible during training). Driven by these motivations, GUIDE **(i).** refers to the inductive graph setting (where test nodes and edges are not visible during training) and **(ii).** proposes a fairness-driven (in terms of statistical parity [256]) partitioning schema for graph sharding.

While these methods cannot guarantee that they output the same model as M_G, and as such cannot be categorized as *Exact Unlearning* methods, they retrain the affected sub-models from scratch, and they guarantee that the influence of the Forget Set $\mathcal{F}$ is removed entirely from the output model, as if it was never in the original graph G.

5.3.2 Approximate Unlearning

With *Approximate Unlearning* methods we refer to methods that output a model M_u that is different than, although closely approximates, the Gold Model M_G. Unlike retraining-based methods, they do not guarantee that M_u will behave as if $\mathcal{F}$ was never in G, opting instead to provide improved efficiency. We classify Approximate Unlearning methods for GU in two subclasses: *Parameter Perturbation* and *Fine-tuning*.

Parameter Perturbation. The methods of this class directly perturb model parameters (weights).

Given a model $M_{\theta^*} = M$ (refer to Section 5.2), these methods compute a perturbation matrix $\mathcal{P}$ on the weights θ^* and produce an unlearned model parametrized by $P + \theta^*$, i.e.,

$$M_u = M_{\mathcal{P}+\theta^*}, \quad \mathcal{P} \in \mathbb{R}^{|\theta^*|}$$

We divide these methods in Non-certified and Certified methods, based on guarantees on their output, as per figure 5.2.

Non-certified. Non-certified methods employ several different strategies for unlearning. GIF (Graph Influence Function) estimates the parameter change inferred from the removal of the Forget Set thanks to influence functions. First, they show that the influence function definition can be used to estimate parameter change before and after unlearning. Let θ^* and θ_u of M and M_u (before and after unlearning), respectively. Then, assuming the loss $\mathcal{L}$ on the original T is convex and twice-differentiable, they estimate the parameter change as: $\theta_u - \theta^* = H_{\theta^*}^{-1} \nabla_\theta \mathcal{L}_\mathcal{F}$, where $\mathcal{L}_\mathcal{F} = \sum_{i \in \mathcal{F}} \ell(M(o, G), y_o)$ [262] [1]. However, this formulation implies that perturbing sample o does not change the state of any other sample. Consequently, to accommodate the graph's relational structure, they extend the traditional influence function with a loss term $\mathcal{L}_{G'}$ that accounts for the influence of neighboring nodes, with the final GIF perturbation being:

$$\mathcal{P}_{\text{GIF}} = \theta^u - \theta^* = H_{\theta^*}^{-1} \nabla_\theta \mathcal{L}_{G'},$$

[1] For lack of space, we redirect the interested reader to the original paper for complete proof.

although they estimate the inverted Hessian for efficiency.

GNNDelete, on the other hand, only focuses on edge unlearning and uses another approach: instead of influence functions, $\mathcal{P}$ is a learnable operator. Let $\mathcal{F} = E_f = (n_i, n_j)$ be the forget set and S_{ij}^k be the enclosing subgraph around nodes n_i and n_j. Then, they extend the traditional GNN layer g^l with the DEL^l operator with a multi-layer perceptron they train on two losses: minimizing the differences of between-node pair representations of every node pair to avoid edge leakage (Deleted edge consistency) and removing the influence of (i, j) from the neighborhood (Neighborhood influence) [47].

As such, GNNDelete defines the perturbation $\mathcal{P}$ as the composition of layer-wise deletion operators, i.e., for a network with m GNN layers,

$$\mathcal{P}_{\text{GNNDelete}} = DEL^0 \circ DEL^1 \circ \cdots \circ DEL^m.$$

In contrast, UnlinkToUnlearn adopts a lightweight strategy that does not explicitly alter the model parameters. Instead, it performs inference on the modified graph G' on the same model M. Although the underlying parameters θ^* remain unchanged (so $\mathcal{P}$ is the identity function), this method modifies the functional behavior of the model by altering its message-passing context and is an implicit parameter perturbation, since the change arises from the altered propagation paths rather than direct parameter updates [241].

Lastly, Erase Then Rectify (ETR) is built on two steps. During the Erase step, they modify the parameters via element-wise scaling based on their importance given by the Fisher Information Matrix [269], and specifically by dampening parameters that are more informative for the unlearned data and its neighborhood. Then, the Rectify phase is a simple gradient descent step on the retain graph. Formally,

$$\mathcal{P}_{\text{ETR}} = \underbrace{(s - \mathbf{1}) \odot \theta^*}_{\mathcal{P}_{\text{Erase}}} + \underbrace{\left(-\lambda \nabla_{\tilde{\theta}} \mathcal{L}_{G'}\right)}_{\mathcal{P}_{\text{Rectify}}}$$

where $s \in (0, 1]^{|\theta|}$ is a parameter-wise scaling vector with each $s_j \in (0, 1]$ (computed from the ratio of values of the Fisher Information Matrix from affected and unaffected nodes) and $\tilde{\theta}$ denotes the post-Erase parameters. ETR is the first work leveraging the Fisher Information Matrix in GU, which is surprising, given its widespread use in general MU.

Certified. All the methods in this subclass provide formal guarantees that the retrained model is reasonably close to the Gold Model (refer to Section 5.2). All the Certified methods currently present in the literature employ Quasi-Newton updates to estimate the optimal parameter shift using second-order information to guarantee certified unlearning. In a very similar manner to GIF (which frames it as computation of influence functions), these methods try to find $\mathcal{P}$ to move the weights from $\theta^* = A(G, T)$ to $\theta_u = A(G', T')$, i.e. where the loss would have been optimized if the removed sample was never in the original graph.

Specifically, CGU introduces the certified Graph Unlearning framework by extending the work on certified Unlearning [116] to graphs. Starting from the traditional certified Unlearning step with regularization $\lambda, \mathcal{U}(M, G', \mathcal{F}) = \theta^* + H_{\theta^*}^{-1} \lambda \theta^* + \nabla \ell(M(n_i, G), y_i)$, they generalize it to the graph setting by replacing the third term with $\Delta = \nabla_{\theta^*} \mathcal{L}(\theta^*, G) - \nabla_{\theta^*} \mathcal{L}(\theta, G')$. The idea is, on the surface, very similar to GIF: incorporating the structural properties of graphs into the loss function, weighted by

the Hessian, to move the weights closer to where they would be without the sample i in G. The perturbation matrix $\mathcal{P}$ defined by CGU is:

$$\mathcal{P}_{\text{CGU}} = H_{\theta^*}^{-1}\lambda\theta^* + \nabla_\theta\mathcal{L}(\theta^*, G) - \nabla_\theta\mathcal{L}(\theta^*, G')$$

The key difference is that GIF explicitly aggregates the losses of both removed and structurally influenced nodes, whereas CGU captures those effects implicitly through the propagated embeddings. However, CGU only works with SGC models, a simplified version of GCNs without non-linear activations. This allows CGU to also compute practical certified bounds: the Quasi-newton step they define comes with guarantees on the "closeness" of the unlearned M_u with the Gold Model M_G [48].

CEU builds on top of CGU by: ***(i).*** Speeding the computation significantly, by estimating the inverse Hessian rather than computing it like CGU does, and ***(ii).*** Extending the approach to GNNs with non-linear activations. However, their approach is only applicable to Edge Unlearning.

CEU is a two-step process. In Step 1, they add perturbation to the loss function, to hide the gradient residual (masking any leakage from removed edges), thus providing the certified unlearning guarantee. In Step 2, CEU applies influence analysis to estimate the impact of the deleted edges and "reverse" their impact, similarly to GIF, defining $P_{CEU} = \frac{1}{N(G)}\mathcal{I}_{E_\mathcal{F}}$, where the term $I_{E_\mathcal{F}}$ estimates the influence of the edges in the forget set $E_\mathcal{F}$. It does so by computing the gradient difference of the perturbed loss on the nodes affected by the edge removal on the original model, propagating through the original graph and the retain graph.

IDEA [82] generalizes the quasi-Newton-style updates introduced in CGU/CEU, and makes them flexible and certifiable across all GNNs and unlearning requests. Specifically, given the optimized parameters θ^* and θ_u defined as in Section 5.2, IDEA defines a formalization for the intermediate state $\tilde{\theta}_{\mathcal{F},\xi} = \theta^* + \xi(\mathcal{L}_{\text{add}} - \mathcal{L}_{\text{sub}})$, where $\mathcal{L}_{\text{sub}} = \sum_{n_i\in\phi(\mathcal{F})}\ell(M(n_i, G), y_i)$ removes the loss of the affected nodes, as $\phi(\mathcal{F})$ returns the nodes affected by the unlearning request, and

$$\mathcal{L}_{\text{add}} = \sum_{n_i\in\phi(\mathcal{F})}\ell(M(n_i, G'), y_i)$$

adds back the loss in the retain graph of the affected neighbors. In their definition, ξ balances the effect of the second term, and if $\xi = 1/|T|$, the inverse cardinality of the training set, they show that $\tilde{\theta}_{\mathcal{F},\xi} = \theta_u$. So, they are finally able to approximate their $\mathcal{P}$ as

$$\mathcal{P}_{\text{IDEA}} = \frac{1}{|T|}(-H_{\theta^*}^{-1}(\nabla\mathcal{L}_{\text{add}} - \nabla\mathcal{L}_{\text{sub}}))$$

which produces only infinitesimal residuals, providing formal certification guarantees [82].

This is not different from what all other Parameter Perturbation methods do: estimating the loss difference propagated through affected nodes, and then execute a quasi-Newton update towards that direction through an inverted Hessian. The difference in methods is mostly in the approximation of such inverted Hessian and the estimation of the loss difference. Certified methods also derive a formal guarantee of "closeness" between M_u and M_G, either by residual computation (CGU,IDEA) or by applying Gaussian noise (CEU).

Lastly, ScaleGUN [272] uses the same unlearning mechanism as CGU, but makes it scalable via an embedding approximation method leveraging a lazy propagation

mechanism. ScaleGUN notably highlights an important weakness that we also discuss in Section 11.4: certified methods (and most GU methods in general) are only tested on very small datasets. However, ScaleGUN is only applicable and tested on linear models like SGC (refer to Table 5.1), undermining its real-world applicability, especially when compared to IDEA, which is tested on non-linear model architectures.

Fine-tuning. The last group of methods continue the training of M to remove the influence of $\mathcal{F}$ or further specialize the model on G'. Let A be the training algorithm resulting in M, as per Section 5.2. These algorithms define $\mathcal{U}(M, G', \mathcal{F}) = A^+ : (\theta^*, G', \mathcal{F}) \rightarrow M_{\tilde{\theta}} \in \mathcal{H}$ as training algorithms that start from θ^*. This paradigm is vastly used in traditional MU.

MEGU was the first work to apply it to Graph Unlearning. It introduces a Linear Unlearning Module (LUM) in addition to the original model. When an unlearning request comes, the original model M is frozen and only provides output Y_{fr}. The predictive model M_p is an unfrozen copy of M that is trained on the empirical risk:

$$\mathcal{L}_p = \sum_{n_i \in HIN} \ell_{CE}\big(M_p(n_i, G),\, M_{\text{fr}}(n_i, G)\big) + \sum_{n_j \in HIN} \ell_{KL}\big(\text{LUM}(n_j, G),\, M_p(n_j, G)\big).'$$

where HIN is the set of highly influenced nodes from the unlearning request [160]. This loss keeps the predictive model accurate, but removes the related knowledge in HIN via KL-divergence loss. Similarly, the LUM is trained via the sum of negative cross-entropy with the frozen model (to diverge from it) and a KL loss from M_p (to maintain accuracy), resulting in $\mathcal{L}_{LUM}$. The final empirical risk from MEGU is $\mathcal{L}_{\text{MEGU}} = \mathcal{L}_p + k\mathcal{L}_{\text{LUM}}$, with k weighting the unlearning loss.

AGU employs a similar fine-tuning approach by proposing a combined loss of ***(i)***. The Deleted Edge Consistency, defined by GNNDelete [47], extended to the common neighbors of the edges in the forget set, resulting in $\mathcal{L}_{EU}$, and ***(ii)***. The feature unlearning loss $\mathcal{L}_{FU}$ as the negative KL divergence w.r.t. the original model on the nodes whose features are unlearned. The first term $\mathcal{L}_{\text{EU}}$ is used for edge unlearning, the second term $\mathcal{L}_{FU}$ for feature unlearning, and their weighted sum is used for node unlearning.

Finally, although early stages, Node Influence Maximization (NIM) is a method that finds the most influential nodes in a Graph, and proposes a larger Scalable Graph Unlearning (SGU) framework that only fine-tunes on the entities that are highly influenced (HIE) by the forget set, found by NIM. The authors claim that fine-tuning on a small set of entities is empirically faster and preserves the knowledge of the model.

5.4 Discussion

We now proceed to critically evaluate the methods presented in Section 5.3. We first infer and discuss literature gaps from the proposed taxonomy. Then, we focus on highlighting evaluation pitfalls present in the literature, and propose desiderata for future research.

5.4.1 Discussion on the taxonomy

Limited methodological diversity. The most glaring takeaway message from Table 5.1 and Section 5.3 is that parameter perturbation methods, along with quasi-newton updates, dominate the scene. All methods in these categories formally derive the distance between M and M_G with a propagated loss through the entities in the graph that are affected by the Forget Set, then map that difference as a parameter shift through an inverted Hessian. While the certified methods are able to provide formal guarantees of removal, their mechanisms are similar within each other and with methods like GIF [262], relying on quasi-Newton updates.

Moreover, certified methods either severely limit their scope (by only considering linear networks) or cannot scale to larger graphs [272]. Although each work is trying to define its own extensible GU framework, the difference between the methods is often only in how they compute and propagate the loss across the affected graph structure. Notable exceptions are GNNDelete, which proposes a learnable operator, and CEU, which achieves certified unlearning through Gaussian noise, similarly to works on the related problem of differential privacy [85]. Several alternative paradigms that have shown effectiveness in MU (e.g., saliency-based modification [95]), have not been investigated for GU.

Sharding is underinvestigated. GraphEraser [43] introduced the SISA framework to graphs, while GUIDE [256] and GraphRevoker [274], evolved the sharding strategy to optimize fairness and efficiency. However, no other work in the literature studied shard-based retraining for GU, nor any other retraining or exact unlearning strategy. This is surprising, as SISA-based retraining strategies are widely adopted in traditional Machine Unlearning [265]. Methods in this category largely overlook the extensive graph partitioning literature and fail to optimize partitioning objectives specifically for unlearning. As a result, it remains unclear how partitioning functionally facilitates unlearning, and whether such partitions can be maintained or updated when unlearning requests involve nodes or edges that cross partition boundaries.

5.4.2 Evaluation pitfalls

Evaluation dimensions. As discussed in Section 5.2, MU methods should be evaluated through three evaluation dimensions (also reported in Table 5.1): Efficacy (how well the model unlearns), Utility (the model utility after unlearning), and Efficiency (the computational cost of unlearning). All these dimensions are crucial: works that do not include a comprehensive evaluation across all three (i.e., GraphEraser, CGU, GUIDE, GraphRevoker, as per Table 5.1) provide incomplete evaluation of their methods. In traditional MU, Efficacy is often measured via the Unlearning Membership Inference Attack (UMIA) [121], which aims to infer whether the model can still leak some of the knowledge that needs to be forgotten. For graphs, this attack can be extended to Link Inference attacks, which aim to infer knowledge about removed edges. While most GU works adopt some form of such attacks, they frequently rely on different threat models, attack implementations, and evaluation protocols, making direct comparison across methods difficult.

More broadly, the field lacks standardized benchmarks and evaluation frameworks for GU. The absence of unified evaluation pipelines, and agreed-upon attack protocols has led to fragmented experimental practices and duplicated evaluation efforts across studies.

Datasets. As detailed in Table 5.1, all works test their methods on the same common datasets, such as Cora and Citeseer, with some methods (i.e., GIF and CEU) critically employing almost exclusively these two datasets, and then inferring generalizability of results from them. We now show how this is improper based on two key factors: ***(i).*** It is well known in the literature [226] that removing a small percentage of edges does not hinder (or, at times, even improves) model accuracy on GCNs for the Cora and Citeseer datasets and ***(ii).*** training times cannot be inferred from such small datasets. In fact, most works that use Cora and Citeseer will show that retraining the model on these datasets from scratch only takes $\sim$ 2 to $\sim$ 4 seconds. Unlearning advantage in this scenario is unclear; differences could be related to statistical variation or technical reasons, like Pytorch's startup lag. Execution time is the critical motivation behind MU and [272] show that other certified methods (CEU, CGU, IDEA) actually take longer than retraining on bigger graphs. Given that retraining the Gold Model is the ideal solution, if a GU method takes longer, then it provides no practical advantage (refer to Section 5.2). Using small datasets to infer generalizability on efficiency scaling is improper and could be misleading; GU methods should only be tested on very large graphs or, alternatively, report less machine-dependent measures (e.g., number of operations, memory).

5.4.3 Evaluation protocol desiderata

Given the taxonomy of Figure 5.2 and the pitfalls discussed so far in this Section, we now move to highlight the desiderata of evaluation protocols for GU methods through easily actionable guidelines.

i. Prioritize large-scale graph datasets. GU methods should be evaluated primarily on large-scale graphs, for which retraining from scratch is non-trivial and unlearning efficiency becomes meaningful. As discussed, reliance on small benchmark datasets such as Cora and Citeseer is problematic. Future research should prioritize datasets such as ogbn-arxiv, papers, and photos [2].

ii. Evaluation must span Efficacy, Utility and Efficiency. GU methods, like MU methods as per [279], should be evaluated for their unlearning efficacy, the utility of the resulting model, and their running time or general efficiency. Any evaluation that omits one or more of these dimensions is inherently incomplete. Efficacy should be assessed using well-established and reproducible link inference or membership inference attacks.

iii. Testing on the retain graph. Under the GDPR (and related regulations), all copies of personal data subject to a removal request must be permanently deleted.[3] Consequently, the retain graph G' should be the only graph used for both training and evaluation after unlearning. However, it remains unclear whether this requirement is met in existing works.

iv. From proliferation to consolidation. GU has reached a level of maturity where further progress is unlikely to come from introducing new end-to-end unlearning frameworks built on the same core principles. A large fraction of existing methods differ in implementation details or strategies, but largely explore the same algorithmic design space. The field of GU is no longer bottlenecked by the lack of methods, but rather by the absence of standardized evaluation protocols and scalable benchmarks. Future research should prioritize consolidation over proliferation: understanding when and at what cost GU is achievable in practice.

[2] https://ogb.stanford.edu/docs/nodeprop/

[3] https://gdprinfo.eu/en-article-17

5.5 Conclusion

In this work, we established a unified formal framework for Machine Unlearning for Graphs, a field rapidly growing in research attention; we surveyed existing works and organized them into the first formal taxonomy of the field. Then, we highlighted what is currently a gap in the literature, inferring that **(i).** most of the work on GU is duplicated effort with very similar methods, **(ii).** Methods are not being evaluated fairly; as some are omitting critical evaluation dimensions and there is no standardized link inference attack to evaluate *efficacy*, and **(iii).** All works are improperly generalizing results obtained on very small datasets, which can be problematic. Critically, some works only rely on these datasets, and certified unlearning methods take longer than retraining on bigger datasets, defeating their purpose as unlearners. Lastly, guided by these gaps, we outline desiderata of standardized benchmarks to guide future research in the field.

Paper	Name	Strategy	Datasets tested	Models	Cert.	E.cacy	Utility	Eff.
[43]	GraphEraser	Shard-based retraining	Citeseer, Cora, CS, Physics, Pubmed	GAT, GCN, GIN, GraphSage	✗	✗	✓	✓
[48]	Certified Graph Unlearning (CGU)	Quasi-Newton step	Citeseer, Computers, Cora, OGBN-arxiv, Photos, Pubmed	SGC	✓	✗	✓	✓
[262]	Graph Influence Function (GIF)	Parameter modification (via influence function)	Citeseer, Cora, CS	GAT, GCN, GIN, SGC	✗	✓	✓	✓
[263]	Certified Edge Unlearning (CEU)	Quasi-Newton step	Citeseer, Cora, CS	GCN, GIN, GraphSage	✓	✓	✓	✓
[47]	GNNDelete	Parameter modification (via learnable operator)	Cora, CS, DBLP, OGB-Collab, Pubmed / OGB-BioKG, WordNet18RR	GAT, GCN, GIN	✗	✓	✓	✓
[256]	GUIDE	Shard-based retraining	Bitcoin, Citeseer, Cora, CS, DBLP	GAT, GCN, GIN, SAGE	✗	✗	✓	✓
[274]	GraphRevoker	Shard-based retraining	Citeseer, Cora, Flickr, LastFM Asia	GAT, GCN, GraphSage, AppNP, JKNet	✗	✗	✓	✓
[160]	MEGU	Fine-tuning (Combined losses)	Citeseer, Cora, Computer, CS, Physics, Photo, Pubmed, PPI, Flickr	GAT, GCN, Graph-SAGE, SAINT	✗	✓	✓	✓
[241]	Unlink to Unlearn	Removal at inference	Corafull, CS, OGB-Collab, Pubmed	GAT, GCN, GIN	✗	✓	✓	✓
[82]	IDEA	Quasi-Newton step	Cora, Citeseer, CS, Physics, Pubmed	GAT, GCN, GIN, SGC	✓	✓	✓	✓
[80]	Adaptive Graph Unlearning (AGU)	Fine-tuning (Combined losses)	Citeseer, Cora, Computer, CS, Photo, Pubmed, Flickr	GAT, GCN, GIN, SAGE, SGC	✗	✓	✓	✓
[159]	Node Influence Maximization (NIM)	Fine-tuning (via influence function)	arxiv, Citeseer, Computer, Cora, Flickr, Reddit, Papers100M, Photo, PPI, Products, Pubmed	GCN, GIN, GraphSAGE†	✗	✓	✓	✓
[269]	Erase Then Rectify (ETR)	Parameter Modification (via Fisher Information)	arxiv, Citeseer, Cora, CS, Physics, Products, Pubmed	GAT, GCN	✗	✓	✓	✓
[272]	ScaleGUN	Quasi-Newton step	arxiv, Citeseer, Cora, Papers, Photo, Products	SGC	✓	✓	✓	✓

TABLE 5.1: Comparison of graph unlearning methods in temporal order. Each approach is evaluated by datasets, models, and four assessment dimensions: *Certified*, *Efficacy*, *Utility*, and *Efficiency*. † denotes that only prevalent GNNs are reported, as defined in their paper.

Chapter 6

Beyond the state of the art: Forget Set Identification

MU has become an effective solution to address increasing demands for data privacy and regulatory compliance, particularly in the context of the "Right to be Forgotten" [75, 154], which grants individuals the request to remove their personal information from the systems of any service provider [154]. Attention to this context has been further fostered by regulations such as the AI Act [90] and the General Data Protection Regulation (GDPR) [174].

However, the use of MU goes well beyond privacy contexts. In fact, machine-learning systems are increasingly trained on massive datasets scraped from the Internet or collected from diverse sources. These datasets often contain malicious, harmful, or "not safe for work" content as well as inherent biases [22, 106]. These biases can lead to unfair treatment of certain groups of individuals or to the propagation of discriminatory behavior within models, raising significant ethical concerns [273, 44]. Furthermore, including such a content in machine-learning models poses legal risks, particularly when these models are deployed in real-world applications.

Motivation. Machine Unlearning methods are typically formulated so that the subset of training samples to be unlearned, commonly referred to as the "*forget set*", is provided as input. The main goal in MU is to come up with a new model which is as much equivalent as possible to the model that would be obtained by retraining the original one on the training set resulting from the removal of the forget set from it. This way, the resulting model is also expected to keep its performance on what is called the "*retain set*", which typically corresponds to the difference of the training set and the forget set. All of this needs to be ideally carried out without retraining the original model from scratch [265].

However, the requirement of having the forget set as input is so restrictive that it might prevent MU from being profitably applied in many real-world scenarios. For instance, individuals requesting data deletion may lack sufficient technical expertise to define the forget set by themselves [255]. Also, it is highly likely that they have no access to the training data, as illustrated in Figure 6.1. In some other cases, the evidence to be unlearned is simply too complex to be directly expressed as a forget set (see applications below). In a more realistic setting, users know what needs to be unlearned in a more general or abstract sense, rather than as a specific subset of training samples. In cases like these, in order to be able to execute MU methods, one needs to preliminarily *identify the forget set based on user's unwanted evidence* (Step 3 of Figure 6.1). This is a crucial and challenging task, which, to the best of our knowledge, has never been tackled so far.

State-of-the-art advancement. In this work, we fill this important gap, and study for the first time what we call the FORGET SET IDENTIFICATION problem (for short, FORSID): find the forget set D_f, such that its removal from the original training set D

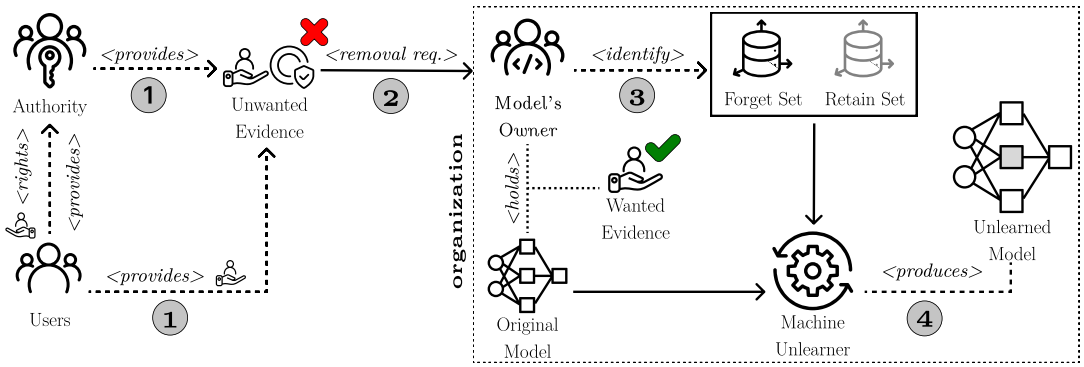


FIGURE 6.1: Overview of the overall Machine Unlearning (MU) process. Directly or indirectly (e.g., passing through some authority), users exercise their right to be forgotten (Step 1). To this end, in Step 2, they provide unwanted evidence that is required to be unlearned through a removal request to machine-learning model's owner. Technically, in Step 3, the organization (that is supposed to hold the wanted evidence to be maintained too) identifies the forget set, as a subset of the upstream training set, and, optionally, a retain set. In Step 4, a MU technique is applied to the original model so as to produce the desired one where user's unwanted evidence has been unlearned. So far, in the MU landscape, Steps 1–3 have been treated as a unique one, i.e., it has been assumed that users provide their unwanted evidence directly in the form of a forget set. This is unrealistic in many scenarios. The Forget Set Identification (ForsID) problem we introduce in this work is aimed at overcoming precisely such a limitation, providing a principled way for identifying a forget set from more abstract forms of unwanted evidence (Step 3).

ensures the unlearning from an existing model M_D trained on D of all user-specified unwanted evidence D_u, while preserving the predictions for organization's desired cases D_w. This way, FORSID identifies the samples to be removed by analyzing model's predictions for both unwanted and wanted examples.

As such, FORSID is an important complementary problem to MU, to be used coupled with (and not in substitution for) MU methods. Specifically, FORSID serves as a foundational step to be performed prior to executing MU methods, allowing for extending the scope of MU to scenarios where directly providing the training samples to be removed is not possible or too difficult.

Applications. The proposed FORSID problem finds natural application in several real-world contexts. For instance, consider an end-user's service based on an image classification system. Suppose that the service provider is receiving a request to forget specific images (e.g., images of individuals who want to exercise their right to be forgotten, or outdated images). The user who performs such a forgetting request likely does not have access to the training set, for privacy reasons or simply because that training set is proprietary to the provider and/or it has been anonymized. Thus, the user cannot provide a specific forget set to perform the unlearning task directly. Instead, the user can provide unlearning evidence in a more abstract form, e.g., examples of self-images (not included in the training data) that indicate broadly what needs to be unlearned, so that identifying a proper forget set corresponding to the desired unlearning evidence will be deferred to our FORSID problem the task.

A similar scenario arises in the context of AI systems trained on copyrighted artworks. Here, an artist may request that the model forget their style or specific artworks. However, they often do not have access to the exact training data used, nor do they know precisely which of their works were included. Instead, they can only provide examples of their art as unlearning evidence, requiring the system to identify which training samples correspond to this evidence to ensure compliance with the forgetting request.

More generally, the limitation of requiring a forget set arises in many MU scenarios where the exact data to be removed is unknown, whereas the undesired behavior is well understood. For instance, when regulations or other forms of sensitive documents need to be discarded from natural-language processing systems despite the lack of precise knowledge about which training data contributed to their learning.

Technical contributions. We provide a combinatorial-optimization formulation of the FORSID problem and show its **NP**-hardness through a reduction from the KNAPSACK problem [137]. We then establish a connection between FORSID and RED-BLUE SET COVER (RBSC) [36], a variant of the well-known SET COVER problem. Particularly, we devise a novel, weighted-coverage-based formulation of RBSC, which is dubbed RED-BLUE FORSID and shown to constitute an effective and efficient proxy for the original FORSID problem. In formulating RED-BLUE FORSID, a major technical challenge that we achieve is to come up with a notion of coverage that can effectively express the desiderata of FORSID. Defining the RED-BLUE FORSID problem mainly serves the purpose of facilitating algorithm design for the FORSID problem. In fact, the ultimate algorithm we propose to tackle FORSID, termed ForSId-via-RBSC, consists of solving the proxy RED-BLUE FORSID problem.

In this paper, we introduce a novel research line in the MU landscape that has not been considered so far. To support further development of this area, we release a codebase,[1] which contains all the code for full reproducibility of methods and experiments of this paper, along with an easily extensible framework which allows

[1] https://github.com/dangeloandrea14/Forget_Set_Identification

researchers and practitioners to seamlessly integrate their own approaches.

Summary and roadmap. To summarize, in this work:

- We study for the first time FORGET SET IDENTIFICATION (FORSID), i.e., the problem of identifying a forget set D_f as a subset of a training set D utilized to train a model M_D whose removal from D allows for unlearning unwanted evidence D_u from M_D and keeping wanted evidence D_w.
- We formalize FORSID and show its **NP**-hardness (Section 6.2).
- We establish a connection between FORSID and RED-BLUE SET COVER, and exploit it to design the ForSId-via-RBSC algorithm for FORSID (Section 6.3).
- We provide extensive experiments on a variety of datasets and settings, and involving a number of baselines (Section 6.4). Results attest the high performance of the proposed ForSId-via-RBSC algorithm, and its superiority over the baselines (Section 6.5).

Section 6.1 overviews the related work. Section 6.6 concludes the paper.

6.1 Related Work

Machine unlearning (MU). Since its introduction [35], MU has been a very active area of research. A plethora of different MU approaches have been devised, either exact or approximate, and adhering to various principles, including data reorganization and model manipulation [265]. There also exist (recent) works that investigate how the properties of the forget set render the MU process more or less difficult [94, 279].

Our work is orthogonal to the bulk of the MU literature. MU approaches take a forget set as input (either explicitly or implicitly, e.g., by deriving it as the difference of the training set and an input retain/preservation set [24, 191]). Instead, we tackle the (so far unstudied) preliminary problem of identifying a forget set from the user's unlearning evidence. As such, the problem and algorithms defined in this work complement MU methodologies, although experimentally testing them on the reliable exact unlearning task is the most grounded way to assess their impact (cf. Section 6.4).

Variants of MU include the *selective forgetting* problem [229], whose goal is to forget one or more entire classes from a trained model, rather than individual samples. This is a goal that is also at the basis of other slightly different, but still class-level forgetting problems [153, 242, 270]. *Model editing* (or *knowledge editing*) has recently emerged in the large language model (LLM) landscape as the problem of modifying an LLM to incorporate specific knowledge, without negatively influencing other irrelevant knowledge [257].

Although similar in spirit, those variants remain anyway different from MU. The proposed FORSID problem is mainly aimed at being auxiliary in a MU context.

Training data influence. Several existing works tackle the problem of assessing the "influence" (or "impact") of individual training samples on the prediction of some given test samples. Approaches of this kind are based on the Fisher information matrix [103], influence functions [142], or similarity metrics, such as the Mahalanobis inner product [150, 31]. As for deep learning models, advanced techniques such as representer points [271] offer a way to measure the influence of multiple samples simultaneously.

In this work, we use training data influence methods as a building block of the algorithm we design for our FORSID problem (cf. Sect. 6.3).

RED-BLUE SET COVER (RBSC) is variant of the well-known SET COVER problem, where two finite sets of "red" (R) and "blue" (B) elements are given, along with a family $\mathcal{S} \subseteq 2^{R \cup B}$ of subsets, and the objective is to select a subfamily $\mathcal{C} \subseteq \mathcal{S}$ that covers all blue elements while minimizing the number of covered red elements [38]. The literature in RBSC has mostly focused on analyzing its theoretical properties, studying the (in)approximability of both the basic formulation [38, 49, 87] and geometric variants of it [1, 11, 41, 171].

In this work, we use RBSC as a tool to take inspiration from for algorithm design for the proposed FORSID problem. Advancing the state of the art in RBSC is beyond the scope of this work.

6.2 Problem Definition

We are given a machine-learning model M_D trained on a training set D. The machine-learning task underlying M_D is left unspecified in our problem: any task can in principle be handled, as long as the notions and functions that are left general in the definition of the problem are properly instantiated to account for the desired task (see next). We assume D to correspond to a set of pairs $\{(\vec{t}_i, y_i)\}_{|D|}$, where each $\vec{t}_i \in \mathbb{R}^d$ is a d-dimensional real-valued numerical vector representing the i-th training set sample, and y_i is the corresponding ground-truth label. The various y_i's – as well as the output $M_D(\vec{t})$ of the M_D model applied to any input $\vec{t}$ – take values from a domain $\mathcal{Y}$, which is instantiated based on the specific machine-learning task. For instance, in case of (univariate) regression $\mathcal{Y} = \mathbb{R}$, for single-label binary classification $\mathcal{Y} = \{0, 1\}$, for multilabel multiclass classification $\mathcal{Y} = \mathbb{N}^k$, $k \geq 1$. Let also:

- $D_f \subseteq D$ be a *forget set*, i.e., a subset of the training set that needs to be discarded in order to meet the unlearning desiderata specified by the user through an "unwanted set" (see next).
- $M_{D \setminus D_f}$ be the machine-learning model obtained by retraining the original input model on the training set $D \setminus D_f$, often referred to as *gold model* in the Machine Unlearning literature [52].
- D_u be an *unwanted set*, i.e., a set $\{\vec{u} \in \mathbb{R}^d\}$ of evidence (that satisfies the model input) that constitutes what the user requires to be unlearned from M_D. In general, $D_u \cap D$ may be nonempty. If this is the case, the samples in $D_u \cap D$ are directly included in D_f. We say that a retrained model $M_{D \setminus D_f}$ has actually unlearned a evidence $\vec{u}$ if the output of $M_{D \setminus D_f}$ on $\vec{u}$ differs from the output of M_D on $\vec{u}$ by at least a certain user-defined threshold $T > 0$, i.e., $|M_{D \setminus D_f}(\vec{u}) - M_D(\vec{u})| \geq T$. The threshold T is set according to the machine-learning task and the application domain. E.g., in single-label classification (either binary or multiclass), $T = 1$, as any $\vec{u}$ is reasonably considered unlearned if its new predicted class is other than the originally predicted one. In regression tasks, T is set so that changes in the numerical predictions that exceed T are large enough to recognize the predictions as actually changed. Note that the threshold T plays the role of quantifying to what extent the various unwanted samples have been actually forgotten. The forgetting on the

unwanted samples is counterbalanced by the desideratum of keeping the original model predictions unchanged on a set of wanted instances according to a properly-defined prediction similarity function (see next).

- D_w be a *wanted set*, i.e., a set $\{\vec{w} \in \mathbb{R}^d\}$ of evidence (that satisfies the model input) on which the prediction of the retrained model (resulting from the unlearning process) is required to be as much as possible equal to the prediction of the original model. We assume $D_w \cap D_u = \emptyset$, as an evidence can clearly exhibit wanted or unwanted behavior only, not both.

- σ be a *prediction similarity function*, i.e., a real-valued function which takes the original model M_D, the model $M_{D \setminus D_f}$ retrained on $D \setminus D_f$, and the wanted set D_w as arguments, and returns a real number expressing the similarity between the outputs of M_D and $M_{D \setminus D_f}$ on the various samples $\vec{w} \in D_w$. Like T, σ is defined based on the machine-learning task and the application. As an example, in single-label binary classification, σ might be defined as the number of samples of D_w for which M_D and $M_{D \setminus D_f}$ give the same classification, i.e., $\sigma := \sum_{\vec{w} \in D_w} 1 - |M_D(\vec{w}) - M_{D \setminus D_f}(\vec{w})|$. In single-label multiclass classification, σ might correspond to the average cross-entropy across all the samples of D_w using the output of M_D as a ground-truth.

Informally speaking, in this work we deal with the problem of identifying a forget set whose removal from the original training set leads to both unlearning all the evidence in a given unwanted set and maximum similarity in the model's outputs on the wanted set before and after the unlearning process. We formalize such desiderata in the following novel combinatorial-optimization problem:

Problem 1 (FORGET SET IDENTIFICATION (FORSID))**.** *Given a training set D, a machine-learning model M_D trained on D, an unwanted set D_u, a wanted set D_w, a prediction similarity function σ, and a threshold $T > 0$, find a forget set $D_f^* \subseteq D$ that maximizes σ subject to the constraint that all the samples of D_u get unlearned:*

$$D_f^* = \underset{D_f \subseteq D}{\operatorname{argmax}}\ \sigma(M_D, M_{D \setminus D_f}, D_w) \tag{6.1}$$

$$\textit{subject to} \quad |M_{D \setminus D_f}(\vec{u}) - M_D(\vec{u})| \geq T, \forall \vec{u} \in D_u. \tag{6.2}$$

The rationale of Problem 1 is as follows. The target scenario of properly identifying a forget set comes with two main desiderata. On the one hand, as Desideratum 1, the unwanted samples D_u need to be "forgotten" as much as possible (where "forgotten" means that the prediction on each sample in D_u after the unlearning process should considerably change with respect to the original prediction). On the other hand, as Desideratum 2, we also need to ensure that the original predictions on the wanted samples D_w are, to the largest possible extent, similar to the original predictions achieved before the unlearning. These two desiderata are clearly conflicting to each other. In order to formalize such desiderata as objectives of an optimization problem, we adopt the usual way of letting one desideratum to correspond to the objective function of the resulting optimization problem, and putting the other objective as a constraint. In our context, and in machine unlearning in general, the emphasis is on forgetting the unwanted samples. Therefore, in the formal definition of our FORSID problem, we decided to model Desideratum 1 as a constraint and Desideratum 2 as the objective function. Based on this, it is apparent that a scenario where forcing to change the prediction of a sample in D_u leads to a change in the

TABLE 6.1: Main notations used in this work.

symbol	*description*
$D = \{(\vec{t}_i, y)\}_{\lvert D \rvert}$	training set
M_D	machine-learning model trained on D
$D_f \subseteq D$	forget set
$M_{D \setminus D_f}$	machine-learning model retrained on $D \setminus D_f$
$D_u = \{\vec{u}\}_{\lvert D_u \rvert}$	unwanted set
T	positive threshold to recognize any $\vec{u} \in D_u$ as unlearned
$D_w = \{\vec{w}\}_{\lvert D_w \rvert}$	wanted set
$\sigma(M_D, M_{D \setminus D_f}, D_w)$	prediction similarity function
$\alpha(M_D, \vec{t}, \vec{z}) \in \mathbb{R}$	impact of $\vec{t} \in D$ on the prediction $M_D(\vec{z})$ of $\vec{z} \in D_u \cup D_w$ by model M_D

prediction of other samples not in D_u is hindered by the objective function of the problem.

The FORSID problem fills an important gap in the MU landscape, addressing the so-far overlooked challenge of identifying a forget set out of user's unlearning desiderata. Once a forget set is obtained, it can be used to perform the actual unlearning by resorting to existing MU methods. As such, FORSID constitutes an essential intermediate step that allows for extending the range of applicability of MU to all those settings where user's unlearning evidence does not correspond to (or is too hard to be directly expressed in terms of) a forget set. This arises in many real-world scenarios, such as users' rights compliance, personalized recommender systems, and regulation update (cf. Introduction). The main notations used in this thesis are summarized in Table 6.1.

Theorem 1. Problem 1 is **NP**-hard.

Proof. We reduce from the well-established **NP**-hard KNAPSACK problem [137]: given a constant $C > 0$ and a set X of items, where each $x_i \in X$ is assigned value $v_i > 0$ and weight $w_i > 0$, find a subset $X^* \subseteq X$ of items that maximizes the total value $\sum_{x_j \in X^*} v_j$ subject to the constraint $\sum_{x_j \in X} w_j \leq C$. Given an instance $I = \langle C, X, \{v_i\}, \{w_i\} \rangle$ of KNAPSACK, we construct an instance $I' = \langle D, M_D, D_u, D_w, \sigma, T \rangle$ of FORSID as follows:

- $D = X$;
- M_D is set to a constant function that always outputs H, regardless of the input instance x_i, where H is an arbitrary positive constant, i.e., $M_D(x_i) = H, \forall x_i \in D$;
- D_u is composed of a single sample, while $D_w = \emptyset$;
- $\sigma(M_D, M_{D \setminus D_f}, D_w) = \sum_{x_i \in D_f} v_i$;
- $T = W$, where $W = \sum_{x_i \in X} w_i$ is the sum of the weights of all the items.

Moreover, for any $D_f \subseteq D$, we set $M_{D \setminus D_f}$ to a constant function that always outputs $H + W + (C - \sum_{x_i \in D_f} w_i)$, regardless of the input instance x_i, i.e., $M_{D \setminus D_f}(x_i) = H + W + (C - \sum_{x_i \in D_f} w_i), \forall x_i \in D$. The construction of such an instance I' can clearly be accomplished in polynomial time.

The objective function of FORSID on instance I' is $\operatorname{argmax}_{D_f \subseteq D} \sigma(M_D, M_{D \setminus D_f}, D_w)$ $= \operatorname{argmax}_{X' \subseteq X} \sum_{x_i \in X'} v_i$, which corresponds to KNAPSACK's objective function on

instance I. As far as FORSID's constraint (Eq. (6.2)), instead, it holds that:

$$\begin{aligned} & |M_{D\setminus D_f}(\vec{u}) - M_D(\vec{u})| \geq T, \forall \vec{u} \in D_u \\ \Leftrightarrow \quad & |M_{D\setminus D_f}(\vec{u}) - M_D(\vec{u})| \geq T, \text{on the only } \vec{u} \in D_u \\ \Leftrightarrow \quad & |H + W + (C - \textstyle\sum_{x_i \in D_f} w_i) - H| \geq W \\ \Leftrightarrow \quad & W + C - \textstyle\sum_{x_i \in D_f} w_i \geq W \;\Leftrightarrow\; \textstyle\sum_{x_i \in D_f} w_i \leq C, \end{aligned}$$

where the absolute-value removal in the second last implication holds because $W + C - \sum_{x_i \in D_f} w_i \geq 0$.

All in all, solving FORSID on instance I' corresponds to finding a subset $D_f^* = X^* \subseteq X = D$ that maximizes $\sum_{x_i \in D_f^* = X^*} v_i$ subject to the constraint $\sum_{x_i \in D_f^* = X^*} w_i \leq C$, which corresponds to solving KNAPSACK on instance I. The theorem follows. □

The above theorem states that the proposed FORSID problem is at least as difficult as KNAPSACK. However, the FORSID instance constructed in the reduction is a simplistic one (it does not use D_u or D_w, and employs constant functions as models M_D and $M_{D\setminus D_f}$). Intuitively, this makes us think that FORSID is more challenging than KNAPSACK to be tackled in practice. This in turn means that the connection to KNAPSACK is not really helpful in terms of algorithm design. In fact, the algorithm we devise for FORSID is based on different technical insights (cf. next section).

6.3 Algorithms

Connection to RBSC. Our FORSID problem can be interpreted in terms of RED-BLUE SET COVER (RBSC) (cf. Section 6.1). The set B of blue elements (the ones to be necessarily "covered") corresponds to the unwanted set D_u. The set R of red elements (the ones to be "covered" the least possible) corresponds to the wanted set D_w. The input family $\mathcal{S}$ of subsets – which the ultimate desired forget set D_f is selected from – corresponds to the set 2^D of all possible subsets of the training set D. That is the easy part. What is challenging is to come up with a proper definition of what "coverage" means here. We discuss this next.

"Coverage" of D_u and D_w. In principle, recognizing a certain set $D_f \in \mathcal{S} = 2^D$ as covering elements of D_u or D_w would require retraining the original model (over a training set $D \setminus D_f$). In fact, according to FORSID's problem statement (Problem 1), $M_{D\setminus D_f}$ needs to be evaluated on both D_w (Eq. (6.1)) and D_u (Eq. (6.2)). Even if algorithms are designed to limit the number of subsets of D that need to be generated and explored during execution, retraining a machine learning model for each of these subsets remains complex and inefficient. Resorting to MU methods to unlearn any of such subsets of D (sacrificing optimality) would make things better, but not that much. For this reason, here we define our desired notion of coverage in terms of the impact that training samples exhibit over the predictions of a trained machine-learning model, without requiring any model retraining. Specifically, for each training sample $\vec{t} \in D$, and each sample $\vec{z} \in D_u \cup D_w$ of the (un)wanted set, let $\alpha(M_D, \vec{t}, \vec{z}) \in \mathbb{R}$ denote the impact of $\vec{t}$ on the prediction $M_D(\vec{z})$ of $\vec{z}$ by model M_D: the higher $\alpha(M_D, \vec{t}, \vec{z})$, the more responsible $\vec{t}$ is for the prediction assigned to $\vec{z}$ by M_D, and vice versa. In other words, the higher $\alpha(M_D, \vec{t}, \vec{z})$, the more likely is that retraining the model on $D \setminus \{\vec{t}\}$ will lead to a change in the prediction of $\vec{z}$, and vice versa.

Algorithm 1 ForSId-via-RBSC

Input: Training set D; machine-learning model M_D trained on D; unwanted set D_u; wanted set D_w; threshold $T > 0$
Output: Forget set $D_f \subseteq D$

1: Compute impact scores $\alpha(M_D, \vec{t}, \vec{z})$, $\forall \vec{t} \in D$, $\forall \vec{z} \in D_u \cup D_w$
2: Build linear program P of Problem 2 based on $\alpha(M_D, \vec{t}, \vec{z})$ and T
3: $\{b_{\vec{t}} \mid \vec{t} \in D\} \leftarrow$ ILPSOLVER(P)
4: $D_f \leftarrow \{\vec{t} \in D \mid b_{\vec{t}} = 1\}$

The α scores can be computed in various ways, such as through similarity between samples, or data-influence measures [166, 103, 142, 150, 31, 271] (cf. Section 6.1).

The RED-BLUE FORSID problem. We translate the above principles into an optimization problem which is a variant of RBSC and is dubbed RED-BLUE FORSID. For reasons that will become clear in a while, we provide an integer linear programming (ILP) formulation:

Problem 2 (RED-BLUE FORSID)**.** *Given a training set D, let each sample $\vec{t} \in D$ be assigned a decision binary variable $b_{\vec{t}} \in \{0, 1\}$ expressing whether $\vec{t}$ is part ($b_{\vec{t}} = 1$) or not ($b_{\vec{t}} = 0$) of the output forget set. Given a training set D, a machine-learning model M_D trained on D, an unwanted set D_u, a wanted set D_w, a threshold $T > 0$, and impact scores $\alpha(M_D, \vec{t}, \vec{z})$, $\forall \vec{t} \in D$, $\forall \vec{z} \in D_u \cup D_w$,*

$$\textit{minimize} \quad \sum_{\vec{t} \in D, \vec{w} \in D_w} b_{\vec{t}}\, \alpha(M_D, \vec{t}, \vec{w}) \tag{6.3}$$

$$\textit{subject to} \quad \sum_{\vec{t} \in D} b_{\vec{t}}\, \alpha(M_D, \vec{t}, \vec{u}) \geq T,\ \forall \vec{u} \in D_u, \tag{6.4}$$

$$b_{\vec{t}} \in \{0, 1\},\ \forall \vec{t} \in D. \tag{6.5}$$

Problem 2 selects a subset $D_f = \{\vec{t} \in D \mid b_{\vec{t}} = 1\}$ (forget set) of the input training set D so that the cumulative impact of the selected training samples on all the wanted samples is minimized (Eq. (6.3)), subject to the constraint that the cumulative impact of the selected training samples on each unwanted sample is large enough, i.e., no less than threshold T (Eq. (6.4)). Conceptually speaking, threshold T in Problem 2 plays a similar role to the threshold used in Problem 1. However, instead of affecting actual changes in predictions like in Problem 1, threshold T in Problem 2 affects impact estimates α. Here, we slightly abuse of notation, as we denote with the same symbol T for the threshold in both Problem 1 and Problem 2. Unless otherwise specified, in the remainder of the paper, when referring to T we mean the threshold in Problem 2. Problem 2 can be easily recognized as a variant of the basic RBSC problem, where the blue and red sets correspond to the unwanted and wanted sets, respectively. The main differences with respect to the basic RBSC are: (*i*) the input family is implicitly defined as all the possible subsets of the training set, and (*ii*) a notion of weighted coverage is used (expressed as the cumulative impact of the various selected training samples).

The proposed ForSId-via-RBSC algorithm. The forget set ultimately selected by solving the RED-BLUE FORSID problem (Problem 2) will contain training samples whose removal from D leads to the least possible change in the prediction of the wanted samples (as quantified by the impact scores α) and to a highly likely change of prediction of each unwanted sample (as quantified by the impact scores α exceeding T). This way, RED-BLUE FORSID's objective function (Eq. (6.3)) and RED-BLUE FORSID's main constraint (Eq. (6.4)) represent reasonable proxies for FORSID's

TABLE 6.2: Characteristics of the datasets used in the experiments, along with the accuracy achieved by a machine-learning model whose network model is reported under brackets (CL: convolutional layers, FC: fully-connected layers).

dataset	*features*	*type*	*#samples*	*#classes*	*Accuracy [network model]*
CIFAR-10	32x32x3	image	60 000	10	$\sim$ 95% [3 CL (64,128,256) + 3 FC]
CIFAR-20	32x32x3	image	60 000	20	$\sim$ 85% [3 CL (64,128,256) + 3 FC]
CIFAR-100	32x32x3	image	60 000	100	$\sim$ 65% [VGG-16]
Wine Quality	11	tabular	4 898	10	$\sim$ 65% [64,32,32 FC]

objective function (Eq. (6.1)) and FORSID's main constraint (Eq. (6.2)), respectively. Note that the use of such proxies eliminates the dependence on a prediction similarity function σ. This side effect is desirable, as it avoids defining an ad-hoc σ for any machine-learning task and application context, which may be otherwise challenging.

Motivated by this, the algorithm we propose for our FORSID problem consists in solving the proxy RED-BLUE FORSID problem. At first glance, as RED-BLUE FORSID is a variant of RBSC, an option to solve it would be to resorting to existing algorithms for RBSC (cf. Sect. 6.1). However, all these algorithms are mostly theoretical (and, often, not really easy-to-implement either). They are mainly aimed at proving (in)approximability properties, rather than being profitably used in practice (in fact, no experimental evaluation has ever been carried out on such algorithms). Moreover, our notion of weighted coverage makes the adaptation of those algorithms to our RED-BLUE FORSID variant challenging (and, perhaps, any adaptation would make them lose their approximation guarantees). For all these reasons, here we propose to solve the RED-BLUE FORSID problem with an ILP solver. We defer to future work the design of different and more specialized algorithms. The resulting proposed algorithm is termed ForSId-via-RBSC and outlined in Algorithm 1.

Complexity. The running time of ForSId-via-RBSC is dominated by the time taken by the ILP solver. The other steps cost as follows. Computing the impact scores (Line 1) depends on the specific technique employed. However, it takes at least time $\Omega(|D| \times (|D_u| + |D_w|))$, as it yields a score $\forall \vec{t} \in D$, $\forall \vec{z} \in D_u \cup D_w$. Building the linear program (Line 2) takes $\mathcal{O}(|D| \times (|D_u| + |D_w|))$ time, while constructing the output D_f (Line 4) takes $\mathcal{O}(|D|)$ time.

The space complexity of ForSId-via-RBSC is $\mathcal{O}(|D| \times (|D_u| + |D_w|))$, which corresponds to both the number of α scores and the size of the linear program.

6.4 Experimental Setting

Datasets (Table 6.2). We selected datasets that are used in the bulk of the literature in MU [53, 110, 103]. These include the well-established CIFAR family of datasets, namely CIFAR-10, CIFAR-20, and CIFAR-100 [148]. The CIFAR family is composed of 60k images of 32x32 color pixels. CIFAR-10 consists of 10 distinct classes, whereas CIFAR-100 includes both coarse-grained labels (forming CIFAR-20) and fine-grained labels (defining CIFAR-100). We chose CIFAR-10 because it has clear decision boundaries within its classes. Instead, CIFAR-20 and CIFAR-100 present high uncertainty and overlapping intra-class decision boundaries (testified by the decreasing accuracy values reported in Table 6.2). As such, CIFAR-20 and CIFAR-100 constitute a highly-challenging testbed. For each experiment with those datasets, we selected a combination of 10 distinct coarse-grained labels for CIFAR-10

and CIFAR-20, whereas, for CIFAR-100, we similarly selected from the fine-grained labels.

Additionally, to the CIFAR family, we incorporated the Wine Quality dataset [56] to assess our method on a smaller-scale (11 features and around 5k samples), multi-class tabular dataset, where labels represent quality scores ranging from 0 to 10. We use the Wine Quality dataset in its standard classification form, as provided by the UCI Repository [56]. As for CIFAR-10 and 20 we sampled 1 000 samples for the unwanted set D_u and 1000 samples for the wanted set D_w. In the case of CIFAR-100 and Wine, as 1 000 instances per class are not available, we selected 100 for both the unwanted and wanted sets.

Baselines. The FORSID problem we tackle in this work is a newly formulated one. As such, no established methods or benchmarks exist for it in the literature. For this reason, here we design well-founded yet nontrivial baseline approaches to be involved for comparison to the proposed ForSId-via-RBSC algorithm. One of such baselines operates in the data space, while the remaining ones exploit the same training data influence function as our ForSId-via-RBSC. We describe these baselines next.

K-nearest neighbors (k-NN): For each sample $\vec{u} \in D_u$ of the unwanted set D_u, this baseline computes $\vec{u}$'s k-nearest neighbors in the training set D using some distance measure $dist(\vec{u}, \vec{t})$ in the data space. The forget set D_f is ultimately computed as the union of all such k-nearest neighbors. We chose $dist(\vec{u}_i, \vec{t}_j) = \|\vec{u}_i - \vec{t}_j\|_2$ as the distance function due to its computational efficiency and widespread use in k-NN searches and image datasets. This baseline aims to identify the closest instances within the data space to assess whether the problem can be effectively addressed through a very simple approach that completely discard the input model M_D.

Top-K Union (TopK-U) computes the forget set D_f as the union of the k most influential samples $\vec{t}$ of the training set D for each sample $\vec{u}$ in the unwanted set D_u, according to the chosen impact score function α, i.e., $D_f = \bigcup_{\vec{u} \in D_u} \text{TopK}(M_D, D, \vec{u}, k)$, where $\text{TopK}(M_D, D, \vec{u}, k) = \sum_{\vec{t} \in S_k} \alpha(M_D, \vec{t}, \vec{u})$, and $S_k \subseteq D$ is the set of the k training samples exhibiting the highest α score.

Note that TopK-U solely leverages the impact of the training samples on the unwanted set. To ensure a fair comparison, this baseline (and the next ones) adopt the same impact score function α, which constitutes their key strength.

Top-K Difference (TopK-D) computes the forget set D_f as the set difference between the union of the k most influential training samples $\vec{t} \in D$ for each sample $\vec{u} \in D_u$ and the union of the k most influential training samples $\vec{t} \in D$ for each sample $\vec{w} \in D_w$, based on the chosen impact score function α, i.e.,

$$D_f = \bigcup_{\vec{u} \in D_u} \text{TopK}(M_D, D, \vec{u}, k) \Big\backslash \bigcup_{\vec{w} \in D_w} \text{TopK}(M_D, D, \vec{w}, k).$$

Impact Disparity (ImpDisp) computes the forget set D_f as the set of training samples $\vec{t} \in D$ for which the disparity between their cumulative impact on $\vec{u} \in D_u$ and $\vec{w} \in D_w$ exceeds a given threshold l, i.e., $D_f = \left\{ \vec{t} \in D \,\middle|\, \left(\sum_{\vec{u} \in D_u} \alpha(M_D, \vec{t}, \vec{u}) - \sum_{\vec{w} \in D_w} \alpha(M_D, \vec{t}, \vec{w}) \right) > l \right\}$.

If and only if the threshold l is set to 0, as in our case, the impact disparity admits the following equivalent optimization formulation:

$$\text{maximize}_{\vec{t} \in D} \quad \sum_{\vec{u} \in D_u} \alpha(M_D, \vec{t}, \vec{u}) - \sum_{\vec{w} \in D_w} \alpha(M_D, \vec{t}, \vec{w}) \tag{6.6}$$

Impact score function α. As stated in Section 6.3, α scores can be computed in various ways, such as via similarity among samples, data-influence measures [166, 103, 142, 150, 31, 271], or the Fisher information matrix [103, 156, 164]. In this work, we adopt the Representer Point framework [271] to instantiate the impact scores $\alpha(M_D, \vec{t}, \vec{z})$. This is mainly motivated by the fact that representer points are particularly well-suited for our problem since the total impact of all training samples on a prediction is decomposed into a sum of individual contributions – a key property that aligns well with the RED-BLUE FORSID formulation.

Assessment criteria. To evaluate the effectiveness of ForSId-via-RBSC and the baseline methods, it is essential to assess their ability to successfully unlearn the unwanted set D_u while preserving the predictions on the wanted one D_w. To achieve this, we define a set of benchmarking measures based on $M_{D\setminus D_f}$ – the machine learning model obtained by retraining the original input model on $D \setminus D_f$. Note that we evaluate our method and the baselines w.r.t. the exact unlearning task (i.e., from-scratch retraining) to avoid any confounding factors of inexact unlearning methods. Specifically, we employ the following measures.

Efficacy, defined on top of a revised confusion matrix where the true positives TP (wanted unchanged) and false negatives FN (wanted changed) correspond to the number of predictions of the samples in the wanted set D_w that have unchanged / changed in the unlearned model $M_{D\setminus D_f}$ with respect to the original model M, respectively. Conversely, the false positives FP (unwanted unchanged) and true negatives TN (unwanted changed) correspond to the number of predictions of the samples in the unwanted set D_u that have unchanged / changed in the unlearned model $M_{D\setminus D_f}$ with respect to the original model M, respectively.

Wanted Efficacy (WE), defined as the ratio of wanted samples that maintain their original predictions relative to the total number of wanted sets, i.e., $\frac{TP}{TP+FN}$.

Unwanted Efficacy (UE), defined in a similar way to WE, it is the ratio of unwanted samples whose predictions changed to the total number of unwanted samples, i.e., $\frac{TN}{FP+TN}$.

Balanced Efficacy (BE), defined as the mean between UE and WE, i.e., $2 \cdot \frac{FP \cdot FN}{TP \cdot TN}$.

Accuracy Loss (Acc. Δ). To assess the impact of the forget set computed by each method in a broader context, we also consider the accuracy divergence between M and $M_{D\setminus D_f}$ on a test set D_t disjoint from all the other sets at hand (i.e., D, D_u, D_w). The rationale is that an effective forget set should induce a maximal change in the unwanted samples' predictions while minimizing the retrained model's overall accuracy loss.

Overall Effectiveness (H). To provide the reader with a comprehensive picture of the wanted and unwanted efficacy with the accuracy loss in an all-in-one score, we compute the harmonic mean of the WE, UE, and Acc. Δ measures, i.e., $\frac{3}{\frac{1}{UE}+\frac{1}{WE}+\frac{1}{\Delta Acc}}$.

Forget set size (FS-Size). We report the size of the identified forget set for each method as the average number of samples removed, providing a quantitative measure of the extent of forgetting.

Running Time (RunT) of any method involved in the comparison, measured in seconds.

Parameters and other settings. As for the baselines using parameter k (namely k-NN, TopK-U, and TopK-D), we selected k ranging from 5 to 30 000. Specifically, we used a progressively increasing sequence, starting with smaller increments (e.g., $k = 5, 10, 20, 30, 50$) and transitioning to larger gaps at higher values (e.g., $k =$

100, 150, 200, 1 000), ensuring broader coverage across different scales. As we cannot guarantee monotonic behavior for our method or the baselines with respect to their parameter, we adopt a linear tuning approach to ensure a more robust and comprehensive analysis. We also systematically investigate the behavior of the methods as the parameter k approaches the size of the full dataset, in order to stress-test the methods and reveal asymptotic trends (this study is shown in-depth in Section 6.7.2). As for our ForSId-via-RBSC, we tuned parameter T ranging from $1e{-}5$ to 50, using a finer granularity in the empirically best-performing range (10 to 20). As a neural networks model, we used VGG-like architecture for Image-based classification, while for tabular classification, we used a feed-forward fully connected neural network with three hidden layers (64, 64, 32). In both cases, we trained the networks with the Adam optimizer for 30 epochs with a learning rate of 0.01. All networks were trained to achieve $>$ 95% training accuracy. The last layer was fine-tuned to ensure that the predictions closely aligned with the representer theorem formulation ($>$ 99% Pearson correlation between ground truth prediction and prediction by representer theorem, as described in [271]). All experiments were run 10 times, with each run sampling the Unwanted Set D_u from a different class (classes 0 to 10, and random classes for CIFAR-100). The test set D_t was kept the same across all runs, comprising 20% of the original dataset.

Testing environment. For timed experiments, we ran all methods on a Rocky Linux 8 server with 32-core Intel(R) Xeon(R) CPU @ 2.30GHz, up to 64GB RAM, NVIDIA A100 (MIG 56 sm 40GB) GPU.

6.5 Experimental Results

In the following, we provide and discuss the experimental results of the proposed ForSId-via-RBSC algorithm (Algorithm 1), and compare them to the baselines described in Section 6.4. In Appendix 6.7.1, we report the tables with the standard deviations, here omitted for the sake of space. In Appendix 6.7.2, we report parameter studies on parameters T and K, and study the behavior of our method and baselines when they grow to the limit of the dataset size.

We conducted extensive experiments designed to capture the real-world complexities of FORGET SET IDENTIFICATION, where the distinction between the wanted and unwanted evidence may not always be clear-cut. Specifically, we structure our evaluation around three different experimental settings, each characterized by varying degrees of uncertainty in the distinction between the unwanted and wanted sets. Moreover, the chosen real-world datasets (cf. Sect. 6.4) allow us to highlight the capabilities of our method in terms of generalization across different data distributions and characteristics, while also exposing potential limitations when faced with complex, overlapping decision boundaries between the wanted and unwanted sets. This experimental design aims to provide a rigorous and transparent evaluation of the ForSId-via-RBSC algorithm, demonstrating its empirical advantages over the baselines.

ONE VS OTHERS. This experiment is intended to have a good separation between the unwanted and wanted sets while keeping low variance in selecting unwanted evidence. We sample the unwanted set D_u from one class c and the wanted set D_w from all the remaining classes. This way, no sample from class c is part of the wanted set. The results of this experiment are reported in Table 6.3.

ONE VS ALL. This experiment has a similar setting to the ONE VS OTHERS one, but here the wanted set D_w is sampled following the distribution of all classes, including

TABLE 6.3: ONE VS OTHERS | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.605	0.554	0.657	0.191	0.639	36814.818	58.122
	TopK-U	0.639	0.345	0.932	0.018	0.585	10205.091	50.029
	TopK-D	0.525	0.055	**0.996**	**0.001**	0.148	613.182	**25.622**
	ImpDisp	0.655	0.522	0.788	0.122	0.676	24278.250	2658.045
	Ours	**0.982**	**0.999**	0.965	0.077	**0.952**	19266.000	1094.689
CIFAR-20	k-NN	0.767	0.608	0.927	0.011	0.795	7676.500	149.123
	TopK-U	0.843	0.798	0.889	0.014	0.878	8988.100	40.479
	TopK-D	0.661	0.322	**1.000**	**0.006**	0.584	1065.778	**31.510**
	ImpDisp	0.790	0.948	0.631	0.102	0.766	25296.900	3223.776
	Ours	**0.986**	**0.993**	0.979	0.011	**0.983**	5247.000	1226.710
CIFAR-100	k-NN	0.774	0.626	0.921	0.034	0.782	8308.800	20.272
	TopK-U	0.902	0.867	0.936	0.014	**0.913**	5098.500	**1.918**
	TopK-D	0.835	0.669	**1.000**	**0.007**	0.846	672.400	3.714
	ImpDisp	0.799	**0.968**	0.630	0.173	0.742	24618.100	322.440
	Ours	**0.968**	0.943	0.993	0.152	0.873	17023.667	187.575
Wine	k-NN	0.855	0.880	0.830	0.038	0.871	2508.200	**0.179**
	TopK-U	0.768	0.800	0.735	0.039	0.764	2186.250	1.326
	TopK-D	0.744	0.708	0.780	**-0.015**	0.735	115.500	0.239
	ImpDisp	0.775	0.761	0.790	0.021	0.763	2310.600	19.750
	Ours	**0.920**	**1.000**	**0.840**	0.028	**0.924**	1795.000	9.848

c. This will cause a higher uncertainty since the samples of class c contribute to both wanted and unwanted sets. The results of this experiment are reported in Table 6.4.

MANY VS MANY. In this last experiment, we pushed unwanted set's variance further. We sample the unwanted set D_u and the wanted set D_w from two disjoint sets of multiple classes. The results of this experiment are reported in Table 6.5.

Result analysis. Across all experimental settings, ForSId-via-RBSC (referred to as "Ours" in Tables 6.3–6.5) consistently outperforms the baseline approaches in terms of Balanced Efficacy (BE), Wanted Efficacy (WE), and Unwanted Efficacy (UE), demonstrating a strong ability to unlearn unwanted instances, while preserving the performance on the wanted set. The method achieves this without the extreme variability seen in some baselines. The forget set produced by ForSId-via-RBSC is of reasonable size (FS-Size). This complies with the intuition that an excessively small forget set may fail to effectively address the FORGET SET IDENTIFICATIONproblem, while an overly large one can degrade model's performance.

Among the baselines, k-NN provides reasonable results but lacks precision in distinguishing unwanted instances (i.e., UE), leading to moderate efficacy scores across all datasets. TopK-U is relatively competitive, particularly when some overlap exists between wanted and unwanted sets, but it does not consistently match the balanced efficacy of ForSId-via-RBSC. TopK-D struggles with unwanted efficacy, excelling in retaining wanted instances but failing to fully remove unwanted ones, leading to an

TABLE 6.4: ONE VS ALL | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.593	0.560	0.627	0.190	0.632	36660.100	54.281
	TopK-U	0.626	0.349	0.903	0.018	0.581	10137.400	50.680
	TopK-D	0.503	0.007	**0.999**	**-0.001**	0.021	38.000	**6.135**
	ImpDisp	0.661	0.562	0.760	0.122	0.662	24621.300	2663.667
	Ours	**0.802**	**0.695**	0.910	0.033	**0.837**	10297.000	1281.942
CIFAR-20	k-NN	0.760	0.626	0.895	0.007	0.799	9828.250	48.670
	TopK-U	0.823	0.820	0.826	0.010	0.867	11907.875	36.859
	TopK-D	0.516	0.032	**0.999**	**-0.004**	0.090	93.875	**5.903**
	ImpDisp	0.793	**0.948**	0.638	0.074	0.788	23874.500	2664.712
	Ours	**0.850**	0.864	0.835	0.022	**0.880**	6858.000	1237.051
CIFAR-100	k-NN	0.800	0.669	0.931	0.026	0.832	6963.625	22.702
	TopK-U	0.926	0.963	0.889	0.020	**0.936**	5156.375	**1.890**
	TopK-D	0.676	0.355	0.996	**0.001**	0.441	524.250	4.170
	ImpDisp	0.829	**1.000**	0.657	0.177	0.757	24636.125	330.794
	Ours	**0.988**	**1.000**	**0.977**	0.187	0.858	20417.667	112.443
Wine	k-NN	**0.797**	**0.820**	0.775	0.033	**0.832**	2368.500	**0.216**
	TopK-U	0.661	0.728	0.595	0.024	0.719	3875.000	3.324
	TopK-D	0.587	0.372	**0.802**	0.039	0.488	88.750	0.374
	ImpDisp	0.661	0.615	0.708	0.001	0.689	2457.750	25.294
	Ours	0.744	0.765	0.723	**-0.029**	0.778	2319.000	11.209

imbalance that undermines its effectiveness overall. ImpDisp performs well in certain cases but exhibits significant variation, particularly in settings where the wanted and unwanted sets overlap substantially, making it less reliable across different scenarios.

In the ONE VS OTHERS setting (Table 6.3), where the distinction between wanted and unwanted sets is clear, and the unwanted set is well defined (i.e., sampled from only one class), our ForSId-via-RBSC achieves near-optimal values for BE, UE, and WE, proving that it effectively unlearns the targeted samples while maintaining correct predictions on the wanted set. TopK-U and k-NN perform adequately but struggle to maintain high BE. TopK-D shows evident limitations, as its methodology of strictly differentiating wanted from unwanted instances does not generalize well in this setting. ImpDisp delivers inconsistent results across different datasets, reinforcing that while impact-based approaches can be useful, they are not universally effective.

When moving to the ONE VS ALL setting (Table 6.4), which introduces more complexity by allowing samples of the same class to belong to both wanted and unwanted sets, our ForSId-via-RBSC maintains superior efficacy scores but, as expected, with a slightly reduced margin compared to the (previous) clear separation setting.

TABLE 6.5: MANY VS MANY | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.552	0.476	0.628	0.216	0.586	39190.600	121.306
	TopK-U	0.579	0.327	0.831	0.072	0.559	27916.400	**75.401**
	TopK-D	0.543	0.093	**0.994**	**0.017**	0.233	6436.600	103.775
	ImpDisp	0.540	0.265	0.814	0.103	0.481	23792.000	3276.857
	Ours	**0.869**	**0.793**	**0.944**	0.262	**0.776**	30727.800	1065.329
CIFAR-20	k-NN	0.752	0.534	0.970	0.001	0.768	3838.429	46.738
	TopK-U	0.733	0.660	0.806	0.061	0.769	27496.143	**25.109**
	TopK-D	0.666	0.333	**1.000**	**0.009**	0.597	10531.600	25.686
	ImpDisp	0.539	0.370	0.709	0.065	0.567	23653.429	2717.737
	Ours	**0.945**	**0.895**	0.996	0.138	**0.849**	23344.200	1314.937
CIFAR-100	k-NN	0.722	0.485	0.958	**0.012**	0.727	6152.000	47.285
	TopK-U	0.800	0.787	0.813	0.189	0.755	28024.000	**13.336**
	TopK-D	0.709	0.421	**0.997**	0.104	0.653	15558.000	31.007
	ImpDisp	0.467	0.380	0.553	0.188	0.508	25987.000	2724.077
	Ours	**0.998**	**1.000**	**0.997**	0.256	**0.795**	24534.000	1756.515
Wine	k-NN	0.857	**1.000**	0.714	0.032	0.866	5196.000	**1.728**
	TopK-U	**0.921**	**1.000**	0.842	0.024	**0.927**	5194.000	59.013
	TopK-D	0.789	0.697	**0.882**	0.073	0.804	1853.000	23.323
	ImpDisp	0.752	0.658	0.845	0.014	0.804	2729.000	262.115
	Ours	0.801	0.752	0.851	**-0.003**	0.857	2638.000	124.070

This highlights its robustness in handling uncertainty. TopK-U improves in this setting, benefiting from its ability to account for influence without rigid constraints. k-NN remains relatively stable but does not improve significantly. TopK-D continues to struggle due to its inability to handle cases where influence is not strictly binary. While still variable, ImpDisp shows some improvement, particularly in datasets where unwanted and wanted samples exhibit structured similarities (i.e., CIFAR's ones).

In the MANY VS MANY setting (Table 6.5), where the wanted and unwanted sets are drawn from different sets of classes with increased variance, our ForSId-via-RBSC again leads in BE, UE, and WE. However, the accuracy loss (Acc. Δ) is higher than in previous settings, suggesting that the task difficulty introduces some trade-offs. However, it is important to note that accuracy alone does not give the full picture of the experiments. The sharp drop in accuracy in Table 6.5, which represents `many vs. many` experiments, is inherent within the problem's formulation. Since we are formulating multiple classes to be "Unwanted", a drop in accuracy is not only expected, but also somewhat desirable as the output of the classes represented in the unwanted set will be changed. Critically, our algorithm maintains near-perfect Wanted Efficacy, preserving the outputs of the sampled indicated as "Wanted" more than 94% of the times on CIFAR-10, and close to 99% on CIFAR-10 and CIFAR-100, as reported in Table 6.5. We also remind that `many vs many` is an edge case with maximum confusion

among samples.

To further corroborate this point, note that the performance of the baselines drops consistently in this scenario. TopK-U and k-NN suffer the most, due to the higher complexity of separating the sets, while TopK-D, although performing slightly better than in the previous setting, remains suboptimal due to its inherent imbalance. ImpDisp particularly struggles, as its reliance on impact differences does not generalize well to highly disjoint distributions, reinforcing the importance of using more structured methodologies.

Looking across datasets, the trends remain consistent. The CIFAR-10, CIFAR-20, and CIFAR-100 image datasets exhibit similar behavior, where our ForSId-via-RBSC retains its superiority across efficacy measures, though some baselines achieve closer results in lower-complexity tasks. The Wine Quality dataset is tabular and smaller in scale, thus it exhibits less pronounced differences across methods, but ForSId-via-RBSC still maintains the highest efficacy scores, showing that its effectiveness is not limited to image data.

An important remark, valid for all the experimental settings, is that the effectiveness of our method is independent on the size of the identified forget set. In fact, as shown in Tables 6.3, 6.4, 6.5, the models exhibit minimal accuracy change (Acc. Δ) for CIFAR-10, CIFAR-20 and Wine, indicating that utility is well-maintained, while achieving high unlearning efficacy (UE), meaning that the unwanted set is effectively forgotten. The tables show that our method is among the methods that identify the smallest forget set (aside from a few cases), despite having the best overall performance on the selected metrics. That means that other baselines need to remove at least as many samples, but still get outperformed (See Appendix 6.7.2 for an in-depth discussion). Determining the ideal or minimal size of a forget set for effective unlearning is a challenging theoretical problem, closely related to fundamental questions in learning theory, such as identifying the minimum number of samples needed to approximate a decision boundary (e.g., PAC learning [251] and VC dimension [252]). Intuitively, the maximum forget set is the set that does not include the samples that approximate the decision boundary (with the exception of those that define the unwanted behaviour). Accordingly, the forget set could be very large indeed, especially if the training set is large and the decision boundary is relatively simple. While we believe this is an interesting and valuable direction for future work, a rigorous analysis is beyond the scope of our current paper. Moreover, adding a maximum amount of samples to be removed in the formulation of our problem is straightforward, but beyond the scope of this initial exploration.

Lastly, the analysis of the running time should be considered carefully. While ForSId-via-RBSC does have a higher running time than some baselines, this is part of a bigger tradeoff. Many cases show that methods with lower running times are really ineffective, as they often remove too few samples or fail to achieve a meaningful separation between wanted and unwanted instances. The tradeoff between efficiency and effectiveness is crucial, and ForSId-via-RBSC 's performance justifies its computational cost. The observation that running time alone does not determine a method's viability is reinforced by the fact that some of the fastest baselines, such as TopK-D, deliver highly imbalanced results and, therefore, they are unpractical despite their lower computational cost.

Overall, the results indicate that the proposed ForSId-via-RBSC achieves a strong balance between efficacy, a reasonable forget set size, and an appropriate computational efficiency. It avoids the pitfalls of excessively large and excessively small forget sets, removing unwanted instances without disrupting the overall model performance.

While some baselines provide useful comparative insights, none achieve the same consistency across different experimental settings and datasets.

6.6 Conclusion

In this paper, we advance the state of the art in Machine Unlearning (MU) by introducing the novel FORGET SET IDENTIFICATION (FORSID) problem: given a trained machine-learning model, and user's unwanted and wanted evidence, find a subset of the training set utilized upstream to train the model (i.e., the "forget set") so that retraining the model on the set resulting from the removal of the forget set from the original training set makes the model unlearn the unwanted evidence and keep the wanted evidence. We formalize the FORSID problem, prove its **NP**-hardness, and design an algorithm for it based on a connection to RED-BLUE SET COVER. Extensive experiments attest high performance of the proposed algorithm. Notably, for future work, researching more solid parameter selection techniques, like bisection, could further improve the results of our algorithm. By defining and studying the FORSID problem, we fill a crucial gap in the MU landscape, allowing for extending the applicability of MU in all those contexts where providing a forget set as input to MU methods is not possible or overly difficult.

In the future, we plan to devise algorithms that tackle FORSID more directly (without passing through proxy problems), deal with approximation properties and algorithms for the proxy RED-BLUE FORSID problem, consider different forms of (un)wanted behavior, investigate ad-hoc methodologies for setting the parameters of our method(s), and experimentally test FORSID in additional scenarios and settings. Specifically, although ILP solvers are powerful, they have practical limitations in terms of scalability and computational efficiency. To address these constraints, we plan to investigate data engineering techniques such as instance selection [62, 197] or genetic algorithms [136], which can help us identify samples more efficiently.

6.7 Detailed Results

In this Section, we corroborate and integrate our results with more in-depth analysis. Specifically, in Section 6.7.1, we present the same tables of Section 6.5, enriched with standard deviations, previously omitted for brevity. In Section 6.7.2, we study the performance trends of FORSID and the baselines as a function of parameters T and k, respectively.

6.7.1 Standard deviation

Here we show Tables 6.6,6.7,6.8, that list the same values of Tables 6.3,6.4,6.5 (shown in Section 6.5 but enriched with standard deviation. The values shown in the following tables are the mean and standard deviations of 10 runs, as described in Section 6.4. The standard deviations reported in these tables are generally small compared to their respective means, indicating that the results are stable across the 10 runs.

This proves that the observed performance patterns are not artifacts of random variation but rather reflect genuine properties of the methods, reinforcing the reliability of the results presented in Section 6.5.

Across all scenarios, we observe that the standard deviations tend to be larger on the Wine dataset compared to the CIFAR datasets. This is an expected outcome given the characteristics of Wine: it is substantially smaller in both sample size and

feature dimensionality, which makes its evaluation metrics more sensitive to minor fluctuations in predictions. Furthermore, the Wine dataset suffers from severe class imbalance, where some classes contain only a handful of samples while others constitute the majority of the dataset [56]. As a result, any small change in the predicted labels can lead to considerable variations in evaluation metrics such as accuracy or F1-score, amplifying the observed standard deviations.

On the other hand, the CIFAR datasets, which are much larger and more balanced in class distributions, exhibit consistently low standard deviations. This is particularly evident for the FORSID method, which maintains remarkably stable performance across all repetitions, regardless of the class or scenario under evaluation. This consistency reinforces its capacity to outperform baseline methods reliably, providing further evidence of its robustness and effectiveness in practical settings. Overall, these additional results support the validity of the findings reported in Section 6.5.

TABLE 6.6: ONE VS OTHERS | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.605 ± 0.071	0.554 ± 0.100	0.657 ± 0.070	0.191 ± 0.036	0.639 ± 0.064	36814.818 ± 2400.515	58.122 ± 11.309
	TopK-U	0.639 ± 0.065	0.345 ± 0.121	0.932 ± 0.011	0.018 ± 0.004	0.585 ± 0.126	10205.091 ± 462.544	50.029 ± 0.781
	TopK-D	0.525 ± 0.010	0.055 ± 0.019	0.996 ± 0.003	0.001 ± 0.004	0.148 ± 0.045	613.182 ± 136.549	25.622 ± 0.872
	ImpDisp	0.655 ± 0.094	0.522 ± 0.150	0.788 ± 0.050	0.122 ± 0.030	0.676 ± 0.100	24278.250 ± 1736.286	2658.045 ± 18.428
	Ours	0.982 ± 0.008	0.999 ± 0.011	0.965 ± 0.009	0.077 ± 0.017	0.952 ± 0.012	19266.000 ± 743.27	1094.689 ± 15.321
CIFAR-20	k-NN	0.767 ± 0.053	0.608 ± 0.106	0.927 ± 0.007	0.011 ± 0.004	0.795 ± 0.063	7676.500 ± 553.122	149.123 ± 112.643
	TopK-U	0.843 ± 0.039	0.798 ± 0.079	0.889 ± 0.020	0.014 ± 0.006	0.878 ± 0.034	8988.100 ± 671.028	40.479 ± 8.478
	TopK-D	0.661 ± 0.021	0.322 ± 0.042	1.000 ± 0.001	0.006 ± 0.003	0.584 ± 0.046	1065.778 ± 40.335	31.510 ± 6.247
	ImpDisp	0.790 ± 0.060	0.948 ± 0.052	0.631 ± 0.086	0.102 ± 0.022	0.766 ± 0.062	25296.900 ± 3539.419	3223.776 ± 530.344
	Ours	0.986 ± 0.042	0.993 ± 0.023	0.979 ± 0.062	0.011 ± 0.024	0.983 ± 0.043	5247.000 ± 1208.212	1226.710 ± 112.642
CIFAR-100	k-NN	0.774 ± 0.099	0.626 ± 0.188	0.921 ± 0.012	0.034 ± 0.017	0.782 ± 0.121	8308.800 ± 653.706	20.272 ± 8.630
	TopK-U	0.902 ± 0.101	0.867 ± 0.193	0.936 ± 0.027	0.014 ± 0.004	0.913 ± 0.093	5098.500 ± 621.340	1.918 ± 0.450
	TopK-D	0.835 ± 0.074	0.669 ± 0.147	1.000 ± 0.000	0.007 ± 0.007	0.846 ± 0.087	672.400 ± 126.945	3.714 ± 0.753
	ImpDisp	0.799 ± 0.054	0.968 ± 0.091	0.630 ± 0.028	0.173 ± 0.017	0.742 ± 0.036	24618.100 ± 1731.776	322.440 ± 56.844
	Ours	0.968 ± 0.068	0.943 ± 0.139	0.993 ± 0.008	0.152 ± 0.021	0.873 ± 0.043	17023.667 ± 836.755	187.575 ± 50.513
Wine	k-NN	0.855 ± 0.128	0.880 ± 0.199	0.830 ± 0.064	0.038 ± 0.019	0.871 ± 0.106	2508.200 ± 1543.119	0.179 ± 0.184
	TopK-U	0.768 ± 0.157	0.800 ± 0.355	0.735 ± 0.089	0.039 ± 0.008	0.764 ± 0.181	2186.250 ± 218.494	1.326 ± 1.384
	TopK-D	0.744 ± 0.310	0.708 ± 0.450	0.780 ± 0.175	-0.015 ± 0.037	0.735 ± 0.405	115.500 ± 30.359	0.239 ± 0.071
	ImpDisp	0.775 ± 0.310	0.761 ± 0.450	0.790 ± 0.175	0.021 ± 0.037	0.763 ± 0.405	2310.600 ± 30.359	19.750 ± 2.147
	Ours	0.920 ± 0.030	1.000 ± 0.000	0.840 ± 0.020	0.028 ± 0.017	0.924 ± 0.010	1795.000 ± 439.820	9.848 ± 0.324

TABLE 6.7: ONE VS ALL | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.593 ± 0.065	0.560 ± 0.102	0.627 ± 0.061	0.190 ± 0.037	0.632 ± 0.060	36660.100 ± 2471.182	54.281 ± 6.424
	TopK-U	0.626 ± 0.058	0.349 ± 0.128	0.903 ± 0.019	0.018 ± 0.004	0.581 ± 0.130	10137.400 ± 441.964	50.680 ± 0.790
	TopK-D	0.503 ± 0.002	0.007 ± 0.005	0.999 ± 0.001	-0.001 ± 0.004	0.021 ± 0.013	38.000 ± 14.252	6.135 ± 0.200
	ImpDisp	0.661 ± 0.117	0.562 ± 0.225	0.760 ± 0.037	0.122 ± 0.023	0.662 ± 0.198	24621.300 ± 2414.427	2663.667 ± 58.910
	Ours	0.802 ± 0.011	0.695 ± 0.032	0.910 ± 0.012	0.033 ± 0.014	0.837 ± 0.078	10297.000 ± 4405.850	1281.942301 ± 32.488
CIFAR-20	k-NN	0.760 ± 0.049	0.626 ± 0.104	0.895 ± 0.012	0.007 ± 0.005	0.799 ± 0.059	9828.250 ± 644.165	48.670 ± 13.028
	TopK-U	0.823 ± 0.038	0.820 ± 0.079	0.826 ± 0.015	0.010 ± 0.005	0.867 ± 0.030	11907.875 ± 971.407	36.859 ± 1.116
	TopK-D	0.516 ± 0.003	0.032 ± 0.006	0.999 ± 0.001	-0.004 ± 0.003	0.090 ± 0.015	93.875 ± 11.777	5.903 ± 0.180
	ImpDisp	0.793 ± 0.024	0.948 ± 0.033	0.638 ± 0.034	0.074 ± 0.009	0.788 ± 0.023	23874.500 ± 1934.396	2664.712 ± 127.513
	Ours	0.850 ± 0.008	0.864 ± 0.020	0.835 ± 0.004	0.022 ± 0.004	0.880 ± 0.001	6858.000 ± 125.865	1237.051 ± 8.424
CIFAR-100	k-NN	0.800 ± 0.064	0.669 ± 0.130	0.931 ± 0.012	0.026 ± 0.006	0.823 ± 0.067	6963.625 ± 465.010	22.702 ± 10.069
	TopK-U	0.926 ± 0.034	0.963 ± 0.062	0.889 ± 0.027	0.020 ± 0.024	0.936 ± 0.028	5156.375 ± 622.996	1.890 ± 0.374
	TopK-D	0.676 ± 0.188	0.355 ± 0.380	0.996 ± 0.007	0.001 ± 0.004	0.441 ± 0.465	524.250 ± 217.462	4.170 ± 1.139
	ImpDisp	0.829 ± 0.034	1.000 ± 0.000	0.657 ± 0.069	0.177 ± 0.019	0.757 ± 0.035	24636.125 ± 1699.436	330.794 ± 72.977
	Ours	0.988 ± 0.008	1.000 ± 0.000	0.977 ± 0.015	0.187 ± 0.004	0.858 ± 0.006	20417.667 ± 260.316	112.443 ± 6.469
Wine	k-NN	0.797 ± 0.093	0.820 ± 0.157	0.775 ± 0.078	0.033 ± 0.017	0.832 ± 0.075	2368.500 ± 946.219	0.216 ± 0.095
	TopK-U	0.661 ± 0.167	0.728 ± 0.278	0.595 ± 0.058	0.024 ± 0.052	0.719 ± 0.140	3875.000 ± 116.167	3.324 ± 1.587
	TopK-D	0.587 ± 0.218	0.372 ± 0.403	0.802 ± 0.051	0.039 ± 0.012	0.488 ± 0.309	88.750 ± 92.662	0.374 ± 0.070
	ImpDisp	0.661 ± 0.105	0.615 ± 0.229	0.708 ± 0.028	0.001 ± 0.016	0.689 ± 0.129	2457.750 ± 117.452	25.294 ± 0.459
	Ours	0.744 ± 0.187	0.765 ± 0.359	0.723 ± 0.062	-0.029 ± 0.024	0.778 ± 0.226	2319.000 ± 194.504	11.209 ± 0.722

TABLE 6.8: MANY VS MANY | Comparison of ForSId-via-RBSC (Ours) with baselines methods. Benchmarking measures as the mean of 10 runs. Bold values are the best overall; underlined ones are second best. Efficacies (BE, UE, WE) are the most important measures.

		BE ↑	UE ↑	WE ↑	Acc. Δ ↓	H ↑	FS-Size	RunT (s) ↓
CIFAR-10	k-NN	0.552 ± 0.089	0.476 ± 0.072	0.628 ± 0.125	0.216 ± 0.034	0.586 ± 0.070	39190.600 ± 2164.686	121.306 ± 92.484
	TopK-U	0.579 ± 0.034	0.327 ± 0.028	0.831 ± 0.042	0.072 ± 0.010	0.559 ± 0.034	27916.400 ± 794.768	75.401 ± 13.846
	TopK-D	0.543 ± 0.013	0.093 ± 0.025	0.994 ± 0.006	0.017 ± 0.006	0.233 ± 0.055	6436.600 ± 816.194	103.775 ± 19.558
	ImpDisp	0.540 ± 0.025	0.265 ± 0.052	0.814 ± 0.065	0.103 ± 0.040	0.481 ± 0.044	23792.000 ± 2973.867	3276.857 ± 825.794
	Ours	0.869 ± 0.068	0.793 ± 0.092	0.944 ± 0.053	0.262 ± 0.037	0.776 ± 0.025	30727.800 ± 1127.532	1065.329 ± 10.010
CIFAR-20	k-NN	0.752 ± 0.011	0.534 ± 0.016	0.970 ± 0.007	0.001 ± 0.006	0.768 ± 0.011	3838.429 ± 59.997	46.738 ± 4.973
	TopK-U	0.733 ± 0.013	0.660 ± 0.012	0.806 ± 0.019	0.061 ± 0.004	0.769 ± 0.010	27496.143 ± 466.265	25.109 ± 0.930
	TopK-D	0.666 ± 0.008	0.333 ± 0.015	1.000 ± 0.001	0.009 ± 0.005	0.597 ± 0.015	10531.600 ± 176.618	25.686 ± 0.891
	ImpDisp	0.539 ± 0.039	0.370 ± 0.039	0.709 ± 0.055	0.065 ± 0.012	0.567 ± 0.038	23653.429 ± 1909.292	2717.737 ± 54.341
	Ours	0.945 ± 0.021	0.895 ± 0.042	0.996 ± 0.003	0.138 ± 0.010	0.849 ± 0.009	23344.200 ± 758.332	1314.937 ± 45.560
CIFAR-100	k-NN	0.722 ± 0.013	0.485 ± 0.022	0.958 ± 0.004	0.012 ± 0.003	0.727 ± 0.019	6152.000 ± 111.903	47.285 ± 4.797
	TopK-U	0.800 ± 0.005	0.787 ± 0.006	0.813 ± 0.017	0.189 ± 0.013	0.755 ± 0.011	28024.000 ± 319.839	13.336 ± 0.074
	TopK-D	0.709 ± 0.011	0.421 ± 0.020	0.997 ± 0.002	0.104 ± 0.007	0.653 ± 0.012	15558.000 ± 77.216	31.007 ± 2.626
	ImpDisp	0.467 ± 0.013	0.380 ± 0.003	0.553 ± 0.025	0.188 ± 0.008	0.508 ± 0.007	25987.000 ± 373.101	2724.077 ± 64.654
	Ours	0.998 ± 0.068	1.000 ± 0.092	0.997 ± 0.053	0.256 ± 0.037	0.795 ± 0.025	24534.000 ± 1127.532	1756.515 ± 93.538
Wine	k-NN	0.857 ± 0.071	1.000 ± 0.007	0.714 ± 0.148	0.032 ± 0.042	0.866 ± 0.081	5196.000 ± 255.766	1.728 ± 2.932
	TopK-U	0.921 ± 0.175	1.000 ± 0.304	0.842 ± 0.046	0.024 ± 0.049	0.927 ± 0.163	5194.000 ± 188.794	59.013 ± 0.066
	TopK-D	0.789± 0.157	0.697 ± 0.332	0.882 ± 0.017	0.073 ± 0.015	0.804 ± 0.298	1853.000 ± 263.272	23.323 ± 0.119
	ImpDisp	0.752 ± 0.157	0.658 ± 0.332	0.845 ± 0.017	0.014 ± 0.015	0.804 ± 0.298	2729.000 ± 263.272	262.115 ± 0.119
	Ours	0.801 ± 0.165	0.752 ± 0.338	0.851 ± 0.009	-0.003 ± 0.040	0.857 ± 0.104	2638.000 ± 9.238	124.070± 7.310

6.7.2 Parameter Study and trends

In this Section, we show a parameter study on parameters T (for our method FORSID) and k (for all the baselines). We report results for all tested parameter values and analyze the behavior of FORSID and the baselines as these parameters approach the total number of samples in the dataset.

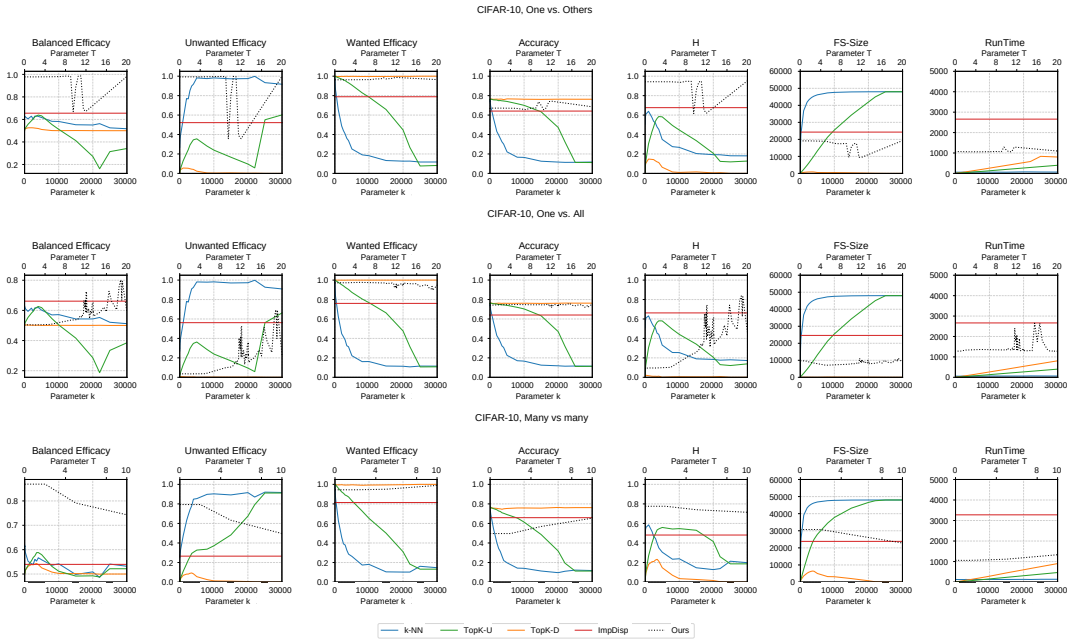


FIGURE 6.2: Comparison of ForSId-via-RBSC (Ours) with baseline methods on CIFAR-10 across three experimental settings: ONE VS OTHERS, ONE VS ALL, and MANY VS MANY, as a function of parameters k and t. The x-axis below represents parameter K, while the x-axis above represents parameter T for FORGET SET IDENTIFICATION.

Figure 6.2 shows the the trends exhibited by each baseline and FORSID when parameters K and T change, for each scenario (from top to bottom). The figure clearly conveys the patterns of each method's behavior under these parameter changes. k-NN and TopK-U have very similar behavior, as they are very similar methods (refer to Section 6.4: when parameter K increases, both tend to become extremely disruptive, increasing their Unwanted Efficacy but at the same time plummeting the Wanted Efficacy, as they are destroying the utility of the model and flattening the predictions as a result. This is also evident from the stark drop in accuracy they cause on all three scenarios and the ever-increasing size of the Forget Set they identify. This is expected: both k-NN and TopK-U strictly increase the size of the Forget Set when K increases.

On the other hand, TopK-D exhibits the opposite behavior: since increasing the parameter K builds a bigger set to remove from the initial set of identified samples, when K increases too much, the Forget Set identified by TopK-D collapses to 0 (refer to FS-Size across all three scenarios). As a result, applying this method is basically like doing nothing: the Wanted Efficacy is extremely high (as no wanted samples are changing prediction) but the Unwanted Efficacy is 0 (as no unwanted samples are changing prediction either), effectively making the method useless.

ImpDisp does not depend on any parameter and is represented by a straight line.

Finally, FORSID is consistently the best performer overall. While each of the baselines, for the aforementioned reason, excel in one area while collapsing on all others,

FORSID demonstrates balanced and robust performance across all evaluated metrics and scenarios. This is well captured by the metric H, an harmonic mean of all the others (refer to Section 6.4), where FORSID outperforms all others, often by a large margin. The change of parameter T does have an impact on FORSID, but it's often negligible with respect to other baselines, which are by far more impacted by the change of parameter K. However, a parameter tuning on FORSID can yield better results.

Moreover, from these trends, it is clear how FORSID is also among the methods that find the smallest Forget Set (excluding TopK-D, that as we explained collapses to 0), while still selecting the right samples to overperform the other baselines across all metrics.

For the sake of completeness, Figures 6.3 and 6.4 illustrate the same trends for CIFAR-20 and CIFAR-100, respectively. As the observed patterns are consistent with those already discussed, we omit further detailed commentary here. However, both images show how FORSID outperforms the baseline while consistently finding the smallest Forget Sets.

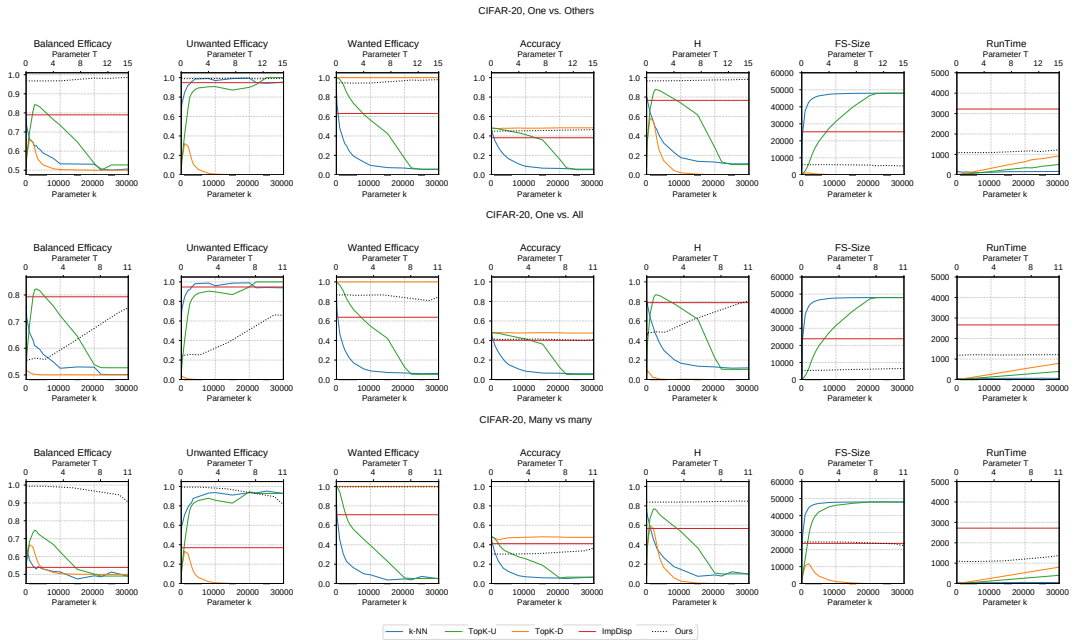


FIGURE 6.3: Comparison of ForSId-via-RBSC (Ours) with baseline methods on CIFAR-20 across three experimental settings: ONE VS OTHERS, ONE VS ALL, and MANY VS MANY, as a function of parameters k and t. The x-axis below represents parameter K, while the x-axis above represents parameter T for FORGET SET IDENTIFICATION.

6.7.3 Alternative Influence Function

In our paper, we focused on implementing our method and all baselines with Representer Points because of their additive decomposition property (Section 6.4). In this section, we additionally evaluate all methods in the One-vs.-Others setting using the influence function defined in [143]. Specifically, for a training sample z_i and a target sample z_t, we use the standard upweighting influence

$$I_{\text{up}}(z_i, z_t) = -\nabla_\theta \ell(z_t, \theta)^\top H_\theta^{-1} \nabla_\theta \ell(z_i, \theta),$$

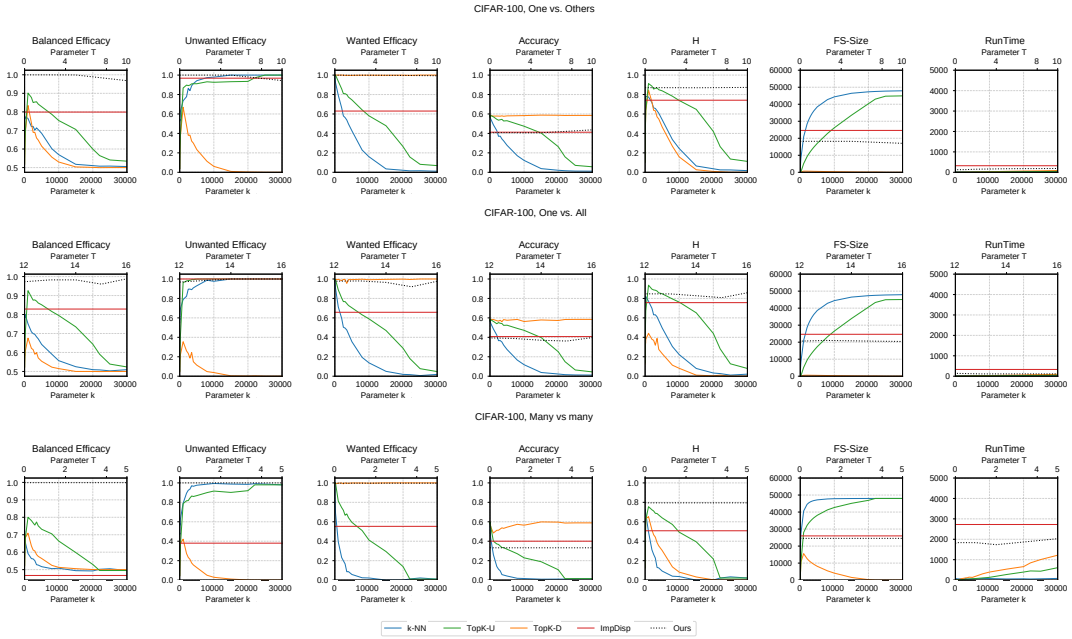


FIGURE 6.4: Comparison of ForSId-via-RBSC (Ours) with baseline methods on CIFAR-100 across three experimental settings: ONE VS OTHERS, ONE VS ALL, and MANY VS MANY, as a function of parameters k and t. The x-axis below represents parameter K, while the x-axis above represents parameter T for FORGET SET IDENTIFICATION.

which measures how an infinitesimal increase in the weight of z_i affects the loss on z_t.

FORSID works regardless of the function used to estimate the influence of each sample, with the only modification needed being the tuning of parameter T. In particular, influence values are signed: positive values indicate that upweighting z_i increases the target loss, while negative values indicate that upweighting z_i decreases it. Since unlearning removes points (i.e., downweights them), the effective removal effect has the opposite sign, and the threshold parameter T must be re-calibrated accordingly. Therefore, FORSID remains function-agnostic, with the only practical adjustment being the tuning (and sign-consistent interpretation) of T under the influence-function scale.

Figure 6.5 shows the results for FORSID and the baselines when employing this alternative influence function.

The balanced efficacy of our method remains one of the best, alongside accuracy (as shown in the plot of H). `k-NN` is the only baseline that is able to surpass of our method in terms of H, but it comes at the cost of a bigger forget set size. The RunTime of our method with this influence function is the highest: this can be mitigated by approximating the inverse Hessian matrix rather than computing it directly.

Despite using a different influence function without additive decomposition property, our method remains among the best in terms of H and Forget Set Size, proving its flexibility and effectiveness.

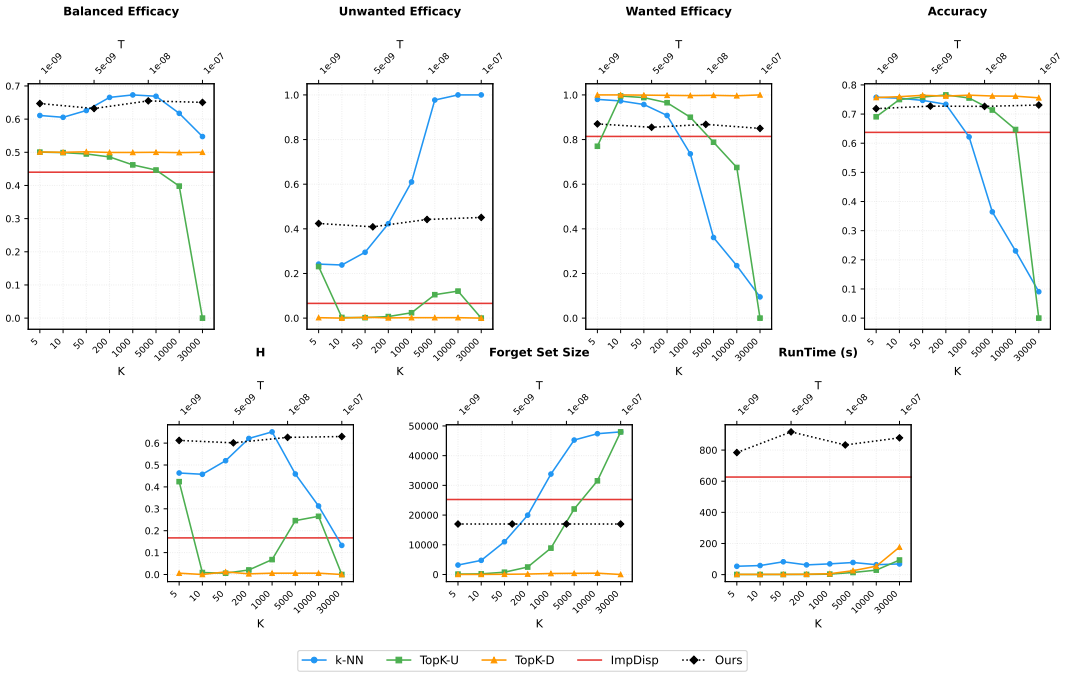


FIGURE 6.5: Comparison of ForSId-via-RBSC (Ours) with baseline methods on CIFAR-10 on the `ONE VS OTHERS` setting using an alternative influence function.

Part II

Building Trustworthiness through Fairness

Chapter 7

Bias in the data

Bias impacts human beings as individuals or groups characterized by a set of legally-protected sensitive attributes (e.g., their race, gender, or religion) by under representing them or by representing them in a wrong manner. If not managed, the inequalities reinforced by search and recommendation algorithms can lead to *severe societal consequences*, such as discrimination and unfairness [118]. Both *search* and *recommendation* algorithms provide users with ranked results that fit and match their needs and interests. Both tasks often convey and strengthen bias in terms of *imbalances* and *inequalities*, mainly if they rely on or encompass machine learning algorithms as those which solve classification problems. For this reason, assuring that search and recommendation systems are fair and do not apply discrimination among any kind of population has become of paramount importance, mainly because they are pervasive in several domains [7, 25] - e.g., justice [207], health care [235], education [13], etc. Clearly, in the context of **Trustworthy AI**, we want systems that do not inherit any bias from the data, and are, as such, considered fair.

The importance of having *fair* and *egalitarian* systems must be a fundamental step to solving some of the 17 sustainable development goals proposed by the United Nations[1]. In particular, we will possibly rely on information systems to accomplish goals 5 (gender equality) and 10 (reduced inequalities) on a large scale. If those information systems include some AI or Machine learning techniques (such as classification and recommendation), it will be essential to identify and mitigate properly algorithmic *Bias and Fairness*.

One of the most obvious presence of bias comes from the imbalance between genders. The problem of *gender gap* has been widely considered and analyzed in the literature under several contexts and domains, like health [214], justice [10], or education [187]. Concerning education, the problem of gender gap gained considerable relevance over the years, and several papers studied this issue from both a technical and sociological point of view [17, 183]. The relevance of this topic is also highlighted by international programs such as the 2030 United Nations Sustainability Development Goals [189] or European Union programs such as EUGAIN, which aims to improve gender balance in Informatics at all levels [30]. Exploring this issue from an academic perspective can provide valuable insights into the systemic challenges that perpetuate the gender gap within educational institutions, enabling the development of evidence-based strategies and policies to promote equity and inclusion.

[1] https://sdgs.un.org/goals

7.1 Related Work on Bias and Fairness

7.1.1 Definitions of Bias and Fairness

Bias (and relative unfairness) can arise from different sources and be defined in several ways. In [180] the authors highlighted that bias can be generated from:

- the **data** used to train the ML algorithms (e.g., *Measurement bias* [237], *Omitted Variable bias* [54, 29], or *Representation bias* [237]);
- the **algorithm** which may introduce bias in the users' behavior (e.g., *Algorithmic bias* [14]);
- the **population** which generates the data used to train the models (e.g., *Historical bias* [237], *Population bias* [196], or *Social bias* [14]).

The former definitions of bias, with the only exception of *Algorithmic bias*, which is strongly related to the ML algorithm, can be grouped into two macro-categories of bias:

- **Unbalanced Groups bias**: in which the bias is generated by an unequal distribution of instances in the population (e.g., *Representation bias*, *Historical bias*, *Social bias*, *Population bias*)
- **Confounding Variables bias**: in which the bias is generated by a wrong interpretation or representation of instances in the population (e.g., *Measurement bias*, *Omitted Variable bias*)

Our proposed approach addresses the first macro-definition of bias by mitigating the unequal distribution of instances through an optimal balancing of them inside the population.
Concerning fairness definitions, *Demographic (Statistical) Parity (DP)* [152, 86] is one of the most used definitions of *group fairness* [180], which assumes the independence among the predicted positive label y_p and the sensitive variables $S_1, \ldots, S_n$. It is defined formally as follows:

Definition 4 (Demographic Parity)**.** Let $\hat{Y}$ be the predicted value, y_p the positive label and S a generic binary sensitive variable where $S = 1$ and $S = 0$ identify, respectively, the privileged and unprivileged groups. A predictor is *fair* under *Demographic Parity* if:

$$P(\hat{Y} = y_p|S = 1) = P(\hat{Y} = y_p|S = 0) \tag{7.1}$$

A different formulation for the DP is the *Disparate Impact (DI)* [99], which considers the ratio among the two probabilities. In this case, following the *80% rule* [99], the value must be between 0.8 and 1.2 in order to have *fairness*. DI is defined formally as follows:

Definition 5 (Disparate Impact)**.** Let $\hat{Y}$ be the predicted value, y_p the positive label and S a generic binary sensitive variable where $S = 1$ and $S = 0$ identify the privileged and unprivileged groups, respectively. A predictor is *fair* under *Disparate Impact* if:

$$0.8 \leq \frac{P(\hat{Y} = y_p|S = 1)}{P(\hat{Y} = y_p|S = 0)} \leq 1.2 \tag{7.2}$$

Equalized Odds (EO) [119] is the third definition of fairness we consider which overcomes the limitation of DP by not removing the correlation among the true and predicted outcomes [254, 119]. In fact, a classifier is considered fair under EO if the probability of an item to be positively classified is the same with respect to the sensitive variable and the ground truth. EO is formally defined as follows:

Definition 6 (Equalized Odds)**.** Let $\hat{Y}$ be the predicted value, Y the true value, y_p the positive label and S a generic binary sensitive variable where $S = 1$ and $S = 0$ identify the privileged and unprivileged groups, respectively. A predictor is fair under *Equalized Odds* if:

$$P(\hat{Y} = y_p | Y = y, S = 1) = P(\hat{Y} = y_p | Y = y, S = 0) \quad y \in \{y_1, \ldots, y_n\} \tag{7.3}$$

Both DP and DI fall into the *We Are Equal* metrics family, which holds that all groups have similar abilities concerning the task (i.e., have the same probability of being classified in a certain way). On the contrary, EO resides in the *What You See Is What You Get* family, which states that the observations reflect the ability with respect to the task (i.e., an item should be classified in a certain way only if the other attributes imply it) [104].
All these definitions were initially proposed for binary classification problems ($y_p = 1$), but they can be easily extended to the multi-class classification domain by identifying one positive label value among the possible ones ($y_p \in \{y_1, \ldots, y_n\}$).

7.1.2 Bias Mitigation

Over the years, many methods have been proposed to mitigate bias at different levels of data processing[180, 39]. In particular, we distinguish among [64]:

- **Pre-processing** methods, which modify the data to remove the underlying bias, such as, [133, 99];
- **In-processing** methods, which change the learning algorithm to remove discrimination during the model training process, such as [78, 3];
- **Post-processing** methods, which re-calibrate an already trained model using a holdout set not used during the training phase, such as [119, 201].

In general, the sooner a technique can be applied, the better because it can be chained with other bias mitigation methods in the later processing phases [261, 5].
Among the different machine learning problems (i.e. classification, regression, clustering, etc.), the classification task has been the most addressed in bias mitigation [180, 39]. In the following, we will focus on stable methods[2] to improve fairness in the classification task.
Most of the methods available in the literature focus only on binary classification with one sensitive variable [180]. Among them, one widely adopted *pre-processing* method is the *Sampling* algorithm proposed by [133]. This method balances both privileged and unprivileged users in the case of binary classification with a single sensitive variable. Formally, let be S the sensitive variable with $\{w, b\} \in S$ representing the privileged and unprivileged groups, respectively, and let be Y the target label with $\{+, -\} \in Y$ defining the positive and negative outcomes. The *Sampling* algorithm first splits the original dataset into four groups:

[2]Stable methods are the ones having an available and stable implementation.

- Deprived group with Positive label (DP): all instances with $S = b \wedge Y = +$;
- Deprived group with Negative label (DN): all instances with $S = b \wedge Y = -$;
- Favored group with Positive label (FP): all instances with $S = w \wedge Y = +$;
- Favored group with Negative label (FN): all instances with $S = w \wedge Y = -$.

Then, for each group, the algorithm computes its *observed* and *expected* sizes. Finally, it balances the groups iteratively by randomly adding and removing instances until the *observed* sizes of the groups are equal to their *expected* ones.

The *Sampling* algorithm is the starting point for the definition of DEMV. In fact, we have extended this algorithm to the multi-class classification domain with multiple sensitive variables and we have employed different instance generation strategies during the balancing process. Very few methods are able to mitigate the bias in the multi-class classification problems [201, 3].

Among those, there is the *Blackbox post-processing* method proposed by [201]. The authors extend the method proposed by [119] to the multi-class setting. Their approach involves the construction of a linear program over the conditional probabilities of the adjusted predictor $P(Y^{adj} = y^{adj} | \hat{Y} = \hat{y}, A = a)$ such that the desired fairness criterion is satisfied by those probabilities. In order to build the linear program, the authors formulate both the loss and fairness criteria as linear constraints in terms of the protected attribute conditional probability matrices. Then, this linear program is used to find the label value, among the possible ones, that minimizes both the loss and the fairness constraints.

An *in-processing* method that solves unfairness in multiple classification settings is the one presented by [3]. The algorithm addresses two definitions of fairness at once: *Demographic Parity* and *Equalized Odds*. The authors formulate such definitions as linear constraints and then build an Exponentiated Gradient (EG) reduction algorithm [140] that yields a randomized classifier with the lowest error subject to the desired fairness constraints. The method follows a MinMax approach in which the players try to minimize the given constraint and maximize the classifier's score. The authors also propose a simplified Grid Search version of the algorithm (GRID), which generates a sequence of relabelling and reweightings, and trains a predictor for each one. The values yielding the best *accuracy* and *fairness* trade-off are selected and thus returned. Although the authors study their algorithms mainly in binary classification problems, they also show how their method can be applied to regression and multi-classification problems.

As fairness metrics are based on the fundamental criteria of independence (equal representation), separation (equal performance), and sufficiency (equal confidence), any two of these three criteria are generally mutually exclusive. Therefore, recent research [69, 45] has shown that no bias mitigation method in the literature can optimize all fairness metrics simultaneously.

To the best of our knowledge, most of the methods in literature are primarily designed for binary classification problems, and few of them can be applied in the *pre-processing* phase. Moreover, only three stable approaches are realized to mitigate bias in multi-class classification problem, and none works in pre-processing phase.

Our proposed method goes beyond the state of art since it works in the pre-processing stage considering multiple sensitives variables, and both binary and multi-class classification task. Since it is a pre-processing method, it can be chained with other algorithms in later processing steps. Finally, it outperforms the state of art in most of the considered settings, as it is shown in the experimental section 9.3.

7.1.3 Gender Gap in Software Engineering

To address how the issue of gender bias in the SE community has been analysed over the years, we review papers from the following relevant venues about gender equality in SE: "Gender Equality, Diversity, and Inclusion in Software Engineering" (GE@ICSE) workshop, the "Software Engineering in Society" track of the ICSE conference, and the EUGAIN European project. To avoid bias in the selection of papers, following the process described in [260], two authors independently read the title and abstract of all the accepted papers and selected papers specifically focusing on gender discrimination and inclusion. Papers selected by both authors were included, while papers selected by only one author were discussed together and eventually included if relevant.

Several works analyse the issue of gender gap in SE from an industry perspective, either addressing the current status of women's employment in SE fields [258, 202] or providing suggestions and insights on how to increase the participation of women in STEM fields [246, 239, 147, 146, 280, 238]. Concerning works analysing the gender gap issue from a SE academic perspective, Santiesteban et al., investigate the issue of gender discrimination in the teaching evaluation of students on SE courses. Through an analysis of private data from an American university, the authors show how there is an intrinsic gender bias in the evaluation of professors [221]. Tahsin et al., discuss instead how an awareness-based approach can help address the gender gap issue in Bangladesh universities [239]. Following a similar awareness-based approach, Hyrynsalmi reports the diversity and inclusion approaches adopted by several SE teachers in Finnish universities [127]. Marquard et al., analyse instead how single-gender interdisciplinary classes can help women's engagement in SE fields [175]. Similar to the work proposed in this Section, Nabot et al., analyse the trend of women's participation in computer science studies in Spain [190]. Finally, Motogna et al., perform a qualitative analysis of the issues impacting women's academic careers in computer science in the context of Romania [188].

From this analysis, we observe how there is a strong interest in analysing the gender gap issue in the SE community. Several works have been proposed to analyse and enhance women's participation both in SE industries and academia. However, no papers study the gender gap issue in academic promotions in SE and informatics communities. This work aims to fill this unexplored area and increase awareness of current challenges and insights about gender equality in academic positions by analyzing data to understand, through a data-driven systematic approach, whether there are mechanisms, patterns, or dynamics that favour the gender gap.

7.1.4 Gender Gap in Academia

Differently than the previous section, we make a wider analysis considering papers dealing with the gender gap in other academic domains. Our aim is to identify related work on the gender gap to identify approaches similar to ours to consider for wider consideration. In fact, our aim is also to analyse how the issue of the gender gap in academia has been analysed over the years. To this goal, we performed a lightweight systematic mapping study to select relevant works. In the following, we summarise the outcome of our study.

Analysing the issue of the gender gap in academia has gained considerable relevance in the literature over the last years, as also motivated by actions from the European Union (EU) like the EUGAIN project [30] or by the United Nations (UN) 2030 Sustainable Development Goals (like *Goal 4: Quality Education* or *Goal 5: Gender*

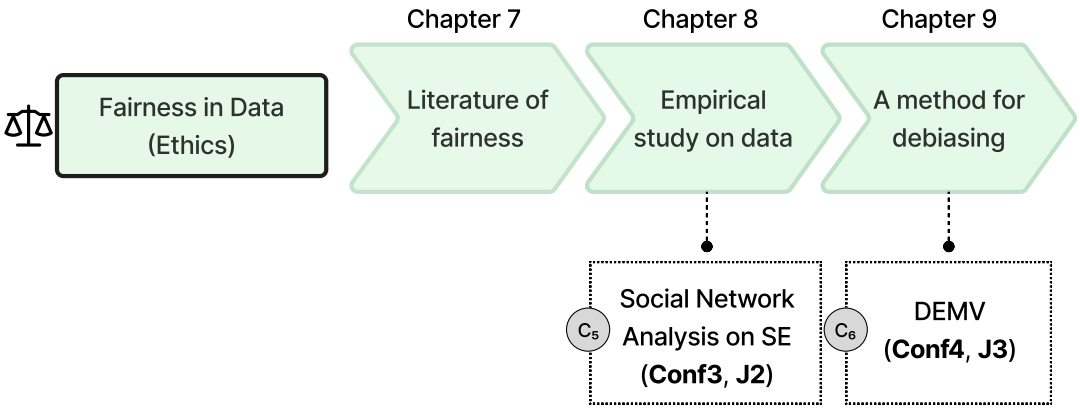


FIGURE 7.1: The research process we followed for the topic of Fairness, related to the TD of Ethics. Each step, where applicable, is annotated with the respective contribution of this thesis.

Equality) [189]. These actions from UN and EU can also explain how the majority of works focus on academic contexts either in Europe or in North America (i.e., the western side of the world). Concerning single countries, we want to highlight how Italy is the country with the most papers on this topic after the US. The reason for this trend can be explained by the fact that the MIUR released public data about people employed in academia [210].

7.2 Our Contribution

As depicted in Figure 7.1, our contribution to the Ethics dimension of Trustworthy AI is applied through studying fairness in data and advancing the field in this direction. Starting from the comprehensive overview of the literature described in Section 7.1, we highlight how most of the proposed analyses are related to broad academic areas, and, in particular, there is no paper addressing the issue of gender bias within the software engineering and informatics communities. Instead, analyses focusing on a specific academic area are needed since each area has a typical set of implicit and explicit promotion rules [223]. Moreover, we notice how most papers use data coming from private sources (like surveys or admission letters) , making such analyses difficult to reproduce and extend.

In an effort to fill these gaps, we analyze how gender gap, specifically, is present in the Software Engineering Community, through a Social Network Analysis, described in Chapter 8. Following the trend of other papers focusing on the Italian context, we also use data from the MIUR public database, but, differently from other works, we integrate these data with other international sources like Scopus or Google Scholar. In addition, we employ both a descriptive and inferential statistical approach using network analysis techniques and a formal bias metric (the Disparate Impact [99]) to assess the amount of gender bias in academic promotions. This gives us empirical validation and motivation for how bias in the data, even relational structures like graphs, can be subtle but important.

Building on this knowledge, we then advance the state of the art with DEMV, a Debiaser for Multiple Variables, presented in Chapter 9.

Within the bias literature, reported in Section 7.1, we notice that, despite the fact that it is widely adopted and constitutes a building block for personalization and search systems, the multi-class classification problem is still not effectively addressed [130].

In addition to fairness, a system must also have high accuracy of the predictions in order to be usable in the real world. However, most of the time the mitigation of bias negatively influences the accuracy of the predictions [39]. This trade-off must be managed, for example, by properly generating or modifying the instances of a dataset in a pre-processing context or by properly modifying the behavior of a method in an in-processing context.
For the aforementioned reasons, with DEMV, we focused, as a building block of other information access systems, on a bias mitigation method capable of *i)* managing an arbitrary number of sensitive variables in the multi-class classification scenario *ii)* preserving the accuracy of the predictions.

Terminology

Following the well-established research practices [2], in this work, we adhere to the convention where the term “gender” refers to the birth sex. We want to note that the literature commonly uses the term “gender” as a biological sex, even though we acknowledge that this might differ from its meaning regarding gender identity.
Moreover, we distinguish between the terms “gender bias" and “gender gap". We use “gender bias" to refer to discrepancies that are formally measured through the formal bias metric. Conversely, “gender gap" is used to describe behavioral mismatches or observable differences in the networking habits of men and women. This distinction allows us to discuss both the potential biases that could be influencing professional networking as well as the empirical differences in behavior.
Finally, we use the word “Researcher” to identify the Italian academic position corresponding to the international “Assistant professor”.

Chapter 8

Uncovering gender gap in academia

Bias in data can be explicit, or it can be embedded more subtly within structural relationships, such as those between nodes in a graph. For our first effort for the topic of bias, we focused on uncovering and evaluating gender gap bias present in academia, and more specifically in the Software Engineering community, by analyzing the structure of co-authorship networks.

As discussed in Chapter 7, most research on the gender gap in academia concentrates on students' educational experiences, often overlooking other important contexts [16]. In this Section, we address the gender gap in education from a broader academic and multi-faceted perspective, focusing specifically on academic promotions in Software Engineering (SE).

To achieve this goal, we collect all the needed data from several open repositories and process them with data mining techniques to make them suitable for analysis. The collected dataset, reusable for further evaluations, is thoroughly anonymized. We conduct a comprehensive analysis with a dual focus, comparing the Italian SE community with the worldwide SE community and the Italian Informatics community, analyzing both co-authorship relationships and the scenarios of academic promotions.

The dataset of reviewed papers, together with the anonymized dataset we collected, and the necessary code to replicate the experimental evaluation are available on Zenodo [70].

In particular, in order to spotlight the current situation about the gender gap in SE and Informatics communities, we aim to answer the following research questions:

RQ1. Do researchers of different gender groups differ in their behavior when selecting co-authors, including whether they consistently work with the same group or not?

This research question aims at highlighting involuntary behavioral patterns that hinder cooperation between researchers of different gender groups. As women are underrepresented in SE, any tendency for same-gender collaboration will disproportionately affect them and reduce their visibility, which is critical for career advancements. To address this question, we build social graphs based on co-authorship relations between authors in the worldwide SE and Italian SE Communities. We perform a Network Analysis on these graphs and evaluate several metrics, including Homophily, Clustering Coefficient, and Modularity.

RQ2. How is the Italian software engineering community dealing with gender bias in academic promotions with respect to the overall Italian informatics community?

This RQ aims at analysing the gender bias in academic promotions in the SE and informatics communities. Performing an analysis of specific academic areas or countries is relevant because each of them has a distinct set of implicit and explicit promotion rules [223]. Tackling this question needs additional data on the career trajectory of the researchers involved, which is challenging to collect on a global scale since different countries have different criteria for promotion. This is the reason why we restrict our analysis to the Italian context for this question. We gather the necessary public data from the Italian minister of education (*"Ministero dell'Istruzione e del Merito - MIUR"*) [210], and we employ Disparate Impact (DI) [99], a formal bias metric, to evaluate and compare the gender bias in academic promotions in the Italian SE community and the Italian Informatics community.

RQ3. What is the degree of gender bias in academic promotions between people with a high number and people with a mid-low number of publications and citations in the Italian informatics and SE communities?

The Italian academic promotion system in STEM is strongly based on bibliometric indexes like the number of publications and the number of citations [129, 4]. This RQ takes into account this factor by assessing if, given the same academic performance (i.e., number of publications and citations), there is still a gender bias in academic promotions within the Italian informatics and software engineering communities. To answer this question, we perform a deeper analysis by employing the number of citations and publications for each author. In particular, we divide the SE and informatics populations into two subgroups: people with a number of publications and citations greater or equal to the 75% of the whole population (called *Best Performers*), and people with a number of publications and citations lower than the 75% of the population (called *Mid-Low Performers*). Next, we compute the yearly DI for Best Performers and Mid-Low Performers groups and compare the results between the SE and informatics communities.

The contributions of this work are the following:

- we collect data on the Worldwide SE community, build co-authorship graphs, and perform Network Analysis to unearth common patterns. (RQ1, RQ2)
- we extend the evaluation of DI on the communities' Best performers and Mid-low performers (RQ3)

This dual assessment of co-authorship networks and academic promotions provides a comprehensive view of gender gap within the SE field. It offers insights into the collaboration dynamics among researchers and the biases affecting their career advancements, allowing us to infer significant results on the state of gender gap within the worldwide SE, Italian SE, and Italian Informatics Communities. Moreover, as demonstrated by other works on the field of gender equality in SE [239, 240, 188], highlighting current challenges and insights about gender equality in the informatics and SE academic communities can increase the awareness on such topics and motivate further research on the field.

8.1 Experimental Settings

In the following, we describe the data collection pipeline, methodology and experimental settings for the Network analysis (8.1) and the evaluation of Disparate Impact

on the collected data (8.1). A comprehensive description of the collected data is then presented in Section 8.1.2. Figure 8.1 depicts the full experimental settings workflow. The data sources we employ are both Italian and international. As international sources, we use data from the Scopus API [88] and from Google Scholar [224]. Concerning Italian sources, we extract data from the MIUR website [210], which contains all the information about people employed in Italian academia. Starting from the collected data, we employ two different processing pipelines. The pipeline for Worldwide/Italian SE (shown on the left side of figure 8.1 and described in detail in Section 8.1) leverages data from Scopus and Google Scholar to obtain co-authorship graphs of both the Worldwide and Italian SE Communities. These graphs are subjects to our Network Analysis and Academic Performance evaluation, aimed to answer the **RQ1**. In particular, we extract patterns exhibited by both gender groups when choosing co-authors and publishing their work. The second pipeline (described in Section 8.1 and shown on the right side of figure 8.1) integrates the international data extracted from Scopus and Google Scholar with the career data obtained from the MIUR website. The collected data are then processed and filtered (using the process described in Section 8.1) into two distinct datasets: one having information on the Italian SE community (*SE Datasets* in figure 8.1) and one having information on the overall Italian informatics community (*INF Datasets* in figure 8.1). We use these datasets to evaluate the Disparate Impact [99] (i.e., the formal bias metric of choice) on the promotion patterns of Researchers to Associate Professors and Associate Professors to Full Professors on the overall INF and SE Italian communities to answer **RQ2**. It is important to note that the academic role labelled as “Researcher" in Italy corresponds to what is typically known as “Assistant Professor" in other countries. Finally, the SE and INF datasets are further filtered to identify people with a number of publications and citations higher than 75% of the whole population (i.e., *Best Performers*). We then compute the DI on the Best Performers and compare the results with the DI computed on the rest of the population (i.e., *Mid-Low Performers*) to answer the **RQ3**.

8.1.1 Pipelines Description

Graphs construction pipeline and selected metrics

This Section describes in detail the pipeline depicted on the left in figure 8.1 (i.e., the *Worldwide/Italian SE* pipeline). The goal of this pipeline is to build graphs for the subsequent Network Analysis and Academic Performance Evaluation. We then describe the metrics we selected for these tasks.

The data employed for this pipeline are the data we collected from Google Scholar and the Scopus API (as described in section 8.1.2) to build a dataset of SE. The dataset is then further processed to construct a worldwide social graph of SE, based on the co-authorship relationships between them. In particular, each node refers to a researcher, and there is an edge between nodes if the researchers are co-authors in some of their papers. This graph is the subject of the network analysis presented in Section 8.2.1. From the worldwide social graph, we obtain the subgraphs of Italian SE by filtering the worldwide SE dataset using information from Google Scholar.

The Network Analysis aims to unearth patterns in the co-authorship selection among researchers of different gender groups. To this end, we selected metrics that are commonly used in the field of social network analysis to quantify the strength and nature of relationships within the co-authorship network. We perform a comprehensive Network Analysis on these graphs by measuring the following properties:

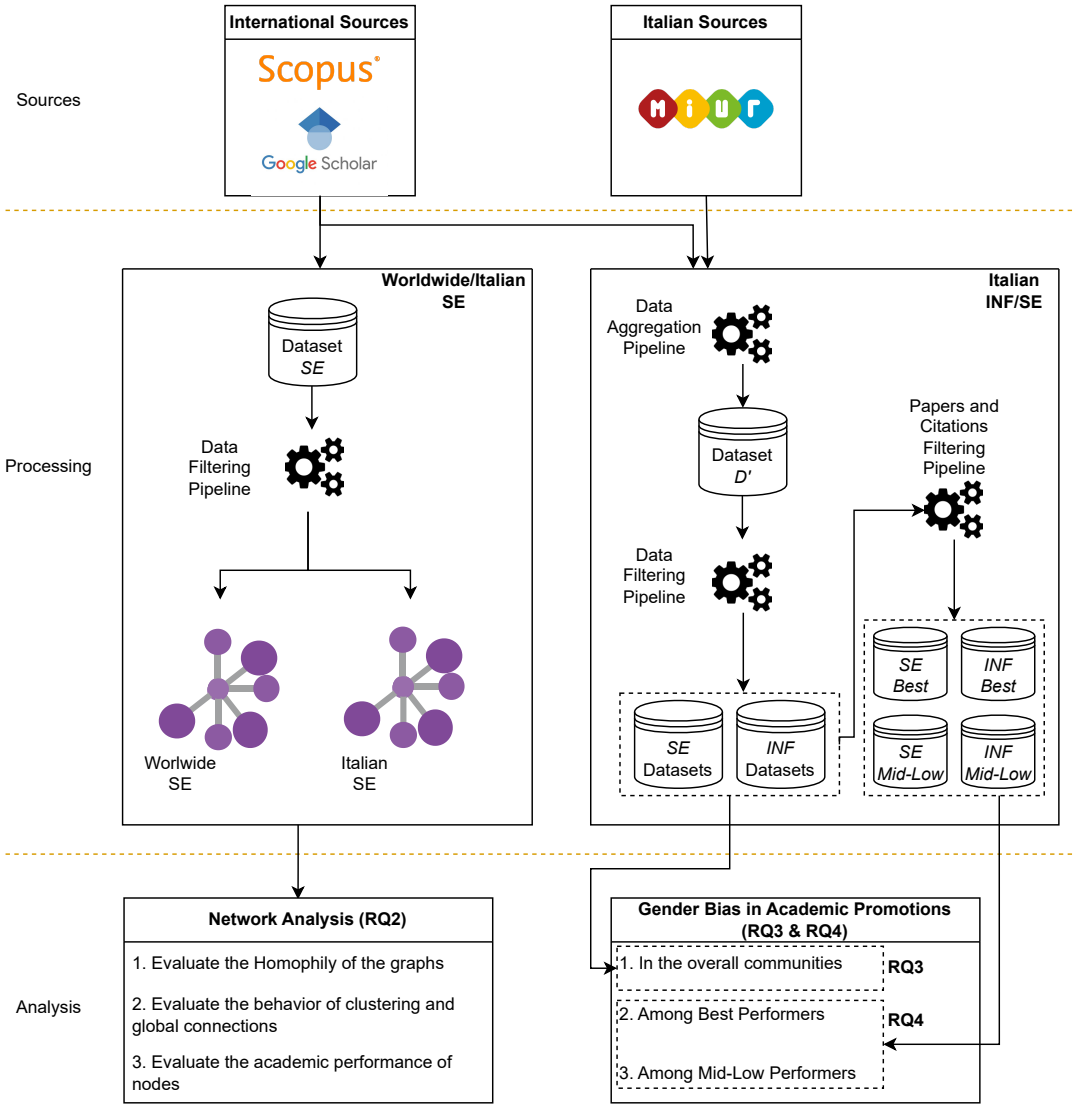


FIGURE 8.1: Experimental Settings pipelines, divided in Sources, Processing, and Analysis.

- **Homophily**: Homophily is defined as the tendency of a group of individuals to seek connections with similar individuals. In our context, homophily is the tendency of researchers to co-author papers with researchers of the same gender. We measure homophily via two different metrics:
  - *Homophily*: defined as the ratio of cross-gender edges over all edges [63]. This is the probability that, given a randomly chosen edge in the graph, the endpoints will be of different genders. Formally, this value is

$$H = \frac{|e_{cross}|}{|e|}$$

, where e_{cross} is the set of cross-gender edges. Assuming a ratio p of men, and a ratio q of women, the ideal value, indicating no homophily, is $w = 2pq$.

– *Coleman Homophily Index*: normalizes the homophily with the maximal value of excess homophily. The formula is

$$C_h = \frac{H - w}{1 - w}$$

with H and w defined as before [63]. In this case, the ideal value is 0.

- **Clustering Coefficient**: The Clustering Coefficient measures, from 0 to 1, the tendency of nodes in a graph to cluster together in tightly-knit groups. In our context, this is the tendency of researchers to co-author papers with a fixed set of people.

  More specifically, the Clustering Coefficient of a node u is defined as [222]:

$$CC(u) = \frac{2|T_u|}{deg(u)(deg(u) - 1)}$$

  where T_u is the set of triangles through node u, and $deg(u)$ is the degree, i.e. number of edges, of node u. A triangle through node u exists if $\exists v, m \in V | (u, v), (v, m), (m, u) \in E$, or simply put, if node u is part of a clique of size 3. The Clustering Coefficient of the entire graph that we compute in section 4.3 is the mean of the Clustering Coefficients of its nodes.

- **Modularity**: Modularity quantifies network structure by computing the extent of division into clusters, reflecting the strength of connections with tightly-knit groups of nodes. In our context, this metric measures the ability of researchers to build global connections outside of their working group. Modularity is defined as [192]:

$$M = \frac{1}{2m} \sum_{ij} (A_{ij} \gamma \frac{deg(i)deg(j)}{2m}) \delta(c_i, c_j)$$

  where m is the number of edges, A is the adjacency matrix of the graph, deg(i) and deg(j) are the degrees of i and j, respectively, and $\delta(c_i, c_j) = 1$ if nodes i and j are part of the same community, and 0 otherwise. γ is the resolution parameter that was set to 1 for our experiments.

After the Network Analysis, we also evaluate the academic performance of the two gender groups in both the Worldwide and Italian SE Communities. To this aim, we consider the number of publications and citations obtained by Scopus for each year, from 2015 to 2022, of each researcher, classifying them by gender.

Italian Informatics and SE Pipeline and selected metrics

This section describes the pipeline shown on the right side of figure 8.1 (i.e., the *Italian INF/SE* pipeline), employed to evaluate the degree of gender bias in the academic positions within the overall informatics (INF) and software engineering (SE) Italian communities. We then describe the formal bias metric we employ to this aim. The informatics community is the conjunction of Areas 1 and 9 of the MIUR scientific areas classification. [210]

This pipeline merges the data from both Italian and international data sources into a single dataset using the *name*, *surname*, *email*, and *affiliation* as join keys.

From such dataset, we provide an anonymized dataset D' on which we perform a set of filtering operations to obtain the final datasets that we use to compute bias

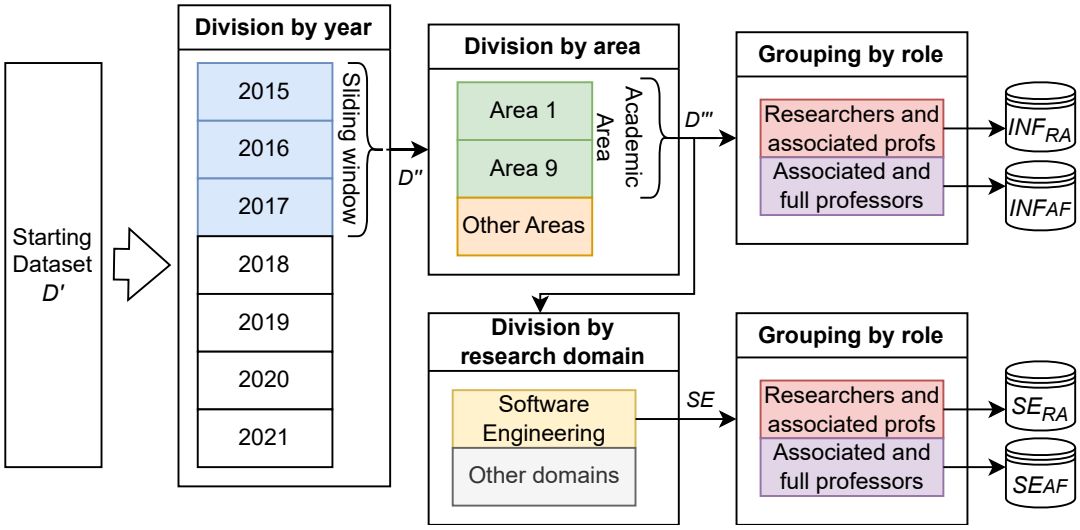


FIGURE 8.2: Filtering pipeline of the Italian datasets.

metrics yearly. The filtering procedure is depicted in Figure 8.2. Since we are interested in the evolution of bias in academic promotions year by year, the anonymized dataset D' is split according to a sliding time window of fixed size. In particular, we considered a sliding window of three years, starting from 2015. Hence, to gather metrics for 2019, with the sliding window size set to 3, we would slice D' from 2016 to 2018. After this operation, we obtain a partially filtered dataset D'' for each sliding window.

The subsequent step selects only specific scientific areas from D''. Because different domains have different promotion criteria, it would be incorrect to group them together. This study focuses on Areas 1 and 9 of the MIUR scientific areas classification, which refers broadly to Science, technology, engineering, and mathematics [210]. In this study, we refer to the conjunction of these two areas as the Informatics community. By selecting only researchers in these areas, we obtain a dataset D'''. From D''', we perform two different operations. In the first, D''' is split into two versions: one without records representing researchers (INF_{AF}) and one without Full Professors (INF_{RA}). In the second phase, D''' is refined by selecting individuals working in the SE field. To this aim, we leverage Google Scholar to find individuals who have expressed interest in *software engineering* or the following related topics: *software architecture*, *model-driven engineering*, *software quality*, and *software testing*. The SE dataset is then divided into two sub-datasets as done above: one consisting of only Researchers and Associate professors (SE_{RA}), and the other consisting of only Associate and Full professors (SE_{AF}).

As a result of the data pre-processing pipeline, four distinct datasets are created. Two of them are for the overall Italian INF community (INF_{RA} and INF_{AF}), while the other two are for the Italian SE community (SE_{RA} and SE_{AF}). It is worth noting how each dataset obtained as a result from the pipeline in Figure 8.2 represents the promotion to a specific role (i.e., from Researchers to Associated professors, and from Associated to Full professors). Finally, we only preserve data for workers employed at an Italian university for the entire time window. This decision was made to ensure that our analysis is not influenced by individuals achieving promotions in other countries, with possibly different criteria.

Once the final yearly datasets INF_{RA}, INF_{AF}, SE_{RA}, and SE_{AF} are constructed, the experiments can occur.

As mentioned, the experiment aims to measure the amount of gender bias in academic promotions and analyze its variation over the years. To calculate the amount of bias, we use the Disparate Impact metric [99]. This metric measures the probability of having a *positive outcome* while being in the *privileged* or *unprivileged* group and is defined formally as:

$$DI = \frac{P(Y = y_p|X = x_{unpriv})}{P(Y = y_p|X = x_{priv})}$$

where Y is the label, y_p is the positive outcome, X is the sensitive variable, and x_{unpriv} and x_{priv} are the values identifying the unprivileged and privileged groups, respectively. The more this metric is close to one, the fairer the dataset.

In our context, the label assigned to a person represents their position for that particular year. In the analysis between Researchers and Associate Professors, the positive label is *Associate Professor*, while in the analysis between Associate and Full Professors, it is *Full Professor*. The sensitive variable is *gender*, where *men* and *women* are the privileged and unprivileged groups, respectively.

The experiments we perform are threefold: on the Overall communities, on the Best performers, and on the Mid-Low performers. For the Overall communities, for each final yearly dataset (INF_{RA}, INF_{AF}, SE_{RA}, and SE_{AF}) and for each year in the considered range (2018-2022), we compute the DI between the two subsets contained in the dataset (either Researchers and Associate Professors or Associate Professors and Full Professors). We also compute the cardinality of each subset per year. Next, for each year, we compute the third quartile (75%) of the total number of publications and citations and select people having a number of publications and citations greater or equal to that value. This group is called *Best Performers*. The rest of the population is called *Mid-Low Performers*. Finally, we compute the yearly DI for both Best and Mid-low performers following the process described above along with the set cardinalities.

8.1.2 Data Description

In this section, we give an exhaustive description of the graphs and datasets obtained at the end of the processing stage of the pipelines shown in Figure 8.1.

Graphs description

From the Network Analysis pipeline (on the left in Figure 8.1), we generate two distinct co-authorship graphs representing the Worldwide and Italian Software Engineering communities, respectively. To construct these graphs, we collected data from Scopus and Google Scholar. Specifically, we compiled a list of Software Engineers by analyzing topic tags on Google Scholar, followed by gathering detailed academic data for these researchers through the Scopus API.

In the final graphs, for each node, we store the gender, the yearly publications and citations in the last century, the publication types (books, conference proceedings, journals), and co-authorship relationships. As gender data is not available from Scopus, we employ automatic tools of gender detection given the First Name. All personal information is then promptly anonymized via ID numbers.

The graph of Worldwide Software Engineers consists of 14661 edges and 3665 nodes, of which 3064 men and 601 women. The graph of Italian Software Engineers consists of 393 edges and 99 nodes, of which 78 are men and 21 are women.

Italian INF/SE dataset description

From the gender bias pipeline (on the right in Figure 8.1), we obtain two datasets: one consisting of Italian Software Engineers (SE) and another covering Italian professionals in the field of Informatics (INF). Both datasets are initially unsplit; however, for analysis purposes, they are later divided into academic performance quartiles. Our descriptions pertain to the data prior to this division.
These datasets are built combining data from Google Scholar, Scopus, and MIUR. From the topic tags available in Google Scholar we discern people in the field of Software Engineering from people in general Informatics. Then, from Scopus, we collect data pertaining to academic performance. In particular, for each record, similarly to the previously described graphs, we store the yearly publications and citations in the last century, the publication types (books, conference proceedings, journals), and co-authorship relationships. Additionally, for each record we store the gender and annual updates on academic roles (Researcher, Associate Professor, and Full Professor). These data is integrated from MIUR, thus avoiding reliance on automatic gender determination methods. Again, all personal information is promptly anonymized.
The SE dataset contains a total of 117 records, of which 99 men and only 18 women. As of 2022, 53 of them were researchers, 32 were Associate professors and 32 were Full professors. On the other hand, the bigger INF dataset contains 29.568 records. Of these, 19330 are men and 10238 are women. As of 2022, 14405 of these are researchers, 9672 are Associate professors and 5491 are Full Professors.

8.2 Experimental Evaluation

The experimental evaluation is organized as follows. In section 8.2.1, we compare the Italian SE community with the Worldwide SE community. We do this by analyzing their co-authorship networks regarding clusters and homophily. We then compare the mean bibliometrics performances of the Italian and Worldwide SE communities for each gender group. In section 8.2.2, we perform an analysis of gender bias in academic promotions by relying on the DI metric [99] for the Italian SE Community and the Italian Informatics Community. In particular, we conduct an in-depth analysis of gender bias in academic promotions by analysing different groups defined by the number of publications and the number of citations. The goal of this evaluation is to paint a complete picture of how the Italian SE Community fares in addressing gender gap against both the Worldwide SE Community and closely related subjects in Italy.

8.2.1 Analysis of Worldwide and Italian SE communities

In this section, we show and discuss the experiments performed in the worldwide and Italian Software Engineering communities. We start by showing the results of the Network Analysis carried out on the co-authorship network of Software Engineers. These experiments were performed on a dataset of 3665 Software Engineers, of which 3064 are men and 601 are women (for the worldwide community), and a dataset of 99 people, of which 78 are men and 21 are women (for the Italian community). The gender distribution is visualized in figure 8.3.

(A) Worldwide (B) Italy

FIGURE 8.3: Gender distribution of the Software Engineering communities, worldwide and in Italy.

Network Analysis

By inspecting the social graphs of Software Engineers, we can infer interesting differences between the co-authorship habits of women and men. In these graphs, each node is a researcher, and each edge is a co-authorship relation between them. In particular, in order to gather significant findings for the network behavior of the two groups, we inspect homophily, clustering coefficient, and modularity of both subgraphs of men and women. These metrics allow us to draw meaningful conclusions about the behaviors of the two groups when working together as co-authors.
Table 8.1 shows the homophily values according to two different metrics, where the Ideal Value means that there is no homophily. If we were to only consider the ratio of same-gender edges over all edges (which is $1 - Homophily$, 0.729) we would conclude that the number of same-gender edges is significantly higher. However, it is crucial to remember that this value does not take into account the proportion of the gender groups inside the graphs. As shown in figure 8.3, this is significant to us, as the imbalance between groups is severe. Consequently, we need to compare Homophily with the ideal value $2pq$ (refer to section 8.1) to account for this imbalance. As shown in table 8.1, the Observed Homophily values for both Europe and Italy are very close to their ideal value, with Italy being slightly less balanced. However, this small variation might be due to a lower sample size. According to the Coleman Homophily, the Observed Value is very close to the Ideal Value of no homophily for both communities. We can thus infer that in both cases the SE community exhibits no Homophily in co-authorship relations, meaning that gender does not seem to be an important criterion for researchers when selecting co-authors for their work.

	Europe		*Italy*	
Metric	**Observed**	**Ideal**	**Observed**	**Ideal**
Homophily	0.271	0.274	0.41	0.35
Coleman Homophily	0.099	0	-0.011	0

TABLE 8.1: Homophily values computed via different metrics.

We then turn our attention to the Clustering Coefficient [231] of the different genders in the graph. Figure 8.4 presents boxplots illustrating results for the subgraphs for both men and women. Since this metric is influenced by the size of the groups, we computed the Clustering Coefficient also on a subgraph of men that has been sampled to match the number of nodes in the subgraph of women. To account for

statistical variation in the sampled subgraph, we computed the mean of the clustering coefficient values for each node over 10 different samplings.

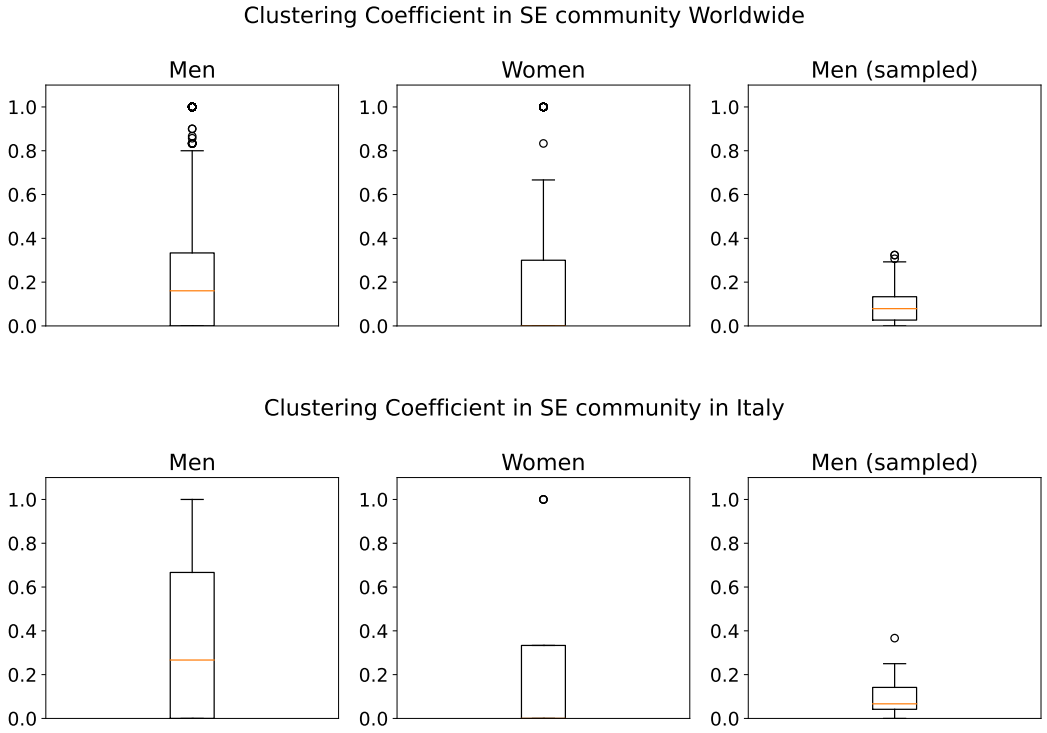


FIGURE 8.4: Clustering coefficient of the different gender groups in the Worldwide Community (upper side) and the Italian community (lower side).

These results show that the subgraph of men exhibits a higher Clustering Coefficient with respect to the subgraph of women. While this may be partially due to the imbalance of groups, the median of the sampled subgraphs of men is still higher than the subgraph of women. This holds for both the Worldwide and Italian SE communities. We can thus infer that men generally work in tightly-knit groups more often than women. The Clustering Coefficients of men and women were also tested via the Wilcoxon-Mann-Whitney test [97], which resulted in a p-value of 7.77×10^{-6}. This proves a statistically significant difference between the two groups with a 95% confidence interval. To corroborate these findings, we investigate the Modularity [193] of the men and women subgraphs.

We computed the normalized modularity for both subgraphs. The Modularity values are 3.5×10^{-4} and 3.3×10^{-3} for men and women in the worldwide SE community, respectively. These values are both very low and hint at the fact that neither group exhibits a strong community structure. However, the modularity of the subgraph of women is much higher in comparison. We can infer that the global structure of the subgraph of women is more pronounced and clustered with respect to the one of the men, indicating that women build global relationships outside of their working group more often than men. For Italy, the Modularity values are 0.01 for men and 0.04 for women. While these values are generally higher, the same line of reasoning can be applied to the Italian community.

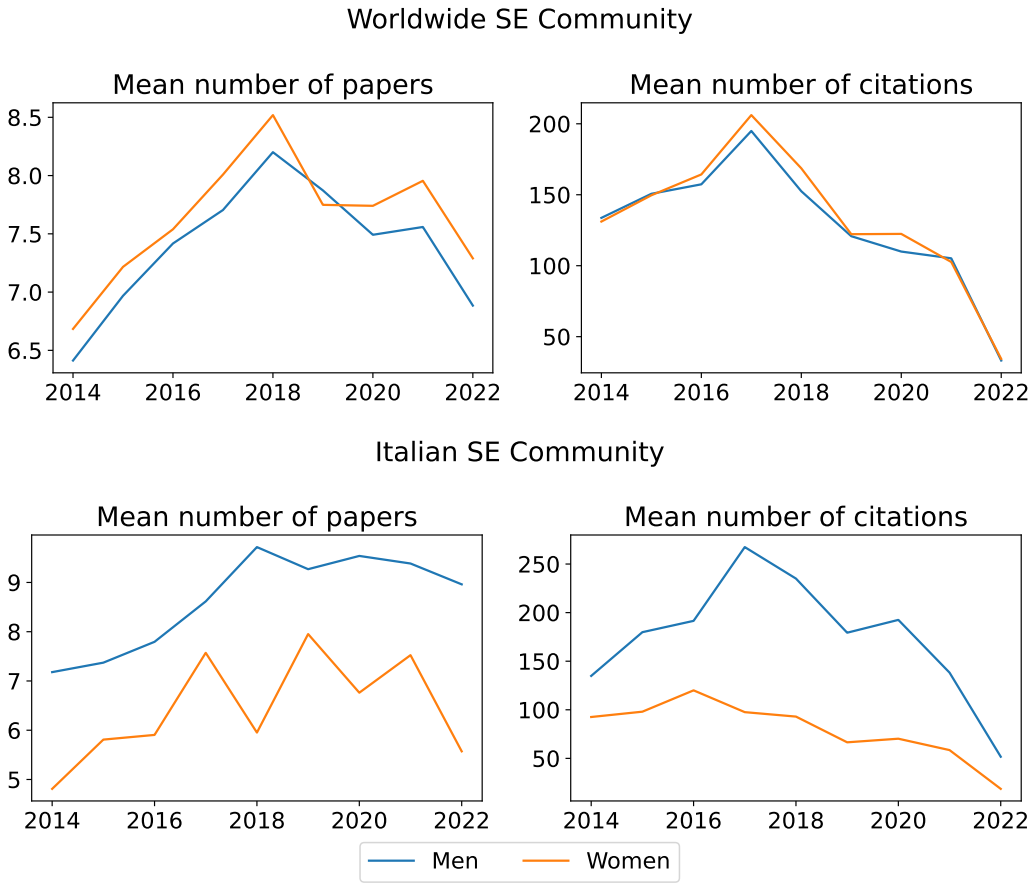


FIGURE 8.5: Mean publications and citations in the Worldwide (upper side) and Italian (lower side) SE communities.

Metrics for academic performance

Figure 8.5 reports the mean number of publications and citations for each gender group both worldwide (upper side) and in Italy (lower side). For the former, the two gender groups have comparable academic productivity, although slightly favorable to women, who generally publish more papers and get cited more often. On the other hand, there is a stark contrast in academic productivity between the two gender groups in Italy. Women appear to publish less and get cited less than men. While part of this might be due to lower sample size and therefore higher statistical variation, the trend remains worrying and should be the subject of future research.
Lower metrics for the year 2022 are due to the data being incomplete for that year, but it is interesting to note that the mean number of papers and citations, in general, seem to be on a downtrend with respect to the peak observed in the years 2018 to 2020.

Discussion

The network analysis conducted on the co-authorship graphs pertaining to both the global Software Engineering (SE) Community and its representation in Italy conveys remarkably analogous results. Women make up, in both cases, around 20% of the SE Community (figure 8.3), and when taking this percentage into account, the networks show no evidence of homophily (table 8.1), e.g., preference of choosing co-authors within the same gender group. In general, however, the subgraph of men exhibits

a higher Clustering Coefficient (figure 8.4) and lower Modularity, which means that they tend to form smaller, more tightly-knit working groups with respect to women, but generally entertain less co-authorship relationships with researchers outside of their cluster. We can draw similar conclusions for both communities since the analyzed metrics are comparable and coherent between the worldwide and Italian SE communities.

The academic performance metrics of figure 8.5 show how academic performance is generally comparable between the two gender groups worldwide, but is severely different in the Italian SE Community. This is a worrying trend that needs to be further investigated.

Answer to RQ1

While there is no evidence of researchers deliberately selecting colleagues of the same gender as co-authors, it is observed that men tend to work in more closely-knit clusters and have fewer connections outside of their respective clusters when compared to women. These results apply to both the Worldwide and the Italian SE (Software Engineering) communities.

8.2.2 Analysis of Gender Bias in the SE and Italian informatics Communities

In this section, we present the comparison between the Italian SE community and the general Italian Informatics Community. We focus on the gender bias occurring in the events of *academic promotion*. In particular, we consider the promotions from Researcher to Associate Professor and from Associate Professor to Full Professor. The analysis is carried out on the four datasets of Informatics Researchers/Associate Professors (INF_{RA}), Informatics Associate/Full Professors (INF_{AF}), SE Researchers/Associate Professors (SE_{RA}), and SE Associate/Full Professors (SE_{AF}).

Disparate Impact Evaluation

Figure 8.6 shows the DI (left y-axis) and set cardinalities (right y-axis) for each of the datasets above (INF_{RA}, INF_{AF}, SE_{RA}, and SE_{AF}) on a yearly basis in the reference period (2018-2022). In the figure, the charts on the left side show results for the Informatics (INF) Community datasets (INF_{RA}, INF_{AF}), while the ones on the right side show results for the Software Engineering (SE) Community (SE_{RA}, SE_{AF}).

Concerning the full set cardinalities (i.e., of both men and women), they exhibit the same trend across all datasets. Since we only consider people who were in the Italian academic system for the entire reference period, we do not consider Researchers who were acquired later than 2018, so their cardinality is bound to decrease. The number of Full professors is rising in both the INF and SE communities, but the increase in the SE community is significantly larger. In 2022, there are more Full professors than Associate professors specifically in the SE subset. This suggests that promotions to Full professorship are occurring at a higher rate among professors in the field of SE compared to the INF community.

Concerning the gender bias in promotions to Associate Professor (INF_{RA} and SE_{RA} in the figure), in both the Informatics and Software Engineering communities the trend of DI appears to be on an upward trajectory. However, the SE community seems to suffer from a higher bias than the overall INF community. The DI for the SE community starts from a value of 0.75 in 2018 to a value of 0.8 in 2022. In contrast,

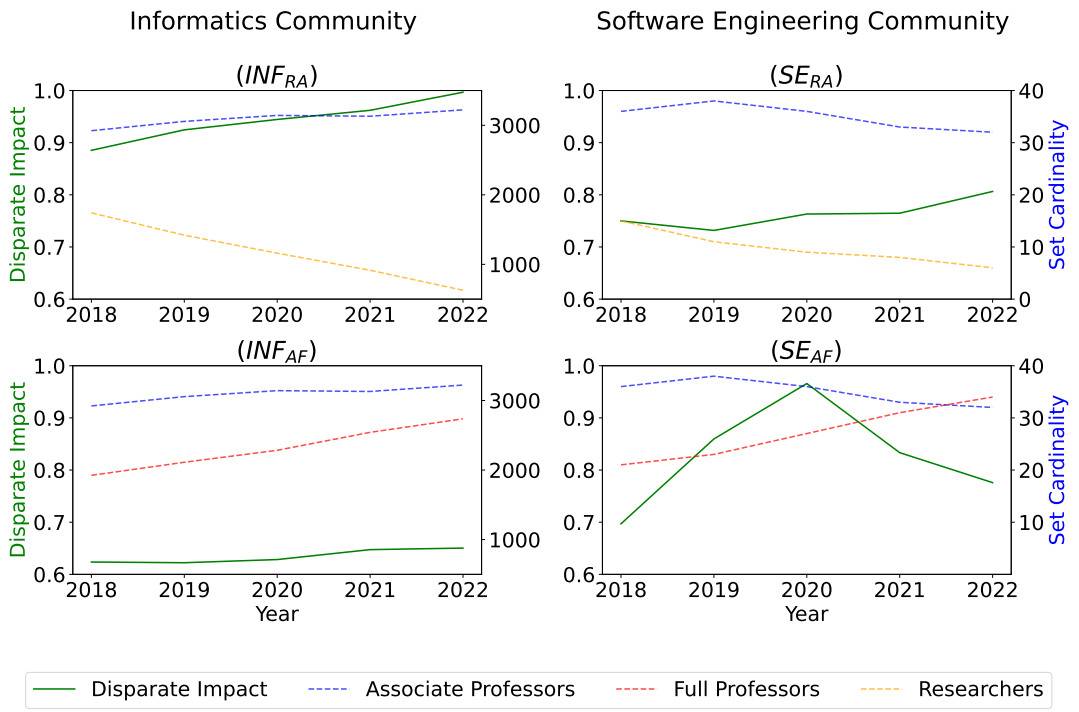


FIGURE 8.6: Year-by-year Disparate Impact and Set Cardinality for the Informatics Community (left column) and Software Engineering Community (right column).

the DI of the INF community starts from a value of 0.9 in 2018 to a value of almost 1 in 2022, meaning a nearly complete absence of bias in academic promotions. In general, we observe how the amount of bias in the SE community is about 20% higher than in the overall INF community.

In contrast, concerning bias in promotions to Full Professors (INF_{AF} and SE_{AF} in the figure), the SE community exhibits a much lower bias concerning the INF community. DI for the SE community starts from 0.7 in 2018, then reaches a peak of 0.95 in 2020, to a final value of almost 0.8 in 2022. This downtrend from 2020 to 2022 can be partially explained by the small set cardinality, which makes the DI more sensitive to small changes (i.e., additions or deletions) in the groups. Instead, the DI for the overall INF community presents a slight increase over the period, starting from a value of 0.63 in 2018 to a value of 0.65 in 2022. In this case, the amount of bias in the INF community ranges from 15 to 35% greater than in the SE community throughout the observed period.

Answer to RQ2

The Italian SE community presents a higher gender bias in promotions from Researchers to Associate professors compared to the overall Italian informatics community. Concerning promotions from Associate to Full professors, the SE community presents a lower gender bias compared to the full informatics community.

Disparate Impact Among Best performers and Mid-Low Performers

We now shift our attention to the Disparate Impact exhibited by two population subsubgroups: people having a number of publications and citations greater or equal

to 75% of the entire population, and the rest of the group. We will refer to them as Best Performers and Mid-Low Performers, respectively. It is worth noticing how, as already mentioned in Section 8.1, people not having a Scopus ID have been excluded from this analysis, since we are not able to extract information about publications and citations for them.

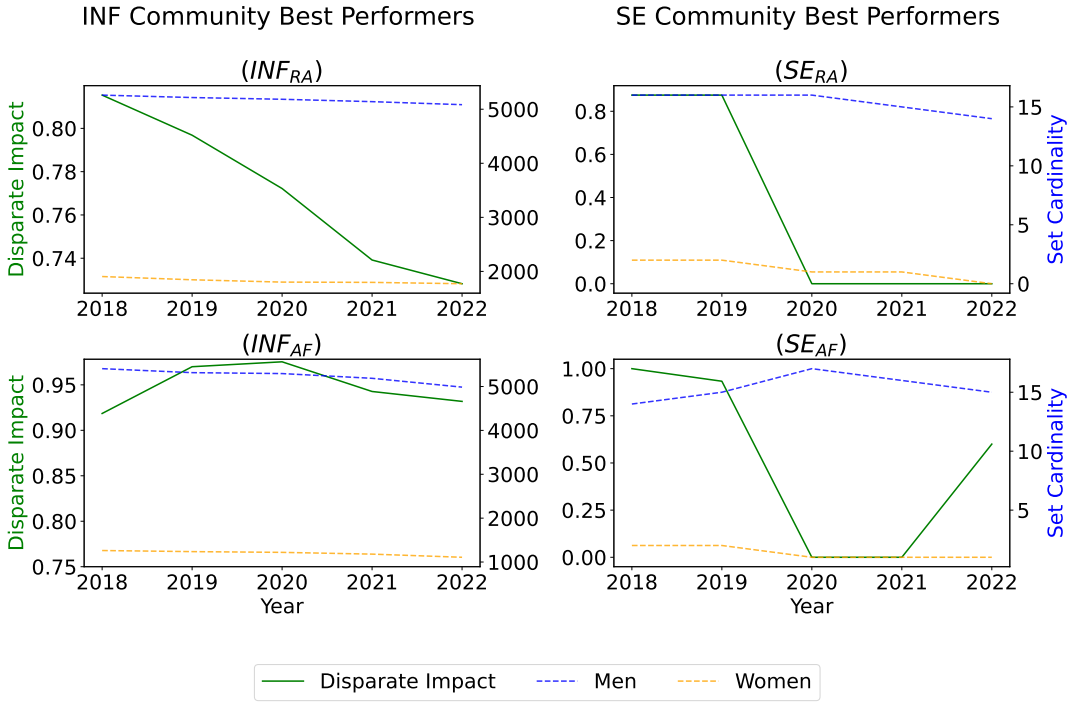


FIGURE 8.7: Year-by-year Disparate Impact and Set Cardinality for the Informatics Community (left column) and Software Engineering Community (right column) for the Best Performers.

Figure 8.7 shows the DI (left y-axis) and set cardinalities (right y-axis) of Italian SE and informatics best performers. Is worth noticing that, in this analysis, the set cardinalities represent the total number of men and women in the specific group. In the Informatics Community, the DI when considering promotions from Researchers to Associate professors is on a downtrend, while the DI when considering promotions from Associate to Full professors is on the rise, despite a small hiccup in the most recent years. This is true despite the ratio between men and women set cardinalities remaining almost constant. This means that the probability of best performers male Researchers becoming Associate professors has increased over the years compared to the probability of female best performers Researchers. The opposite holds for Associate versus Full professors. However, it is also worth noticing that the DI concerning the promotions from Researcher to Associate professor is still very high (around 0.7 for the year 2022), meaning that, even if in a downtrend, the gender bias in such promotions is not very high. Concerning the SE community, the DI for the best performers is subject to extreme variation due to the very low number of women in this subset (with a maximum of 2 women in the comparison between Researchers and Associate professors and 1 woman in the comparison between Associate and Full professors). For certain years, the number of women with the positive label (i.e., Associate professors in case of the comparison with Researchers and Full professor in case of the comparison with Associate professors) is zero, causing the DI to be impossible to compute (we default to 0 in this case, the most biased outcome).

We cannot draw meaningful conclusions in this case, but the number of women best performers in SE remains worrying and subject to future research.

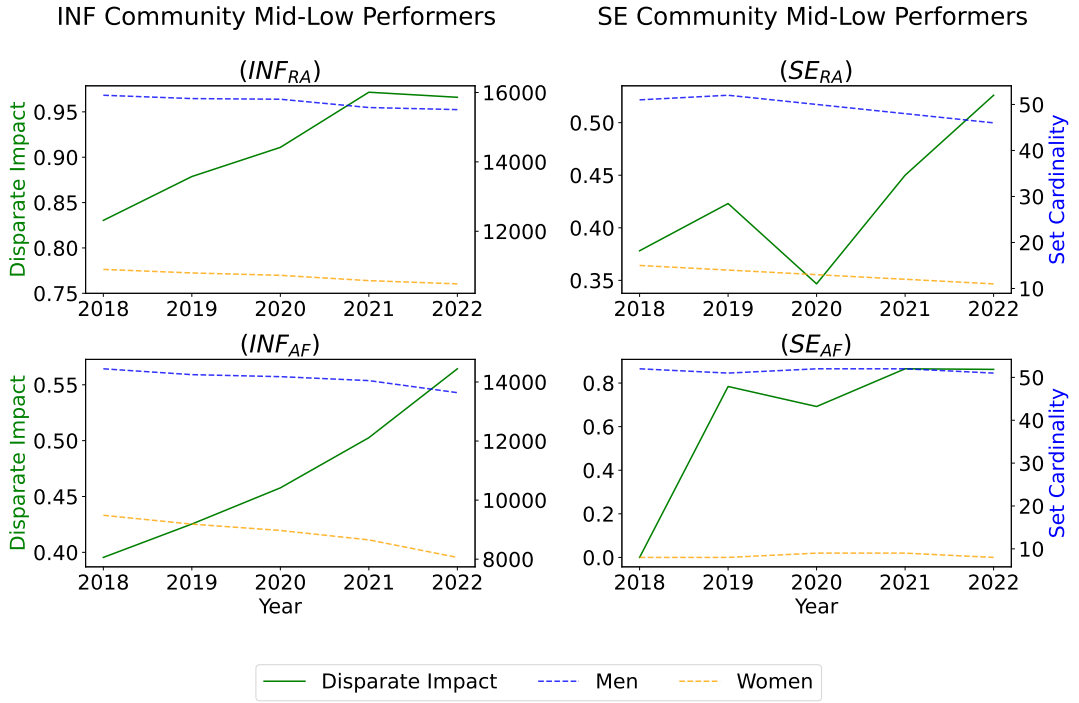


FIGURE 8.8: Year-by-year Disparate Impact and Set Cardinality for the Informatics Community (left column) and Software Engineering Community (right column) for the Mid-Low Performers.

Figure 8.8 shows the Disparate Impact for Mid-Low Performers. In contrast with the best performers, this time the DI for both promotion scenarios in the informatics community is on an uptrend. However, while INF_{RA} reaches almost perfect fairness (DI = 1) in 2021, the level of DI for the promotions from Associate to Full professors is still low, with a maximum value of around 0.6 in 2022. From this scenario, we can conclude how, while there is almost perfect gender fairness in the promotions from Researchers to Associate professors, males still have a higher probability than women of becoming Full professors. On the other hand, the DI for the SE group, while still subject to heavier variation (given by the smaller dimension of the groups), is more stable with respect to the Best Performers. In particular, we observe an opposite trend with respect to the informatics community. In fact, the DI for the promotions from Researchers to Associate professors is low (with a value of around 0.5 in 2022). This means that males have a higher probability than women of becoming Associate professors. On the contrary, we observe an uptrend in gender fairness concerning promotions from Associate to Full professors (with a value of DI of around 0.75 in 2022). Finally, we note from the data how until 2019 there were no women Full professors with a mid-low number of publications and citations in the SE community.

Answer to RQ3

Concerning best performers, we observe a low gender bias in both promotions to Associate and Full professors in the Italian informatics community. Instead, the low number of women best performers in the SE Italian community does not allow us to draw meaningful conclusions about gender bias in this context. Concerning mid-low performers, we observe how there is still a gender bias in promotions to Full professors in the Italian informatics community, while we detect a low bias in promotions to Associate professors. Mid-low performers of the SE community follow an opposite trend since we observe higher gender bias in promotions to Associate professors, while there is gender fairness in promotions to Full professors.

8.3 Threats to Validity

This section discusses possible threats that can hamper the results of the performed evaluation.

Conclusion Validity concerns issues about the results drawn from our evaluation. In this context, there may be other variables that can influence academic promotions, like the number of years in academia. To address this issue, we fix our analysis to a specific range of years and consider only people being in the academic context for the whole considered range. Moreover, we plan to extend our analysis considering also other relevant features like participation in the organising and steering committee and in the editorial board of the most relevant conferences and journals on SE.

Internal Validity concerns internal factors that can impact the results of our evaluation. Firstly, we want to acknowledge that we are considering gender as a binary variable in our analysis despite being aware that some individuals may not identify with this classification. However, the data sources from which we have extracted gender information present it as a binary variable, limiting our ability to perform a more detailed gender analysis. Secondly, we have chosen to restrict our analysis of the Italian SE and informatics communities to individuals within the Italian academic system for all the year ranges assessed. This decision was made to ensure that our analysis of gender bias in academic promotions is accurate and not influenced by individuals leaving or joining the academic system. Lastly, we have used Google Scholar tags to identify people belonging to the SE community. While we recognize that Google Scholar tags may not be completely accurate and may not encompass all individuals in the SE community, we were unable to find more reliable sources for this information.

Construct Validity concerns how the performed experiment is adequate to address our research questions. To address this threat, we employ state of the art network analysis techniques and a formal bias metric to analyse the gender bias in academic promotions. However, we publicly release the datasets used in our experiments for future experiments and analyses.

External Validity concerns the generalizability of our results. Our analysis is focused on the Italian academic system and is not applicable to countries having a different academic system.

8.4 Conclusion and Future Work

In this work, we presented an analysis of gender gap within the Italian Software Engineering and informatics communities. We started by performing a review of how the literature analysed the subject of gender gap in academia and in the SE community. From this review, we have shown how there is growing interest in analysing the issue of gender gap both in SE and academia. However, no papers study the issue of gender gap from a perspective of academic promotions in specific academic areas, like informatics and SE. Upon inspecting the literature, we proposed to fill the unexplored area pertaining to gender gap in the Italian Software Engineering community. We started by presenting a data collection, filtering, and processing pipeline to gather and refine academic data from multiple Italian and international public sources. We also made these datasets available online for public access. Next, we performed two sets of evaluations: *i)* we compared the Italian SE community with the Worldwide SE community via Network Analysis techniques and metrics, *ii)* we compared the Italian SE community with the Italian Informatics community to infer differences in the pattern of academic promotions for the different gender groups considering also differences between best performers and mid-low performers in terms of publications and citations. For the former, we infer that, despite the lack of exhibited homophily, men tend to cluster together in more tightly-knit groups and build fewer global connections with respect to women. We also conclude that the networks do not exhibit homophily, meaning that gender is not a primary factor for researchers when selecting co-authors. For the latter, we found that the Italian informatics community presents a higher gender bias in promotions to Full professors compared with the SE community. On the contrary, the SE community presents a higher bias in promotions to Associate professors. Finally, concerning the number of publications and citations, we observed a low gender bias in both promotions to Associate and Full professors for best performers in the informatics community, while the low number of women best performers in the SE community does not allow us to draw meaningful conclusions. Concerning mid-low performers, we observed a high gender bias in promotions to Full professors in the Italian informatics community, while the SE community presents a higher bias in promotions to Associate professors.

Concerning future works, an avenue of future research would be expanding the analysis on multiple countries, taking into account specific rules and regulations. Next, the Network Analysis study could be expanded upon by considering a weighted graph of co-authorship relations, which at the time of writing, were not available via the Scopus API. The Disparate Impact computation could be expanded as well by considering other factors that are relevant to academic promotions, like participation in the organization of top-tier SE conferences. Finally, while analyzing promotion patterns is important, future efforts might involve studying the degree of gender bias in the process of initial recruitment as well.

Chapter 9

Debiaser for Multiple Variables

9.1 Introduction

Over the years, many recommendation systems have been proposed that rely on multi-class classification systems. For instance, consider the one proposed by [18] to recommend admissions to a graduate school, the one proposed by [267] to improve the learning experience, the one proposed by [236] to recommend fertilizers, the one proposed by [179] for crop recommendation, or the one proposed by [275] to recommend physical therapy. These systems embed a multi-class classifier to solve multi-class classification tasks. Assuring *fairness* of these systems is also essential to achieving some of the aforementioned sustainable goals: i.e., goal 2 (zero huger) [236, 179], goal 3 (good health and well-being) [275], and goal 4 (quality education) [18, 267]. In [234] the authors even highlight how integrating multi-class classification into context-aware recommendation systems can improve the overall recommendation. Different methods have been proposed to mitigate bias at several levels of data processing for both classification and recommender systems [180, 39]. In general, ensuring fairness for all kinds of tasks, including multi-class classification, is a considerable step towards building **trustworthy** AI systems. We tackle the challenge of de-biasing datasets, to build fairer multiclass classifiers, in this Section. Specifically, to conduct our research and to better address the problem of bias mitigation in multi-class classification, we formulate the following four research questions (RQ) that will help to highlight the fundamental findings and novel contributions of this paper.

RQ1. What are the strengths and limitations of current existing approaches addressing bias mitigation in multi-class classification problems?

In this paper, we analyze three baselines designed to mitigate bias in multi-class classification problems, namely *Exponentiated Gradient* and *Grid Search* methods from [3], and the *Blackbox* method from [201]. To the best of our knowledge, these are the only ones implemented for the multi-class classification task. To highlight their strengths and weaknesses, we apply each method to a heterogeneous set of binary and multi-class datasets that are widely used in research (extensively discussed in section 9.3.2).

RQ2. How can we design a novel approach that goes beyond the existing baselines?

To overcome some of the limitations of the analyzed baseline, we present an improved version of *Debiaser for Multiple Variables (DEMV)* presented in [68]. DEMV is a generalization of the *sampling* algorithm proposed by [133]. DEMV is model- and data-agnostic, allowing its introduction in already existing systems without particular effort and without introducing structural changes.

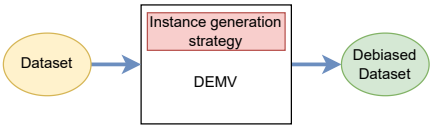


FIGURE 9.1: Application of DEMV

DEMV is the first proposed pre-processing method to mitigate bias caused by an unequal distribution of instances in the population (i.e. *unbalanced groups* bias) in an agnostic way in both binary and multi-class classification considering multiple sensitive variables. As highlighted in figure 9.1, DEMV takes as input a generic dataset and returns in output the debiased dataset without considering the classifier involved in the task. We implement DEMV with a plug-in approach where the user can select different *Instance generation strategies*. The source code is provided on GitHub and on the Territori Aperti RI.

RQ3. How can DEMV keep a high level of accuracy while improving fairness?

Since DEMV is a pre-processing algorithm, the fairness and accuracy trade-off can be managed by better manipulating the instances of the dataset. Since our approach tackles the *unbalanced groups* bias, this means generating new instances that are coherent with the existing ones in terms of values and distribution. We plug-in in DEMV three different generating strategies, namely *Uniform*[1], *SMOTE* [42] and *ADASYN* [122]. In order to evaluate the influence of each strategy on DEMV's ability to enhance the fairness and accuracy of the classifier, we extensively evaluate DEMV by employing nine datasets extensively used in literature (see section 9.3.2). We perform this analysis both in binary and multi-class settings.

RQ4. In which conditions does DEMV goes beyond the existing baselines?

To answer this question, we run a set of experiments whose aim is to evaluate the performance of DEMV in improving fairness while keeping a high level of accuracy. In particular, we evaluate it in binary and multi-class classification problems and consider sensitive groups identified by up to three sensitive variables. To demonstrate the wide validity of DEMV, we employ in the experiments an heterogeneous set of ML classifiers, namely Logistic Regression, Multi-Layer Perceptrons, Gradient Boosting Classifier and SVC. As we show in section 9.3.4, DEMV outperforms the baselines in the binary task in the case of more than two sensitive variables, while it remains competitive with one or two sensitive variables. In the case of the multi-class task, DEMV outperforms the baselines in every setting (i.e., number of sensitive variables) for all the considered datasets. Finally, DEMV improves the fairness of every of the analyzed classification methods without affecting their behavior and keeping a considerable high level of accuracy.

Note that in this paper, we extend our previous work presented in [68] with the following main contributions:

1. We analyze strengths and weaknesses of currently established methods for bias mitigation that can be applied to both binary and multi-class classification with multiple sensitive variables;

[1]Uniform strategy replicates the instances of the dataset with a uniform probability distribution.

2. We revise the original algorithm by generalizing the functions for generating and removing instances during the process. In particular, we rewrite the algorithm as a plug-in approach in which different functions for the creation and deletion of instances can be called;

3. We evaluate the impact of different *instances generating strategies* on the accuracy and fairness achieved by DEMV;

4. We extensively evaluate DEMV considering several settings:

   - employing a larger set of binary and multi-class datasets. The selected datasets are heterogeneous both in terms of data scope and size, enlarging the generality of our analysis;
   - considering a larger set of established baselines able to handle bias in binary and multi-class classification with any number of sensitive variables. We highlight the strengths and weaknesses of the baselines and show how DEMV is able to overcome them in all settings when the multi-class classification task is considered, and in the case of more than two sensitive variables, in case of binary classification task;
   - executing a deeper analysis that employs a different number of sensitive variables. In particular, we consider sensitive groups made by one, two, and three sensitive variables. We highlight how the number of sensitive variables (badly) influences the ability of baselines to improve fairness while DEMV performs consistently;
   - employing a more extensive set of classifiers and analyzing the impact of DEMV on their behavior. In particular, we consider the following categories of classification methods: Linear, Boosting, Support Vector Machines, and Neural Networks.

This Chapter is structured as follows: in Section 9.2, we describe in detail the proposed approach; Section 9.3 is dedicated to the experimental analysis that has been conducted to evaluate the best-generating strategy and to compare DEMV both in binary and multi-classification problems while discussing its current strengths and weaknesses, also, a description on how to reproduce the experiments is provided; in section 9.4 we discuss the results, and we answer to the four research questions; finally, Section 9.5 reports possible points of improvements of DEMV and concludes the paper.

9.2 Debiaser for Multiple Variables (DEMV)

In this section, we describe in detail the *Debiaser for Multiple Variables (DEMV)* approach, a pre-processing bias mitigation method for multiple sensitive variables in the classification context.
The main idea behind the proposed method is that to enhance effectively the classifier's fairness during pre-processing is necessary to consider all possible combinations of the values of the sensitive variables and the label's values for the definition of the so-called *sensitive groups*. Under the definition of bias considered in this paper (i.e., *unbalanced groups bias*), if a dataset is biased, we observe that the size of the sensitive group identified by the privileged value of the sensitive variable (e.g., *men*) and the positive label (e.g., *high income*) should be larger than expected. In comparison, the size of the sensitive group identified by the unprivileged value of

the sensitive variable (e.g., *women*) and the positive label (e.g., *high income*) should be smaller than expected. In the same way, the size of the sensitive group identified by the unprivileged value of the sensitive variable and the negative label should be larger than expected, and the group size determined by the positive value of the sensitive variable and the negative label should be smaller than expected. For this reason, to enhance the fairness of the classifier, we have to perfectly balance the size of these groups by adding or removing items to remove disparity.

We approach the problem by recursively identifying all the possible groups given by combining all the values of the sensible variables with the belonging label (class). Next, for each group, we compute its expected (W_{exp}) and observed (W_{obs}) sizes[2] and look at the ratio among these two values. If $W_{exp} \backslash W_{obs} = 1$, it implies that the group is fully balanced. Otherwise, if the ratio is less than one, the group size is larger than expected, so we must remove an element from the considered group accordingly to a chosen deletion strategy. Finally, if the ratio is greater than one, the group is smaller than expected, so we have to add another item accordingly to a generation strategy. For each group, we recursively repeat this balancing operation until $W_{exp} \backslash W_{obs}$ converge to one. It is worth noting that, in order to keep a high level of accuracy, the new items added to a group should be coherent in their values and distribution with the already existing ones.

Hence, DEMV can be defined as an algorithm made of two separate procedures: identification of the sensitive groups and balancing of them. In the following, we first illustrate the procedure used to identify the sensitive groups; then, we describe the balancing step.

The full implementation of DEMV, comprising of all the code to reproduce the experiments, is available at the Territori Aperti RI[3] and on GitHub[4] as well (see also section 9.3.5).

9.2.1 Sensitive Groups Identification

The identification and management of the sensitive groups are performed by the *DEMV* recursive function, whose pseudo-code is shown in listing Algorithm 2[5].

The main scope of this function is to identify and manage all the possible sensitive groups of a dataset. To this aim, this function takes as input the dataset D, the categorical sensitive variables $S_1, \ldots, S_n$, the label L and other parameters useful for the recursion: a counter i initially set to 0 (used to count the number of explored sensitive variables), an array G initially empty (used to collect the balanced, sensitive groups), and a boolean *condition* initially set to *true* (used to define the condition needed to identify the different sensitive groups). Lines from 2 to 11 define the base condition of the function. This condition checks if all the sensitive variables needed to identify a sensitive group have been explored (i.e., the counter i equals the number of sensitive variables). If so, the algorithm iterates the possible values of the label and creates, for each of them, a corresponding sensitive group g is defined as $\{X \in D | S_1 == s_1 \wedge S_2 == s_2 \wedge \cdots \wedge S_n == s_n \wedge L == l\}$, where $s_1, \ldots, s_n$ are possible values of the sensitive variables and l is a value of the label.

[2] the formal definition of these values is given in section 9.2.1

[3] https://bit.ly/3scwtaB

[4] `https://github.com/giordanoDaloisio/demv2022`

[5] We recall that a recursive function is generally made of three main sections: a *base condition*, which defines the main return statement of the function, a *recursion point* in which the function calls itself, and an *end condition* representing the end of the process.

Algorithm 2 Pseudo-code of DEMV

Input: Dataset D, Sensitive variables $S_1, S_2, \ldots, S_n$, Label L, $i = 0$, $G = []$, condition $= true$
Output: Sampled dataset D_S

1: $n = \text{length}(\{S_1, S_2, \ldots, S_n\})$
{Base condition: check if all sensitive variables have been explored}
2: **if** $i == n$ **then**
3: **for all** $l \in L$ **do**
4: $g = \{X \in D \mid \text{condition} \wedge L == l\}$
5: $W_{exp} = \frac{|\{X \in D | \text{condition}\}|}{|D|} \cdot \frac{|\{X \in D | L == l\}|}{|D|}$
6: $W_{obs} = \frac{|g|}{|D|}$
7: $g_b = \text{BALANCE}(g, W_{exp}, W_{obs})$
8: add g_b to G
9: **end for**
10: **return** G
11: **end if**
{Recursion point: select a new sensitive variable}
12: $i = i + 1$
13: **for all** $s \in S_i$ **do**
14: $G' = \text{DEMV}(D, S_1, \ldots, S_n, i, G, \text{ condition} = \text{condition} \wedge S_i == s)$
15: add G' to G
16: **end for**
{End condition: check if all groups explored}
17: **if** $length(G) == |L| \cdot (\prod_{i=1}^{n} |S_i|)$ **then**
18: $D_S =$ merge all $g \in G$
19: **return** D_S
20: **else**
21: **return** G
22: **end if**

Then, for each group, the algorithm computes expected and observed sizes. These two values are defined respectively as:

$$W_{exp} = \frac{|\{X \in D | S = s\}|}{|D|} * \frac{|\{X \in D | L = l\}|}{|D|} \tag{9.1}$$

$$W_{obs} = \frac{|\{X \in D | S = s \wedge L = l\}|}{|D|} \tag{9.2}$$

where $S = s$ is a generic condition on the value of the sensitive variables[6] (i.e., *condition* variable in algorithm 2) and $L = l$ is a condition on the label's value. It is worth noting that $|\{X \in D | S = s \wedge L = l\}|$ is equal to the size of the sensitive group g identified by the conditions $S = s$ and $L = l$.
Next, the algorithm balances the group by invoking the BALANCE function (listing Algorithm 3). This function implements the balancing strategies described in section 9.2.2. Finally, the approach adds the balanced group g_b to the array G (used to collect all the balanced groups) and returns it.
Lines from 12 to 16 identify the recursion point of the function. The purpose of the

[6]The variables can be binary, discrete or categorical ones

recursion is to build the condition needed to determine the sensitive groups dynamically. In particular, if the algorithm has not explored all the sensitive variables (i.e., the value of i is not equal to the number of sensitive variables), the algorithm starts exploring a new one (variable S_i in the code). The exploration is done by iterating all the possible values of the current sensitive variable S_i. Each value of the sensitive variable corresponds to a new sensitive group that must be identified and balanced. Each identified sensitive group is collected inside a temporary set G'. The returning set of sensitive groups G' partially identified by the given sub-condition is then merged with the given set of sensitive groups G.
Finally, lines from 17 to 22 define the end condition of the function. In particular, the total number of sensitive groups obtainable from a dataset with n sensitive variables and a label L is equal to the product of all the possible values of the sensitive variable and the values of the label, that is:

$$|L| * \left(\prod_{i=1}^{n} |S_i| \right)$$

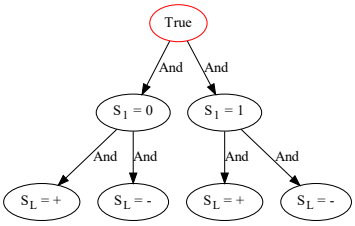
If the length of G is equal to this value, then the function has considered and balanced all the groups and returns the final sampled dataset D_S. Otherwise, the function, being in the middle of the recursion returns G, will be again merged with the result of the previous recursive calls. This procedure shown in Algorithm 2 can also be applied to binary classification problems; in that case, the number of sensitive groups will be equal to

$$2 * \left(\prod_{i=1}^{n} |S_i| \right)$$

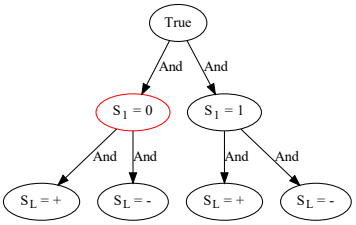
We like to note that, even if the number of sensitive groups grows exponentially with respect to the number of sensitive variables, this number is still manageable in the real-world case scenario where a small amount of sensitive variables are typically considered (e.g., solely 32 groups are present in a multi-class classification task where four classes and three binary sensitive variables are considered).
To better clarify the behavior of DEMV, figure 9.2 shows an example execution of the first steps of the algorithm on a dataset with one binary sensitive variable and a binary label. For the sake of simplicity in this example we are using binary variables, but as highlighted in section 9.3, DEMV can also be applied to multi-class labels and categorical sensitive variables. Figure 9.2a represents step 0 of the algorithm, in which the counter is set to 0, and the condition is set to *true*. Next, the algorithm starts a depth-first exploration of the depicted tree. In figure 9.2b, the algorithm adds the condition $S_1 == 0$ to the initial *true* condition, and in figure 9.2c the condition $S_L == +$ is also added. Figure 9.2d depicts the identification of the first sensitive group defined as $\{X \in D | S_1 == 0 \wedge S_L == +\}$. Finally, figures 9.2e and 9.2f show the identification of the second sensitive group, this time defined as $\{X \in D | S_1 == 0 \wedge S_L == -\}$. The algorithm then proceeds to balance the other sensitive groups. When all the groups have been balanced, they are merged to return a fully balanced dataset.

9.2.2 Balancing Strategies

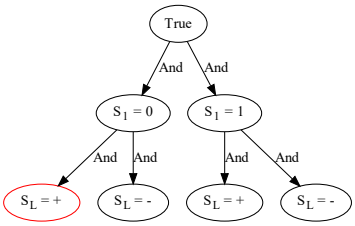
The group-balancing operation is implemented by the BALANCE function, whose pseudo-code is depicted in listing Algorithm 3. This function takes as input the group g and the expected (W_{exp}) and observed size (W_{obs}). The core of this algorithm



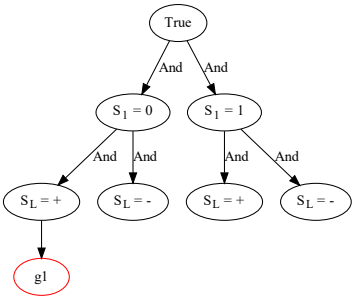
(A) Step 0: $i = 0$ and $condition = true$



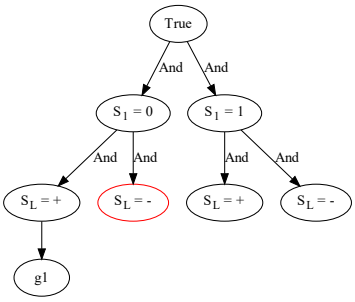
(B) Step 1: $i = 1$ and $condition = (true \wedge S_1 == 0)$



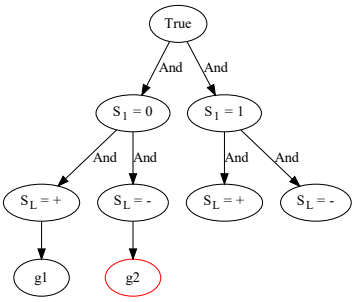
(C) Step 2: $i = 1$ and $condition = (true \wedge S_1 == 0 \wedge S_L == +)$



(D) Step 3: first sensitive group identified



(E) Step 4: $i = 1$ and $condition = (true \wedge S_1 == 0 \wedge S_L == -)$



(F) Step 5: second sensitive group identified

FIGURE 9.2: Example execution of the first steps of DEMV algorithm

is a loop that checks if the value of $W_{exp} \backslash W_{obs}$ is different from 1. If the ratio is < 1, then it means that the size of the group is higher than expected. In this case the algorithm selects an index in the range of $(0, size(g) - 1)$ accordingly to the deletion strategy REMOVE to remove the corresponding item from the group. Otherwise, if the ratio is > 1, then the size of the observed group is lower than expected. In this case, the algorithm generates a new sample by using the generative strategy GENERATE, then it adds the new generated sensitive group. Finally, the algorithm returns the balanced group when the while condition becomes true.

To better understand the overall process we need to also discuss the two underlying removal and generative strategies. The simplest of the two strategies is the removal one, where the removal candidate must be selected among the samples already present in the group. The removal strategy implemented in REMOVE function is typically based on a sampling function that follows a given distribution (e.g.

Algorithm 3 Pseudo-code of BALANCE

Input: Group g, expected size W_{exp}, observed size W_{obs}
Output: Balanced group g

The group is not yet balanced
while $W_{exp}/W_{obs} \neq 1$ **do**
  if $W_{exp}/W_{obs} < 1$ **then**
    Group is larger than expected, remove an item
    $i \leftarrow \text{REMOVE}(0, \ldots, \text{size}(g) - 1)$
    remove item i from g
  else if $W_{exp}/W_{obs} > 1$ **then**
    Group is smaller than expected, add an item
    $i \leftarrow \text{GENERATE}()$
    add item i to g
  end if
  recompute W_{obs}
end while
return g

uniform).
Conversely, the generative strategy implemented in the GENERATE function might be the most tricky since it is responsible for providing new samples used by the subsequent learning task. For instance, a simple approach might be to duplicate one of the samples already present in the group according to a sampling function that follows a certain distribution (e.g. uniform). It might be also possible to adopt other well-known generative approaches in the literature. In this work, we adopt a uniform sampling for the removal step while we will test and discuss three generative approaches (i.e. Uniform Sampling, SMOTE and ADASYN) in the experimental section 9.3.

9.3 Experimental analysis

This section describes the experiments we have conducted to evaluate DEMV: section 9.3.1 reports the used experimental setting comprising the selected metrics and baselines; section 9.3.2 describes the employed datasets and their characteristics; section 9.3.3 reports on the analysis we have performed to select the best instance generation strategy to be plugged-in DEMV; section 9.3.4 shows DEMV's evaluation results both in multi-class and binary classification; and finally, section 9.3.5 reports a description on how to reproduce the performed experiments using the available code. We like to remark that the full implementation of DEMV and the code to reproduce all the perfomed experiments is available at the Territori Aperti RI[7] and on GitHub[8] as well.

9.3.1 Experimental setting

We evaluate DEMV under heterogeneous conditions by applying a set of binary and multi-class datasets. As a base classifier, we used a Logistic Regression model [182] since it is very efficient from a computational point of view, natively supports

[7] https://bit.ly/3scwtaB
[8] https://github.com/giordanoDaloisio/demv2022

multi-class classification, and being a white-box method, it is comprehensible and promotes transparency. In addition, we also performed some specific experiments involving more sophisticated classifiers to analyze the impact of DEMV on these methods. The involved classifiers are: Gradient Boosting [105], Support Vector Machine (SVM) [195], and Neural Network with *ReLU* activation function [117]. For all the experiments, we adopt the implementation from the `scikit-learn` library [199] with the default hyper-parameters.
For all the experiments, we compute the following metrics on the testing set:

- *Absolute Statistical Parity (SP)*, defined as the absolute value of the original Statistical Parity from [86]. We normalized this metric to reduce his variability and better evaluate each method's performance (i.e., avoid situations in which we measure values like 0.2 and -0.2 in two different runs, resulting in a mean of zero with a high standard deviation). The optimal value is **zero**.

- *Disparate Impact (DI)* [99], where the optimal value is **one**. To avoid the occurrence of reverse bias (i.e., metric value firmly higher than one), we adopt the formulation proposed by [205]:

$$\text{DI} = min\left(\frac{p(\hat{y}=1|s=1)}{p(\hat{y}=1|s=0)}, \frac{p(\hat{y}=1|s=0)}{p(\hat{y}=1|s=1)}\right) \tag{9.3}$$

  This metric computes the minimum among two formulations of DI wherein one, the unprivileged group ($s = 0$) is at the numerator, and the other is at the denominator. The metric value is between zero and one, where one means complete fairness.

- *Absolute Equalized Odds (EO)*, defined as the absolute value of the original Equalized Odds from [119]. We normalized this metric for the same reasons as SP. The optimal value is **zero**.

- *Zero-one Loss (ZO Loss)* [81], where the optimal value is **zero**.

- *Accuracy (Acc)* [213], where the optimal value is **one**.

- *Harmonic Mean (H-Mean)* [100] of the above metrics. In particular, concerning the metrics whose optimal value is zero (i.e., SP, EO, and ZO Loss), we beforehand perform a value's permutation to have the optimal value equal to one, then we compute the H-Mean using these new values. Formally, H-Mean is computed as follows:

$$\text{H-Mean} = \frac{5}{\frac{1}{(1-|SP|)} + \frac{1}{(1-|EO|)} + \frac{1}{(1-|ZOLoss|)} + \frac{1}{DI} + \frac{1}{Acc}} \tag{9.4}$$

Table 9.1 shows the list of performed experiments. Specifically, we performed four main sets of experiments.
The first is the comparison of different implementations of DEMV embedding diverse generative strategies (see section 9.3.3). We consider the following three strategies:

- *Random sampling on Uniform Distribution (UNIFORM)*, where the algorithm duplicates an item present in the group with a uniform probability distribution;

- *Synthetic Minority Oversampling Technique (SMOTE)* from [42];

TABLE 9.1: Description of the performed experiments

Experiment	Reference section	Scope	Task	Involved classifier	Number of sensitive vars	Involved debiaser methods
1	9.3.3	Comparison of different implementations of DEMV embedding diverse generative strategies both in binary and multi-class classification	Binary and multi-class	Logistic Regression	2	DEMV Uniform DEMV Smote DEMV Adasyn
2	9.3.4	Analyze the behavior exposed by debiaser methods with sensitive groups identified by a different number of sensitive variables	Binary	Logistic Regression	1	No one EG Grid Blackbox DEMV
					2	No one EG Grid DEMV
					3	No one EG Grid DEMV
3	9.3.4	Analyze the behavior exposed by debiaser methods with sensitive groups identified by a different number of sensitive variables	Multi-class	Logistic Regression	1	No one EG Grid Blackbox DEMV
					2	No one EG Grid DEMV
					3	No one EG Grid DEMV
4	9.3.4	Analyze the behavior exposed by debiaser methods involving more sophisticated classifiers both in binary and multi-class classification	Binary and multi-class	Logistic Regression[a] Gradient Boosting Support Vector Machine Neural Network	2	No one EG Grid DEMV

[a] This classifier has not been directly employed in this experiment, but for clearness we report the results obtained in the previous experiments

- *Adaptive Synthetic Sampling Approach (ADASYN)* from [122].

This analysis has been performed in binary and multi-class classification tasks on all the considered datasets considering two sensitive variables and using the Logistic Regression as a classifier.

After identifying and settling on the best generation strategy, we compare DEMV with the selected baselines by performing three main sets of experiments (please refer to section 9.3.4). Experiments two and three in table 9.1 are focused on analysing the behavior exposed by debiaser methods with sensitive groups identified by one, two and three sensitive variables. Experiment two focuses on binary classification task (see sub-sections 9.3.4), while experiment three focuses on multi-class classification task (see subsection 9.3.4). In both these experiments we employed a Logistic Regression model as a classifier. To have a more concrete representation of the behavior of DEMV and the other baselines, at the end of section 9.3.4 we also report a comparison of normalized confusion matrices for the privileged and unprivileged groups of a particular dataset.

Finally, experiment four in in table 9.1, is devoted to analyze the behavior exposed by debiaser methods involving more sophisticated classifiers: Gradient Boosting [105], Support Vector Machine (SVM) [195], and Neural Network with *ReLU* activation function [117]. Since these models are more complex from a computational point of view, this last experiment has been performed in binary and multi-class classification tasks on a reduced but heterogeneous data set considering the two established sensitive variables (see sub-section 9.3.4).

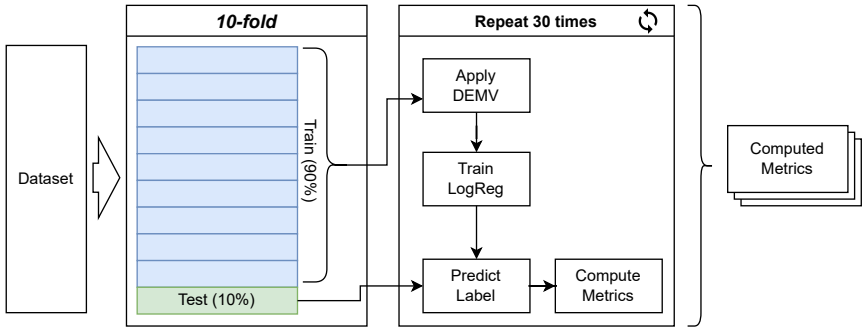


FIGURE 9.3: Evaluation procedure of DEMV for each train-test fold

In all the experiments (with the exception of experiment one) we compare with the following baselines:

- a biased classifier, where no debiasing method is applied, identified in the following by ***No one***;
- the *Exponentiated Gradient* (***EG***) and *Grid Search* (***Grid***) in-processing methods from [3][9];
- the ***Blackbox*** post-processing method from [201][10]. This method has been employed only in the analyses with one sensitive variable since, by the time of this paper, it does not support multiple sensitive variables.

Concerning Exponentiated Gradient and Grid Search, in agreement with the documentation available online [93], we used the Absolute Statistical Parity and the Zero-one Loss as constraints for binary and multi-class problems, respectively. Instead, Blackbox does not require a specific configuration of the hyperparameters.
For all the experiments showed in table 9.1 we follow a *10-fold* cross-validation [208], repeated 30 times for those methods that expose a stochastic behaviour (namely DEMV) as depicted in figure 9.3. In particular, to better reproduce a production scenario, we apply DEMV only on the training set and train the Logistic Regression classifier using the balanced dataset. Then, we predict the labels using the original biased testing set and compute the metrics described above. In addition, since the balancing of the groups has a stochastic behavior, for each train-test fold, we repeat the aforementioned process 30 times so that we can investigate how the removal or duplication of different items can influence the *accuracy* and the *fairness* of the classifier.
In all the performed experiments, we report the mean and standard deviation of all the metrics calculated over all the involved datasets. In the representation of such metrics we use bar plots where larger bars depict the mean of the metrics and thin bars show their standard deviation. In representing plots, we distinguish between metrics whose optimal value is 0 (showed on the left side of the figures) and metrics whose optimal value is 1 (reported on the right side of the figures).

[9]The adopted implementation of Exponentiated Gradient and Grid Search methods are available on the Fairlearn library [21]

[10]The considered Blackbox implementation is available at the following link: `https://github.com/scotthlee/fairness`

9.3.2 Employed datasets

The experiments are conducted by employing nine well-known datasets (3 for the binary classification and 6 for the multi-class task) from the Bias and Fairness literature. For each dataset, we consider sensitive groups identified by three sensitive variables: two variables are the ones established as sensitive variables by the literature, while the third one is selected, for each dataset, among the variables that could create discrimination, like age, education and so on. Note that PARK dataset has been excluded by the analysis with three sensitive variables because it does not have a third variable suitable for discrimination.
Table 9.2 depicts the descriptive statistics for the employed datasets. Concerning the sensitive variables, we highlight in bold the two ones established as sensitive by the literature. In the following, it is provided a brief description of the 9 considered datasets.

1. **Adult Income (ADULT)** [144]: This binary dataset comprises 30,940 items by 102 features (one-hot encoded). The goal is to predict if a person has an income higher than 50k a year. This information is represented by the `income` variable. The protected attributes are `sex`, and `race` and the unprivileged group is *black women* (items with `sex` and `race` equal to zero). In the analysis with three sensitive variables, we also introduced the `bachelor` variable, indicating if a person has a bachelor's degree or not. In this case, the sensitive group is *black women with no bachelor's degree*. The positive label is *high income*.

2. **ProPublica Recidivism (COMPAS)** [10]: This binary dataset is made of 6,167 samples by 399 attributes. The sensitive variables are `sex` and `race`. The goal is to predict if a person will recidivate in the next two years. The favorable label, in this case, is *no*, and the unprivileged group is *Non-Caucasian men* (items with `sex` and `race` equal to zero). In the test with three sensitive variables, we also introduced the `age` attribute. In this case, the sensitive group is `Non-Caucasian men with less than 50 years`.

3. **German Credit (GERMAN)** [206]: This binary dataset classifies people described by a set of attributes as good or bad credit risks (`credit` variable). The dataset consists of 1,000 instances by 59 features (one-hot encoded). The sensitive variables are `sex`, and `age` and the unprivileged group is *women with less than 25 years*. The positive label is *low credit risk*. In the experiment with three sensitive variables, we also introduced the `investment_as_income` variable, meaning if a person has more than the 30% of his income invested. In this case, the sensitive group is *women with less than 25 years and with less than 30% of their income invested*.

4. **Contraceptive Method Choice (CMC)** [162]: This multi-class dataset comprises 1,473 instances and ten columns about women's contraceptive method choice (*not-use*, *short-use*, and *long-use*). The sensitive variables are `religion` and `work`. The unprivileged group is *Islamic women who do not work* (both values equal one), and the positive label is *long-term use*. In the analysis with three sensitive variables, we introduced the education (`edu`) variable. The sensitive group, in this case, is *Islamic women who do not work and with no education*.

5. **Communities and Crime (CRIME)** [207]: This multi-class dataset is made of 1,994 instances by 100 attributes and contains information about the per-capita

violent crimes in a community (variable `ViolentCrimesPerPop`). Since the label is continuous, we transformed it by grouping the values in 6 classes using equidistant quantiles. Following [33] the sensitive attribute is the percentage of the black population, but we also considered the ratio of the Hispanic population to have two sensitive variables. The unprivileged group is communities with a *high percentage of both black and Hispanic people* (both variables equal to 1), and the positive label is `100` (class of *low rate of crimes*). In the experiment with three sensitive variables, we also introduced the `MedRent` variable, showing the average price of rents in a community. In this case, the unprivileged group is *communities with a high percentage of black and Hispanic people and a low cost of the rent*.

6. **Drug Usage (DRUG)** [98]: This multi-class dataset has 1,885 instances and 15 attributes about the frequency of drugs consumption (variable `y`). The classes are *never used*, *not used last year*, and *used last year*. The sensitive variables are `race` and `gender` and the unprivileged group are *white women* (`race` equal to one and `gender` equal to zero). The positive label is *never used*. In the test with three sensitive variables, we also used the `age` variable. In this case, the sensitive group is *white women less than 50 years*.

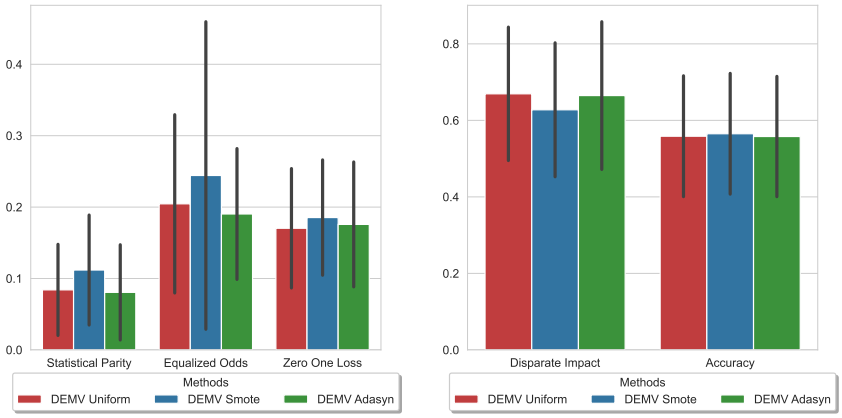
7. **Law School Admission (LAW)** [13]: This multi-class dataset comprises 20,694 samples by 14 attributes and contains information about the bar passage data of Law School students. We grouped the continuous label (`GPA`) in 3 groups using equidistant quantiles. The sensitive variables are `race` and `gender` and the unprivileged group are *black women* (both variables equal to one), and the positive label is 2 (class of *high scores*). In the analyses with three sensitive variables, we also introduced the `age` variable. In the experiments with three sensitive variables, we also used the `age` variable. In this case, the unprivileged group is *black women with less than 61 years*.

8. **Parkinson's Telemonitoring (PARK)** [247]: This multi-class dataset comprises 5875 items and 19 features about Unified Parkinson's Disease Rating Scale (UPDRS) score classification (variable `score_cut`). The classes are *Mild*, *Moderate* and *Severe*. The sensitive variables are `sex` and `age` and the unprivileged group are *males with more than 65 years* (`age` equal to one and `sex` equal to zero). Since this dataset does not have a third variable suited for identifying sensitive groups, we used it only in the experiments with one and two sensitive variables.

9. **Wine Quality (WINE)** [55]: This multi-class dataset comprises 6,438 instances and 13 attributes about wine quality (variable `quality`). The classes are four increasing values indicating quality (the higher, the better). The sensitive attributes are the wine's color (`type` variable) and the alcohol percentage lower or higher than 10 (`alcohol` variable). The unprivileged group is *white wine with an alcohol percentage* ≤ 10, and the positive label is 6 (*high quality*). In the experiment with three sensitive variables, we also introduced the `density` variable. In this case, the unprivileged group is *white wine with an alcohol percentage* ≤ 10 *and with a density less than 1.1%*.

TABLE 9.2: Descriptive statistics for the employed Datasets (boldface are highlighted the protected variables established in the original dataset)

	Adult	Compas	German	CMC	Crime	Drug	Law	Park	Wine
Scope	Social	Justice	Social	Social	Justice	Social	Education	Health	Food
Instances	30,940	6,167	1,000	1473	1,994	1,885	20,427	5,875	6,438
Features	102	399	59	10	100	15	14	19	13
Classes	2	2	2	3	6	3	3	3	4
Positive label	*high income*	*no*	*low-credit risk*	*long-term use*	*100 (low percentage class)*	*never used*	*2 (high scores class)*	*mild*	*high quality*
Sensitive variables	**sex race** bachelors	**sex race** age	**sex age** investment	**religion work** edu	**black hisp** Medium Rent	**race gender** age	**gender race** age	**age sex**	**type alcohol** density
Percentage of sensitive group with two sensitive vars	5.02%	54.71%	10.50%	64.83%	23.62%	45.78%	8.42%	39.45%	11.40%



(A) Metrics whose best value is zero (B) Metrics whose best value is one

FIGURE 9.4: Comparison of generation strategies of DEMV for multi-class classification with two sensitive variables

9.3.3 Selection of the best generative strategy

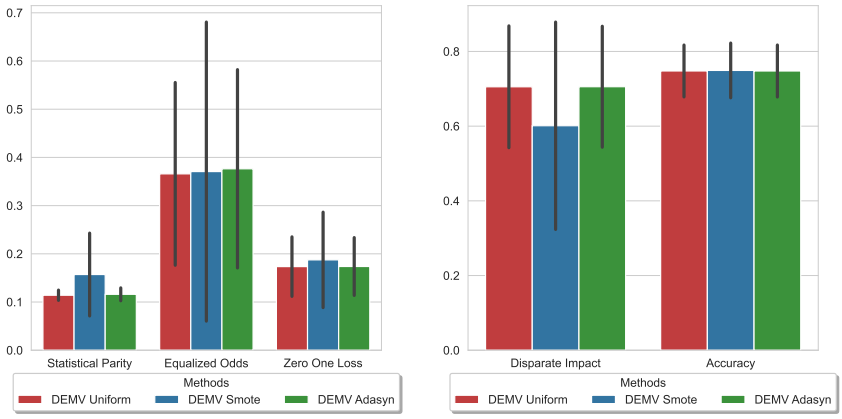
In this section, we show the experiments made to select the best instance generation strategy to plug-in in DEMV. As described in section 9.3.1, we consider the following generation strategies: Uniform sampling, SMOTE, and ADASYN. We perform the comparison with both binary and multi-class datasets using, for each dataset, sensitive groups identified by the two sensitive variables specified in literature.

The aggregated metrics for multi-class datasets are shown in figure 9.4. From this first analysis, *Uniform* sampling and *ADASYN* give similar results in fairness and accuracy, while *SMOTE* behaves worse.

Figure 9.5 confirms that, also in the case of binary classification, the *Uniform* sampling and *ADASYN* have comparable performances concerning accuracy and fairness.

Since both the *Uniform* sampling and *ADASYN* generative strategies expose similar performance in terms of classifier's fairness and accuracy, we decide to analyse their computational performances in order to select the best and efficient strategy to embed in DEMV. In particular, we focused on their execution time (expressed in seconds) that we report in figure 9.6.

This experiment has been conducted on a *MacBook Air M1 2020* with 16 GB of RAM.



(A) Metrics whose optimal value is zero (B) Metrics whose optimal value is one

FIGURE 9.5: Comparison of generation strategies of DEMV for binary classification

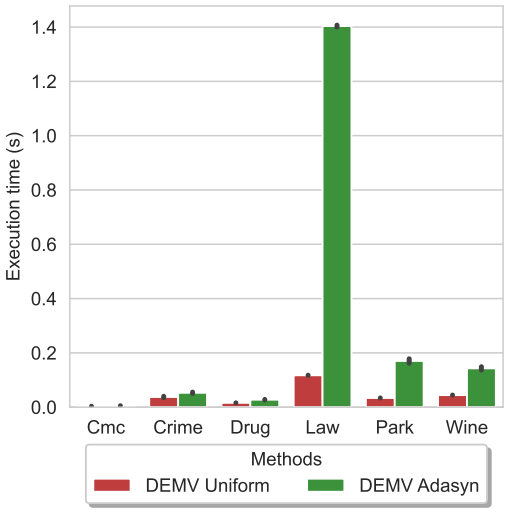


FIGURE 9.6: Execution time in seconds of DEMV Uniform and DEMV Adasyn in multi-class classifications tasks

The results show that DEMV implementing *ADASYN* takes much more time for completion, especially in larger datasets, while DEMV with *Uniform* always takes a reasonable execution time.
Considering all the analysis made, we adopt the *Uniform* sampling as the generation strategy to compare against the baselines because the obtained metrics are comparable to ADASYN and its execution time is lower.

9.3.4 DEMV evaluation in classification tasks

This section presents the quantitative results of the DEMV's evaluation. We compare the performance of DEMV with the selected baselines shown in section 9.3.1.
Even if DEMV is a debiaser for the multi-class classification problem, we decided to evaluate it also in binary classification problems to identify its potentialities in this scenario (see section 9.3.4). However, since the binary classification task is not the primary scope of our work, we decide for readability to show, for each method, the variation of H-Means at the increasing of sensitive variables.
In section 9.3.4, we present the DEMV's evaluation with multi-class classification tasks. In this case, we report the mean and standard deviation of each measure described in the section 9.3 using the bar plots. Then, as an overall view, we show the variation of each method's H-Mean at increasing sensitive variables.
Finally, in both binary and multi-class classification scenarios:

- for each dataset, the variation of H-Means, at the increasing number of sensitive variables is reported using line plots in which each line identifies one method. We recall that since the *Blackbox* algorithm does not support multiple sensitive variables, it has been applied only in the experiments involving sensitive groups identified by one sensitive variable, so it is represented as a point in such plots;
- as already said, each dataset has two sensitive variables. To run the experiment with one sensitive variable, we averaged the results of two independent experiments, one for each sensitive variable;
- it is reported the statistical significance of all experiments computed using the ANOVA test in each analysis [178]. This test checks for the null hypothesis that all groups have the same mean; if the probability value (*p-value*) is less than 0.05, the test rejects the null hypothesis, which means that the groups have a different mean value.

Comparison in the binary classification task

In this section, we compare DEMV with the other baselines in a binary classification context. It is worth noting that, in the context of binary classification with one sensitive variable, DEMV coincides with the original *Sampling* method [133] it derives from.
The results of the experiments are reported in figure 9.7. As reported in the figure, DEMV (represented with the red line) better mitigates the bias with an arbitrary number of sensitive variables, producing results that are generally competitive and even better when more than two variables are considered. A closer analysis lets us notice that EG outperforms the other methods when the number of sensitive variables is one or two. At the same time, it dramatically fails when three sensitive variables are considered. Blackbox method (reported by a single triangle in correspondence to one variable) is a good performer only when one sensitive variable is

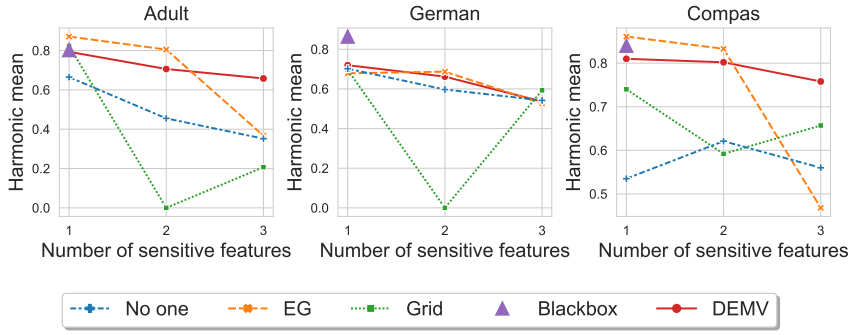


FIGURE 9.7: Comparison of overall H-Mean at different number of sensitive variables for binary classification datasets

needed. In contrast, its adoption will not be applicable in cases where more sensitive variables must be considered.

TABLE 9.3: Overall H-Mean of all methods with different sensitive variables in the binary classification context

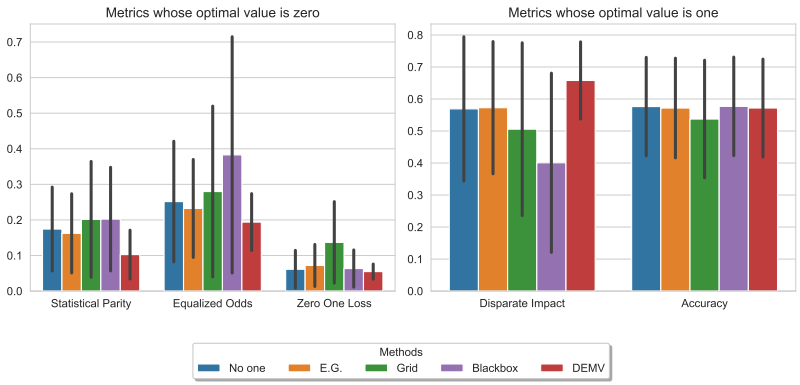
Sensitive variables	**Methods**				
	No one	**Blackbox**	**EG**	**Grid**	**DEMV**
1	0.648 ± 0.034	$\mathbf{0.835 \pm 0.031}$	$\mathbf{0.835 \pm 0.048}$	0.761 ± 0.056	0.777 ± 0.036
2	0.558 ± 0.09	-	$\mathbf{0.775 \pm 0.077}$	0.197 ± 0.342	0.723 ± 0.072
3	0.485 ± 0.115	-	0.454 ± 0.081	0.486 ± 0.243	$\mathbf{0.651 \pm 0.111}$

To give an overall view of the performances of each method, we provide a synthetic version of the above results in table 9.3 where, for each method, we report the average of the H-Mean computed overall for the considered datasets. This summary confirms what we observed in the details above; that is, in binary classification with one sensitive variable, Blackbox and EG perform similarly. EG also behaves well in case of two variables. Finally, DEMV produces competitive results with one or two sensitive variables while outperforming the other baselines when three variables are needed. However, no clear winner can be picked out of the shelf, and more evaluation should be provided to determine which method to apply in different settings, including dataset characteristics. In addition, a quality that can be decisive in selecting the best method is the computational complexity, which we will consider in the future for a better evaluation of DEMV.

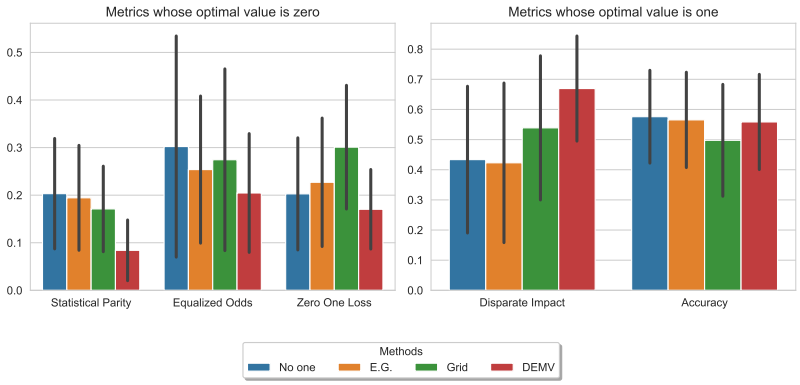
Comparison in the multi-class classification task

In this subsection, we report the results of the experiments conducted in the context of multi-class classification. The experiment's results are reported in figure 9.8 and figure 9.9. The former reports the mean and the standard deviation of the metrics computed by each method on overall datasets, distinguishing among the usage of one (a), two (b), and three (c) sensitive variables. The latter instead, provides a different view, and for each dataset, it reports the values of H-Mean for each method at the increasing of sensitive variables.

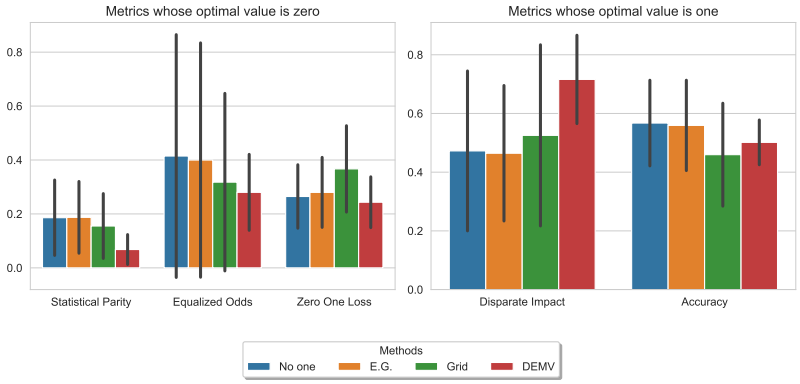
In particular, figure 9.8a focuses on the experiments involving one sensitive variable. The high standard deviation of all metrics is explained by the fact that the metrics are here calculated putting together the results of two separate experiments. From the



(A) Application with one sensitive variable



(B) Application with two sensitive variables



(C) Application with three sensitive variables

FIGURE 9.8: Comparison of DEMV with the baselines in multi-class classification

figure we can see that, in average, DEMV overcomes all the baselines. In addition, we observe that DEMV is the method performing in a more stable and coherent way. This is highlighted by an overall lower standard deviation for all metrics.

The performances of DEMV in case of one variable are confirmed by figure 9.9, where it can be seen that DEMV overcomes the baselines in all dataset with the only exception of CMC (in which the best method is Grid), and Wine (in which the best method is Blackbox).

Figure 9.8b reports the results of the experiments with sensitive groups identified by two sensitive variables. As before, DEMV overcomes all the other baselines in all the involved datasets, and its stability is confirmed by an overall lower standard deviation. Figure 9.9 shows that, also in this context, DEMV outperforms the baselines

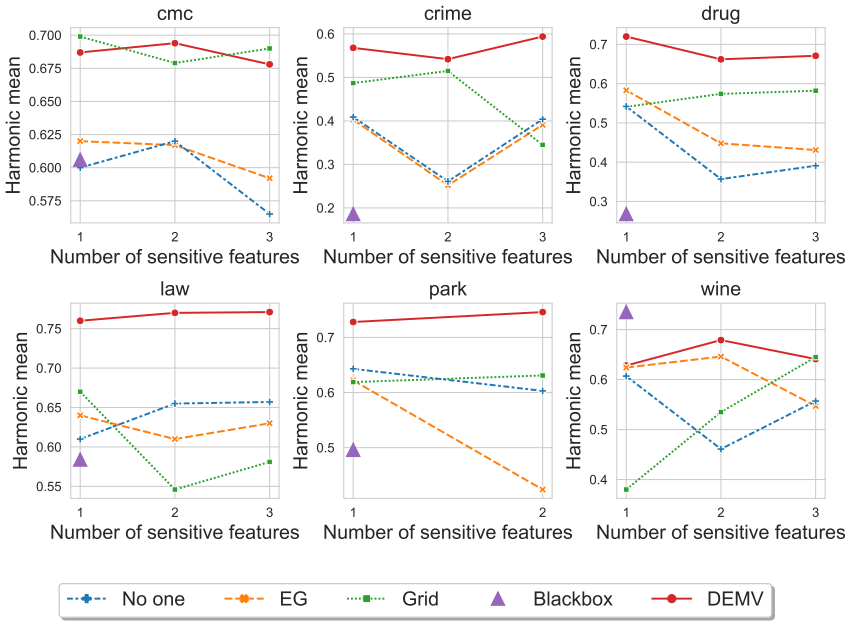


FIGURE 9.9: Comparison of overall H-Mean at different number of sensitive variables for multi-class classification datasets

in all datasets.

We recall that the Park dataset has not been used in this experiment since it does not have a third variable suitable to be treated as sensitive.

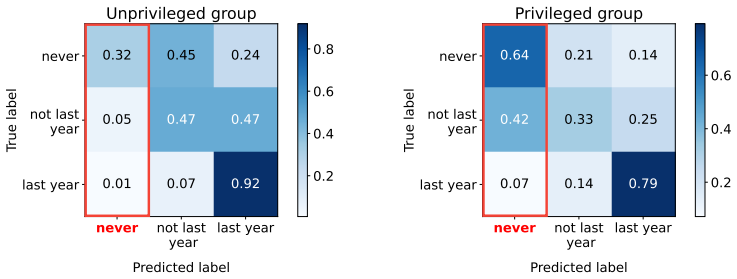
In this case, from figure 9.9 it can be seen that Grid performs slightly better in CMC (with a delta of H-Mean of about 0.01 points) and Wine (with a delta of 0.005). As for the other two experiments, DEMV performs more consistently with an overall standard deviation lower than the different baselines. Again, the ANOVA test confirms the statistical significance of the experiments, with the only exception of EO metrics (with a p-value of 0.27) which has a high variability especially with EG and with the biased classifier (identified by *No one* label) in the figure.

TABLE 9.4: Overall H-Mean of all methods with different sensitive variables in the multi-class classification context

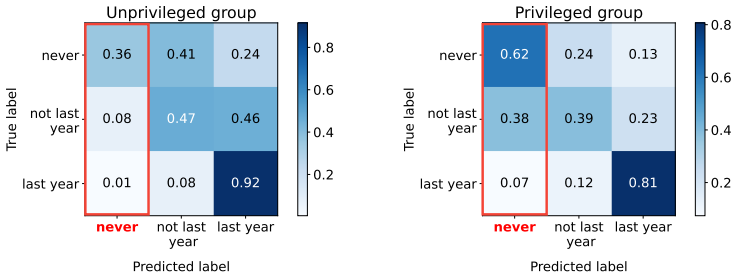
Sensitive variables	**Methods**				
	No one	**Blackbox**	**EG**	**Grid**	**DEMV**
1	0.568 ± 0.085	0.479 ± 0.211	0.582 ± 0.09	0.566 ± 0.121	$\mathbf{0.682 \pm 0.072}$
2	0.493 ± 0.16	-	0.505 ± 0.16	0.58 ± 0.063	$\mathbf{0.677 \pm 0.081}$
3	0.486 ± 0.135	-	0.49 ± 0.128	0.529 ± 0.182	$\mathbf{0.646 \pm 0.08}$

As for the experiments involving binary classification datasets, in order to have a complete concise overview, in table 9.4 we report the overall H-Mean of all the methods in the three performed experiments overall the datasets. Note that, DEMV generally overcomes the other baselines in all the explored contexts increasing the H-Mean by up to 0.2 points with respect to the biased classifier (i.e., *No one* in the table) in the experiments with two and three sensitive variables.

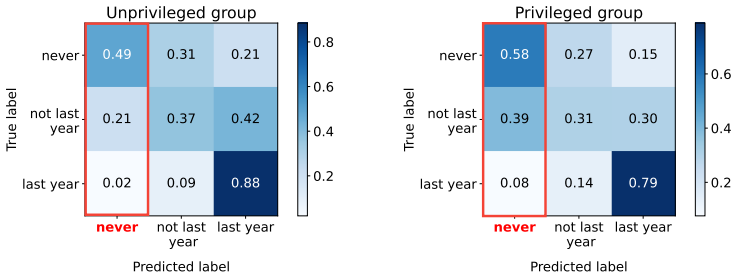
Finally, to have a more concrete representation of the behavior of all the analyzed methods, in figure 9.10 we report a comparison of the normalized confusion matrices [149] for the privileged and unprivileged (i.e., biased) groups of the Drug dataset



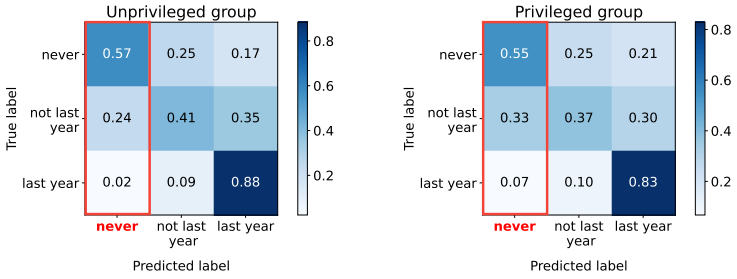
(A) Biased classifier



(B) EG



(C) Grid



(D) DEMV

FIGURE 9.10: Normalized confusion matrices of privileged and unprivileged groups for each baseline on Drug dataset

with two sensitive variables. We decided to choose the Drug dataset for this experiment since it is among the ones having a high bias and showing better the inequality among the privileged and unprivileged groups. In all the matrices, we highlight in red and in boldface the predicted positive label (i.e., *never*), which identifies the column of the matrix affected by bias (highlighted in red as well). In particular, figure 9.10a shows the confusion matrices of the biased classifier. From the picture, it can be seen how the probability of the privileged group having a predicted positive label (i.e., column corresponding to *never*) is much higher than the unprivileged

group. The confusion matrices related to EG and Grid (figures 9.10b and 9.10c respectively) do not differ much from the ones of the biased classifier, meaning that these two methods are not able to improve the fairness of the classifier. Instead, in figure 9.10d, it can be seen how DEMV is able to balance these two matrices, and the probability of having the positive label predicted is almost the same for the two groups, meaning that the fairness of the classifier has increased.

Comparison using more sophisticated classifiers

In this subsection, we report the results of the experiments conducted in binary and multi-class classification context using more complex classifiers. As already described in section 9.3.1, the employed classifiers are: Gradient Boosting, Support Vector Machine (SVM), and Neural Network with *ReLU* activation function. Since these models are more complex from a computational point of view, we performed these experiments on a reduced, but heterogeneous set of data using sensitive groups identified by two sensitive variables. The selected datasets are: Adult (binary large dataset), COMPAS (binary small dataset), CMC (multi-class small dataset), and Law (multi-class large dataset).

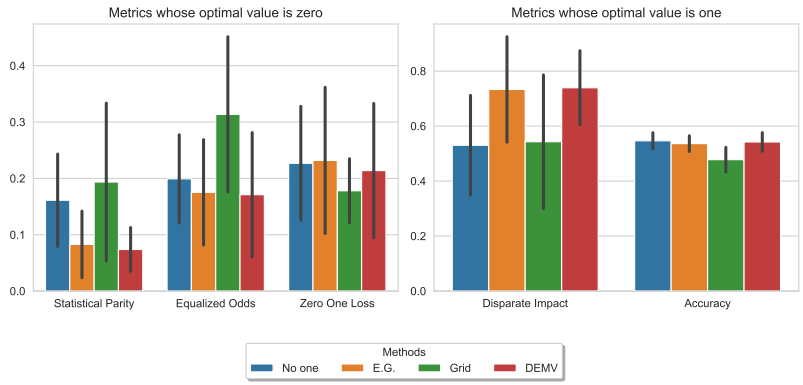
Finally, considering the debiaser approaches, it is worth noting that EG and Grid can not be applied when a Neural Network model is used as classifier. In fact, EG and Grid apply arbitrary weights to the instances in order to remove bias, but Neural Networks by their nature do not allow the specification of weights to the instances. For this reason, in the experiments involving Neural Networks, we only compared the performance of the original classifier with the performance of the classifiers after the application of DEMV.

TABLE 9.5: Overall H-Mean of all methods with different classifiers in the binary classification context

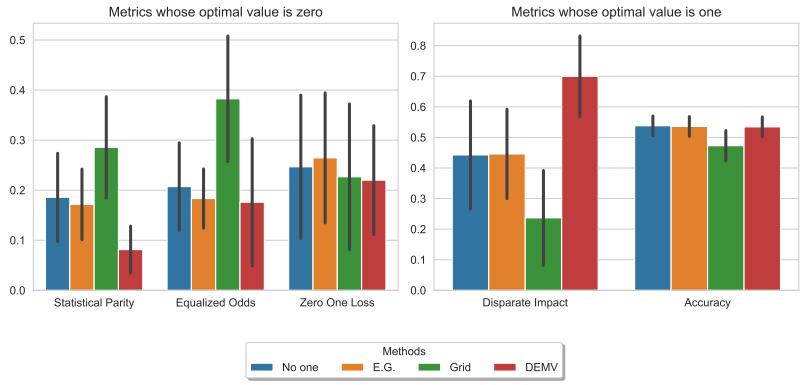
Classifier	**Methods**			
	No one	**EG**	**Grid**	**DEMV**
Logistic Regression	0.558 ± 0.09	$\mathbf{0.775 \pm 0.077}$	0.197 ± 0.342	0.723 ± 0.072
Gradient Boosting	0.588 ± 0.19	0.476 ± 0.383	0.582 ± 0.228	$\mathbf{0.724 \pm 0.057}$
SVM	0.57 ± 0.201	0.554 ± 0.238	0.59 ± 0.205	$\mathbf{0.721 \pm 0.066}$
Neural Network	0.584 ± 0.202	-	-	$\mathbf{0.69 \pm 0.127}$

Concerning binary classification, table 9.5 reports the overall H-Mean of all the baselines for each involved classifier overall the involved datasets. Note how, differently from the experiments with a Logistic Regression classifier, DEMV overcomes the other baselines in all of the performed analyses, with a delta up to around 0.2 points in case of EG with a Gradient Boosting classifier.

Concerning multi-class classification, figure 9.11 reports the mean and the standard deviation of all the metrics computed overall datasets. In particular, figure 9.11a shows the results of the experiments involving the Gradient Boosting classifier. In this context, DEMV outperforms the baselines under the SP and EO definitions of fairness, while it almost equals EG under the DI definition of fairness. The ANOVA



(A) Application with Gradient Boosting



(B) Application with Support Vector Machines

(C) Application with Neural Networks

FIGURE 9.11: Comparison of DEMV with the baselines in multi-class classification using other classifiers

test confirms the statistical significance of this experiment, with the only exception of Zero One Loss which has a p-value of 0.732. Figure 9.11b, reports instead the results of the experiments involving Support Vector Machines. In this context, it can be seen how DEMV overcomes the baselines under all the considered definitions of fairness, keeping an accuracy level almost equal to the original biased classifier. Finally, figure 9.11c details the results of the experiments performed with Neural Networks. We recall that in this case EG and Grid can not be applied, hence we compared DEMV only with the biased classifier. Also in this case DEMV is able to improve the fairness of the classifiers keeping an almost unchanged level of accuracy.

Table 9.6 reports the overall H-Means of all the methods for each classifier overall

TABLE 9.6: Overall H-Mean of all methods with different classifiers in the multi-class classification context

Classifier	Methods			
	No one	**EG**	**Grid**	**DEMV**
Logistic Regression	0.493 ± 0.16	0.505 ± 0.16	0.58 ± 0.063	$\mathbf{0.677 \pm 0.081}$
Gradient Boosting	0.653 ± 0.061	0.72 ± 0.049	0.607 ± 0.121	$\mathbf{0.729 \pm 0.018}$
SVM	0.603 ± 0.069	0.613 ± 0.054	0.392 ± 0.127	$\mathbf{0.716 \pm 0.012}$
Neural Network	0.656 ± 0.038	-	-	$\mathbf{0.728 \pm 0.016}$

the selected datasets. It can be seen how DEMV generally overcomes the other baselines with all the selected classifiers with a delta up to around 0.3 points concerning SVM with Grid method.

9.3.5 Reproducibility of the experiments

Nowadays, ensuring that the proposed methods and their corresponding results are sound and reliable is one of the challenges for research in machine learning. To ensure that the findings are valid, it is essential for the experiments to be repeatable and to yield results and conclusions comparable or identical to the originally reported ones [200]. For this reason, we choose to release the full code of the DEMV algorithm along with the replication package of all the performed experiments. This section is dedicated to describing how to use such code in order to reproduce the experiments described in the previous sections. We recall again that the full implementation is available in the Territori Aperti RI and on GitHub as well. The repository also includes a specification of all the python dependencies required for a correct execution of the code, which can be installed using `anaconda`[11] or `pip`[12].
The program that must be called to replicate the experiments is `generatemetrics.py`, which is responsible to generate the measures computed and aggregated in all the experiments of section 9.3. Noticing that the code to reproduce the plots presented in this paper has not been included in the replication package, but can be easily implemented using the metrics generated from the given program. The code `generatemetrics.py` can be invoked from the command line through the python interpreter in the following way:

```
$ python generatemetrics.py <DATASET> <METHOD> <
    NUMBER_OF_FEATURES> --sensitivefeature <
    SENSITIVE FEATURE?> --classifier <CLASSIFIER?>
    --cm <CM?>
```

and accepts the following parameters (please refer also to the `README` of the GitHub repository for a more precise description of these parameters):

[11] https://www.anaconda.com/
[12] https://pypi.org/project/pip/

- **DATASET**: the dataset on which apply the experiments. Can be one of the datasets described in section 9.3.2.
- **METHOD**: debiaser method to use. Can be one of the debiaser methods employed in this work or `biased` in case of no methods.
- **NUMBER OF FEATURES**: number of sensitive variables to identify the sensitive groups. Can be an integer up to 3.
- **SENSITIVE FEATURE**: optional parameter to specify the sensitive variables for the identification of the sensitive groups in case `NUMBER OF FEATURES` is equal to one or two. To ensure that the selected variables are truly sensitive, they must be among the three sensitive variables defined for each dataset in section 9.3.2.
- **CLASSIFIER**: optional parameter to specify a classification method. Can be one of the classifiers employed in the experiments of this paper. The default is the Logistic Regression classifier.
- **CM**: optional boolean value to plot the confusion matrices of the two sensitive groups. Default is `false`.

The following command can also be executed to receive help and information on the requested parameters:

```
$ python generatemetrics.py -h
```

The execution of this script will produce a `.csv` file containing all the measures described in section 9.3.1 for each train-test fold. We like to remark that, in case of DEMV, the number of measures will be equal to 30 times the number of train-test folds (see the description of the experiment setting in section 9.3.1).
To reproduce the experiments of section 9.3.3, the script can be called passing as input any of the involved dataset, a number of sensitive variables equal to 2, and `UNIFORM`, `SMOTE` or `ADASYN` as debiaser method. For instance:

```
$ python generatemetrics.py cmc uniform 2 --
    sensitivevariable religion,work
```

will produce the metrics of the CMC dataset using the DEMV Uniform debiaser strategy. Aggregating the results of the execution of this script for all of the involved datasets and all of the analysed generation strategies makes possible to reproduce the experiments and charts shown in the section 9.3.3.
Similarly, it is possible to reproduce the experiments of section 9.3.4 by running `generatemetrics.py` on all the combinations of datasets, methods, number of sensitive variables and classification methods and then aggregating the results. For instance, the following command:

```
$ python generatemetrics.py adult eg 3 --classifier
    gradient
```

will generate the metrics for the Adult dataset with three sensitive variables, using EG debiaser method and the Gradient Boosting classifier.

Finally, confusion matrices for the privileged and unprivileged groups can also be created using this script. For instance, the command:

```
$ python generatemetrics.py crime uniform 3 --cm
```

will generate the confusion matrices of the Crime dataset for the privileged and unprivileged groups.
It is also possible to refer to the `README` file on the GitHub repository for a more complete description of the method and the parameters.

9.4 Discussion

In this section, we discuss the results of the experiments conducted in section 9.3, by referring to the research questions highlighted in section 9.1. Hence, we can draw the following considerations:

- **RQ1**. From the experiments conducted in section 9.3, we have seen that almost all the baselines can improve fairness in the binary classification context and classification problems involving one sensitive variable. However, no baselines can consistently handle bias in the multi-class classification domain with multiple sensitive variables either because they do not support it (like in the case of *Blackbox*) or because they perform very poorly (like in the case of *EG* and *Grid*). More specifically, the strengths and weaknesses we have observed for each baseline in the performed experiments are as follows:
  - **EG.** It can improve the fairness in the context of binary classification with very relevant results. It has much more difficulty in improving fairness in multi-class classification. This weakness might be imputed to the constraint metric suggested by the authors to be used in the case of multi-class classification, i.e., ZO Loss. In addition, this method is highly influenced by the involved classification algorithm and can not be applied if the employed classifier is a Neural Network.
  - **Grid.** His performances are strictly related to the dataset and the search space size. Since, in our experiments, we always used a grid size of 20 (the default value of the adopted implementation), this method performed well with some datasets and worse with others in which a larger search space was needed. This results in very high variability of the overall obtained metrics. In particular, our experiments observed that Grid performs well with the CMC and Wine multi-class datasets. At the same time, in the binary classification task, the Grid method exposes a higher variability even in the same dataset but among a different number of sensitive variables. Finally, as for EG, this method is strongly influenced by the classification algorithm and can not used if the employed classifier is a Neural Network.
  - **Blackbox.** This method performs well in mitigating bias in binary and multi-class classification. However, it does not support multiple sensitive variables. In addition, we observed high variability in the overall metrics that let the method be considered unstable.
- **RQ2**. In this work, we have presented the *Debiaser for Multiple Variables (DEMV)*, which is, to the best of our knowledge, the first *pre-processing* approach able to

improve fairness both in binary and multi-class classification with multiple sensitive variables. DEMV generally overcomes the other baselines in binary and multi-class classification tasks with one, two, and three sensitive variables. In addition, being a pre-processing method, DEMV can be applied to a heterogeneous set of classification methods without impacting or being influenced by their behavior. DEMV is also the method that performs more consistently in all the experiments, resulting in less variability of the overall metrics.

- **RQ3**. The generative strategy that must be adopted to rebalance groups in DEMV is the Uniform sampling. The Uniform generating strategy is preferable from two points of view: i) is the best performer (among the other generative strategies) in terms of fairness and accuracy; ii) its computational complexity is negligible. This approach has been adopted in DEMV and compared with the other baselines obtaining excellent results.

- **RQ4**. The performed experiments showed that DEMV can improve the fairness in binary and multi-class classification contexts with any number of involved sensitive variables keeping a high level of accuracy. In particular, our method overcomes the other baselines in multi-class classification problems with any number of sensitive variables. In contrast, as expected, other specifically designed methods may perform better in binary classification with one or two sensitive variables. Instead, we have noticed that when the sensitive variables are more than two, DEMV overcomes the baselines also in the case of binary classification. In addition, we have shown how DEMV can consistently improve the fairness of several classification methods without impacting their behavior, while other debiaser methods behave differently according to the involved classification method or can not be applied at all (like EG and Grid with a Neural Network). Finally, we observed that when the size of the sensitive group is tiny, DEMV has more difficulty improving fairness and finding the optimal group size. This issue can be explained by the fact that when the group size is small, the addition or removal of a single item impacts more on the expected and observed size ratio, so the optimal balancing is more complex or impossible to be achieved.

9.5 Conclusion and Future Work

In this paper, we addressed the problem of bias mitigation in the multi-class classification context by proposing the *Debiaser for Multiple Variables*, a novel approach extending the work of [133] to the multi-class classification domain with multiple sensitive variables. To the best of our knowledge, DEMV is the first pre-processing method able to handle bias in both binary and multi-class classification problems with any number of sensitive variables.

We have exhaustively evaluated our algorithm by comparing it with three established baselines using a heterogeneous set of binary and multi-class datasets, with a different number of sensitive variables, and by employing a heterogeneous set of classification methods. In addition, we have also evaluated how different instances generation strategies can influence the ability of DEMV in improving fairness. The conducted experiments show that our method is the better choice to adopt in the multi-class classification context with one, two, or three sensitive variables. Instead, we noticed how other specifically designed methods might perform better in binary classification with one and two sensitive variables. However, our method is still the

better solution in binary classification problems with three sensitive variables and, being a pre-processing method, it can be successfully applied even with classifiers not supported by other baselines (e.g., Neural Networks).
Finally, we have seen that the Uniform sampling of existing instances is the best strategy to manipulate groups in terms of accuracy and fairness.
In the future, we want to overcome the current main weakness of DEMV, highlighted when the sensitive groups are very small. We will address this issue by investigating if there are situations that lead to optimal fairness before a complete balance within the groups. If so, we want to identify further which are the characteristics that leads to these situations. In addition, we want to improve our analysis by studying the impact of the size of the dataset and the number of their attributes that are not the sensitive variables on DEMV and his behavior. Next, we also want to asses the computational complexity respect to the other baselines and we want to study the impact that different removal strategies may have during the balancing procedure and thus on the capacity of DEMV to mitigate bias and improving fairness. Finally, we will study the impact that DEMV have on the fairness of the full pipeline of recommender systems that embed a multi-class classifier.

Part III

Building Trustworthiness through Explainability

Chapter 10

Dense Retrieval and Explainability

Robustness is one of the three TDs of Trustworthy AI, and it includes explainability, as described in Section 1. By robustness, we refer to the capacity of machine learning models to remain reliable under distributional drift, whether arising from changes in the underlying data or from shifts in the *deployment environment*.

One key challenge in this area is making choices from ML models explainable to the end user. While trained ML practitioners may be able to interpret model behavior through technical diagnostics, less experienced users often require simpler, more grounded explanations, particularly in sensitive domains such as medicine [84, 114]. Providing such accessible explanations is therefore a crucial component of achieving practical robustness.

In recent years, large language models (LLMs) have grown enormously in influence, and so have Retrieval-Augmented Generators (RAGs), which augment LLM outputs with information retrieved from an external knowledge store. This development is directly relevant to robustness and explainability: when a model's answers depend not only on its parameters but also on what it retrieves, understanding why a specific piece of information was surfaced becomes essential for explaining the final output. Moreover, retrieval errors, such as retrieving irrelevant, outdated, or biased documents, represent a new and increasingly important source of drift.

For these reasons, for the research carried out in this thesis, we focus on dense retrieval, the core mechanism driving modern RAG systems. By studying how dense retrievers select and rank information, we aim to better understand and improve robustness of both retrievers and RAGs, while also enabling more transparent and user-oriented explanations of their behavior.

In retrieval-based systems, two components are critical: the retrieval of potentially relevant documents and the ranking of those documents according to estimated relevance. Both processes introduce opportunities for drift, bias, and lack of transparency. We will now introduce the task of Document Ranking that we aim to tackle. Then, we explore related work on this subject, including approaches that aim to be more explainable. Lastly, we outline our contribution.

10.1 Document Ranking

Document ranking is a fundamental task in the field of Information Retrieval [15], serving as a ground role across a broad spectrum of systems, from academic research platforms to large-scale web search engines. Given a corpus of documents and a user input query, the goal is to provide an ordered list of documents ranked by their relevance to the user's information need.

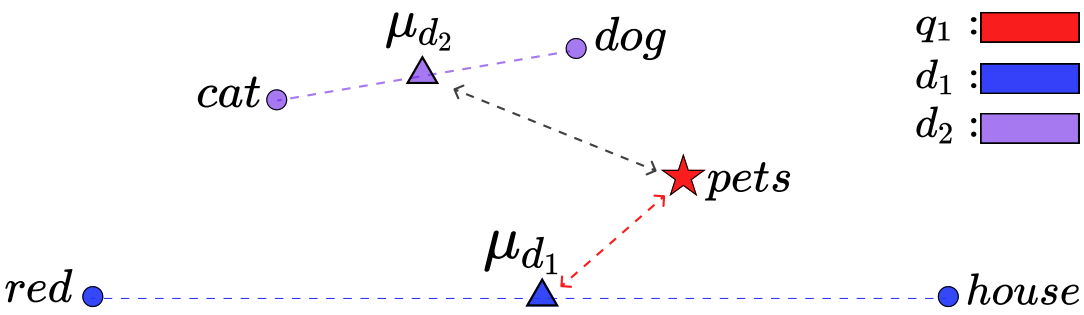


FIGURE 10.1: Toy example: Averaging embeddings can lead to sub-optimal results and loss of explainability.

Document ranking is inherently an unsupervised problem. Accordingly, we focus on the unsupervised setting and design our method specifically for this scenario, which serves as the reference framework throughout the Section.

The ranking of the documents is typically established on a relevance score computed by a function commonly referred to as the *scoring function*. Traditional approaches for document ranking, such as Bag-of-Words (BoW) models [217], represent documents as high-dimensional, sparse vectors based on term frequencies. However, these methods are now largely outdated, as they fail to capture semantic and contextual relationships between words.

Differently, recent advances leverage dense contextual embeddings produced by large language models (LLMs), enabling semantically richer representations of each token (word). Thus, in these models, document-level embeddings are typically obtained by aggregating token-level embeddings through a simple pooling function, such as max or mean pooling [209, 135] which typically involves aggregating the embeddings of individual tokens across a document or query. These pooling approaches are widely adopted due to their simplicity and efficiency, including in popular frameworks like HuggingFace [125].

It follows that the scoring function is typically formulated as the similarity (e.g., cosine similarity or inner dot product) between the pooled mean dense vectors of the query and the document [209]. Despite its widespread use, this simple approach introduces two fundamental limitations:

- *Suboptimal ranking results*: Aggregating an entire document into a single tensor tends to hide multiple semantic signals, leading to the loss of fine-grained contextual information, reduced sensitivity to nuanced meanings [248]. This limitation is particularly evident in long documents covering multiple topics. Empirical evidence of this issue is also discussed in Section 11.3.

- *Lack of control and explainability*: The previously mentioned loss of fine-grained semantic information also undermines the model's ability to perform effective credit assignment across related tasks. This reduction in granularity significantly limits the model's capacity to attribute relevance to specific terms within a document, thereby compromising the explainability of document ranking for end users [266, 8]. In the absence of clear term-level contributions, ranking decisions become opaque and less interpretable.

Both these problems are illustrated in Figure 10.1. It depicts the embeddings, in a bi-dimensional space, of two documents $d_1 := \{red, house\}$, $d_2 := \{cat, dog\}$ (in blue and light violet respectively in the figure), and of a query $q_1 := \{pets\}$ (in red

in the figure). As humans, we would expect that d_2 will be more relevant than d_1 in the documents' ranking because *cat* and *dog* are two *pets*. However, due to the use of the mean of the embeddings, the document d_1 will be considered closer (red dashed line with respect to the black dashed one) to the query embedding. Thus, it will be unexpectedly ranked first (i.e., considered more relevant). The mean of this illustrative example is to present to the reader in a straightforward manner how averaging an entire document's representation into a single aggregated tensor can lead to undesirable outcomes, and how the ranking will, consequently, become less explainable.

10.2 Related work

Embeddings Creation. Typical Document Ranking approaches are based on the *Vector Space Model (VSM)* [218], utilizing the *Bag of Words* [120] framework. Among these BoW-based approaches, BM25 [212, 60, 211] has been the most widely adopted, ranking documents by combining term frequency, inverse document frequency, and a document length normalization factor.

These models represent words as independent dimensions in the vector space, without capturing semantic relationships between them. To address this limitation, a foundational development was the introduction of Word2Vec [185], which encodes words as dense, low-dimensional vectors. However, these embeddings are context-independent, assigning the same representation to a word regardless of its surrounding context or syntactic role in a sentence. Modern pre-trained language models, such as BERT [79] and GPT [204], have transformed the field by embedding contextual semantics using transformer architectures [253]. These models leverage multi-head self-attention mechanisms to understand text sequences more deeply, significantly improving the quality of text representations for Semantic Search and Dense Retrieval tasks [61].

The creation of embeddings is outside the scope of this work. Both the proposed model (CoV) and method (DbU-Cloud) are embedding-agnostic and can operate with any type of precomputed embeddings, regardless of their origin. As we show in Section 11.2, higher-quality embeddings tend to yield better ranking performance.

Supervised Document Ranking. Although pooling layers themselves are unsupervised aggregation operations, they are widely used within supervised document ranking models. For example, Dense Passage Retrievers (DPR) [135] independently encode queries and documents into embeddings, ranking documents according to the dot product or cosine similarity of their respective embeddings. Expanding upon this framework, sentence-level transformers such as Sentence-BERT (SBERT) [209] adapt BERT by incorporating a pooling layer, thereby generating sentence embeddings specifically fine-tuned for tasks involving semantic similarity. SBERT has served as a foundational architecture for numerous state of the art models, evolving into a comprehensive library hosting a wide range of pre-trained and fine-tuned models for sentence and document embeddings[1].

Among other notable supervised approaches, ColBERT [139] retains token-level embeddings and applies a late interaction mechanism between queries and documents using the MaxSim operator. Unlike DPR and SBERT, which compress documents into a single vector, ColBERT compares each query token to all document tokens. However, it operates in a supervised setting and thus requires task-specific fine-tuning.

[1] https://sbert.net/

While effective, *all of the methods discussed above are supervised* and typically require substantial amounts of labeled data, which can be costly to obtain and challenging to scale across diverse information retrieval tasks. In contrast, unsupervised approaches operate without annotated relevance judgments, making them more adaptable, cost-effective, and particularly suitable for document ranking in low-resource or domain-specific scenarios.

In this work, we focus on an unsupervised setting; thus, while supervised document ranking methods are out of scope here, we discussed them for completeness and for the following reasons.

As an unsupervised method, DbU-Cloud operates in a fundamentally different setting, and direct comparisons with supervised approaches are therefore out of scope. However, in our experimental analysis, we leverage embeddings produced by supervised models such as fine-tuned DPR and a state of the art SBERT derivative. Notably, despite these models being fine-tuned specifically for mean pooling-based aggregation, we demonstrate that using their token-level embeddings within DbU-Cloud significantly improves document ranking performance. This highlights that the benefits of advanced supervised pretraining can be effectively harnessed within purely unsupervised ranking pipelines, without any task-specific fine-tuning.

Unsupervised Document Ranking. Relatively little research has addressed Unsupervised Document Ranking. For example, Zhang et al. propose distribution-based models such as the Spherical Paragraph Model [278, 277], which leverage word embeddings in a probabilistic framework to represent entire documents. While effective in capturing semantic structures without supervision, these methods operate as standalone probabilistic models rather than components within LLMs. Consequently, they are not intuitively plug-and-play for most transformer architectures, as they require separate training and inference procedures outside of the LLM.

Mei et al. [181, 76] introduced probabilistic box embeddings, representing queries and documents as high-dimensional hyperrectangles to capture semantic diversity and uncertainty, improving effectiveness and indexing efficiency in document ranking tasks. However, they rely on external learning objectives and constraints tailored to box-based representations rather than being embedded as internal components of transformer models.

Despite the development of such alternative approaches, their lack of plug-and-play compatibility with transformer architectures has hindered widespread adoption. As a result, the predominant strategy for leveraging pre-trained large language models in Document Ranking remains the use of simple pooling methods—most commonly mean or max pooling—as default aggregation mechanisms. This is corroborated by the fact that the vast majority of models available on HuggingFace [125] and Sentence-Transformers [209] use mean pooling as default, despite its drawbacks recognized in the literature. For example, Tu et al. [248] showed that the representation quality of single fixed-width tensors decreases as text length increases. Thakur et al. [243] demonstrated challenges in zero-shot domain adaptation for fine-tuned models, while Xu et al. [266] and Anand et al. [8] emphasized the difficulty of explaining retrieval results when using dense embeddings aggregated into single vectors.

Unlike pooling-based methods that collapse embeddings, DbU-Cloud preserves token-level granularity and rewards local embedding density alongside similarity. Built on the CoV framework, it enables fine-grained, unsupervised ranking in a plug-and-play manner with existing embeddings as those obtained through pre-trained LLMs.

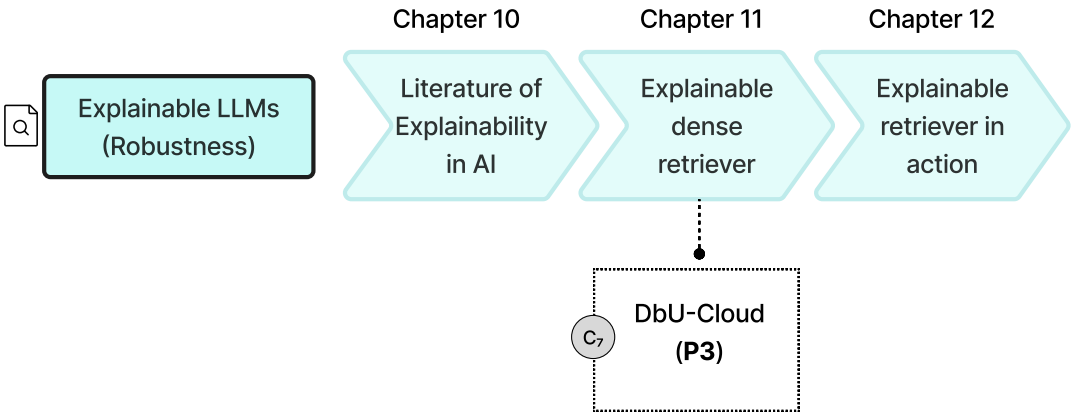


FIGURE 10.2: The research process we followed for the topic of Explainability, related to the TD of Robustness. Each step, where applicable, is annotated with the respective contribution of this thesis.

Performance Considerations in Dense Document Ranking. While efficiency optimization lies beyond the scope of this work, which primarily focuses on introducing a novel model, we acknowledge prior research investigating strategies to accelerate computation in dense document ranking tasks leveraging dense embeddings [34, 168]. Notably, our approach exhibits memory and computational complexity comparable to that of ColBERT [139] and can similarly take advantage of a wide range of established efficiency-enhancing techniques.

10.3 Our Contribution

Figure 10.2 illustrates our research trajectory within the broader theme of Robustness for Trustworthy AI. Among the various subtopics, we identified explainability as the most relevant direction for addressing the technical dimension of robustness. The body of work on explainability is extensive, far more mature than the corresponding literature on Machine Unlearning (see Section 2) or Fairness (see Section 7). This motivated our decision to focus on explainability as a foundation for developing robustness-oriented methods in this thesis.

Specifically, in this work, we study LLM retrievers for information retrieval, and how token-level dense representations can outperform pooling-based document ranking approaches in the unsupervised setting, and how this choice can result in more explainable results.

To address this, we propose the *Clouds of Vectors (CoV)* as a novel reference model for representing both queries and documents. CoV treats the document as a set of dense token embeddings, preserving the semantic richness and contextual fidelity of individual term embeddings, mitigating the information loss inherent in aggregation-based approaches. Briefly, CoV offers a fine-grained, non-aggregated representation while leveraging dense embeddings to encode contextual meaning more effectively.

We then define the DbU Relevance Score, a novel unsupervised scoring function built on CoV that exploits token-level embedding. Unlike traditional approaches, DbU avoids embedding pooling and instead introduces a density-based token-level scoring function that quantifies the relevance of a document with respect to a query. As such, it simultaneously accounts for document-query proximity and the local density of semantically meaningful embeddings, ensuring that relevant documents are not only close to the query in embedding space but also rich in contextually

aligned content. The combination of the Clouds of Vectors (CoV) representation and the DbU Relevance Score constitutes DbU-Cloud, a fully unsupervised document ranking approach that consistently achieves high-quality performance while eliminating the need for any additional fine-tuning.
One of the main barriers to adopting state of the art unsupervised document ranking methods [276, 181, 76] is the lack of plug-and-play compatibility with existing LLMs. We address this issue directly: by replacing the pooling layer of LLMs, DbU-Cloud takes full advantage of the rich term-interaction representations learned by the LLM during pretraining, without further adaptation efforts.

> Pooling methods are inherently unsupervised, as they aggregate embeddings without the need for labeled data. Similarly, our method remains fully unsupervised, focusing on plug-and-play integration with LLMs rather than supervised fine-tuning. As such, **direct comparison to supervised ranking models remain out of scope for this study**. However, since most of these supervised models are still able to produce embeddings, we employ them as direct input to DbU-Cloud and show that, even though they were fine-tuned for mean pooling, our method still outperforms both Mean Pooling (MoE) and Max Pooling (XoE).

We assess the effectiveness of DbU-Cloud approach by benchmarking it against baseline methods employing mean pooling and max pooling of embeddings. Our experiments, conducted across five distinct LLMs and four linguistically diverse corpora, consistently demonstrate that DbU-Cloud outperforms these baselines across all evaluation metrics. The comparison with LLM-based baselines serves to quantify the performance gains introduced by our method. Additionally, we include BM25 — a classic Bag-of-Words model frequently used in the literature — as a point of reference, offering a meaningful contrast from a fundamentally different retrieval paradigm. Moreover, we provide empirical anecdotes showcasing the superiority of DbU-Cloud approach with respect to state of the art approaches. Finally, we provide an ablation study on the impact of the embedding sizes on DbU-Cloud.
The contributions of this study can be summarized as follows:

- In unsupervised document ranking, we discuss the limitations of conventional pooling strategies and advocate for alternative approaches that more effectively preserve the contextual richness and structure captured by LLM embeddings.
- We define a novel reference model - namely *Clouds of Vectors (CoV)* - for both queries and documents, able to better preserve the rich semantics and contextual integrity of term embeddings introduced by the adoption of the LLMs;
- We propose the DbU relevance score as the foundation of our novel *Cloud Density-Based Unsupervised ranking approach* for document ranking that effectively leverages the intrinsic knowledge encoded in LLMs, incorporating the complete algorithmic framework required for relevance estimation.
- We conduct an extensive evaluation of the proposed approach under various conditions, highlighting both its advantages and disadvantages. Our evaluation encompasses five LLMs and four corpora, for a total of 20 experimental settings. On top of these, we study the impact of the parameter of our method and the impact of embedding size on both our method and the baselines. This

comprehensive assessment demonstrates the method's robustness and effectiveness compared to existing techniques.

- We discuss the limitations and threats to the validity of DbU-Cloud and propose directions for future research to address these challenges and enhance the model's performance.
- We publicly release code to ensure reproducibility and facilitate further studies. The full implementation is publicly available on GitHub[2].

[2] https://github.com/dangeloandrea14/dbu_cloud

Chapter 11

DbU-Cloud: an explainable, density-based dense retrieval

11.1 DbU-Cloud

This Section introduces DbU-Cloud, a novel density-based unsupervised Document Ranking methodology utilizing dense embeddings. Figure 11.1 depicts an high-level view of DbU-Cloud. The process begins by ***(a).*** Inferring embeddings for a document (in blue) and a query (in red) using a large language model (LLM) or any other embedding generation model (see Section 11.1.1). Instead of aggregating these embeddings into a single vector, ***(b).*** We propose the Cloud of Vectors (CoV) framework (Section 11.1.2), which models each document or query as a distribution of embeddings. We define a key property of this framework: its density zones correspond to semantic areas in which the document or query is specialized. Building on this foundation, ***(c).*** Section 11.1.3 introduces the DbU relevance metric, which leverages the local density of all query embeddings with respect to their neighboring document embeddings. Finally, Section 11.1.4 details the complete DbU-Cloud algorithm, integrating all components described in steps (a)–(c).

11.1.1 Preliminaries

In this Section, we provide a formal definition of the Document Ranking problem, how embeddings generation (Figure 11.1a) is usually performed, and review the existing approaches in the literature that we employ as baselines.

Let V be a predefined vocabulary of tokens[1] t_i and let be $\mathbb{C}$ a corpus of documents. We define a document $D = \langle t_1, \ldots, t_l \rangle$ as an ordered sequence of tokens $t_i \in V$. Likewise, a query Q is defined as an ordered sequence of tokens $Q = \langle t_1, \ldots, t_q \rangle$. Each document $D_i \in \mathbb{C}$ has its own length $|D_i|$ (i.e., in general, each document length might differ from the length of the other documents or the length of the queries $|Q|$).

Given a Query Q, the task of Document Ranking involves computing a Relevance Score $sim(Q, D_i) \quad \forall D_i \in \mathbb{C}$. The documents are then ordered based on these scores in descending order of relevance. The relevance score reflects how well a document D_i satisfies the information need expressed by the query Q [15]. In this work, we assume that the embeddings are generated from Large Language Models, as they are the state of the art solutions for embedding generation. Thus, we define a Large Language Model as the function $LLM[\langle t_1, \ldots, t_l \rangle, \Phi] : V^l \rightarrow \mathbb{R}^{l \times d}$ which transforms the input sequence of l tokens (i.e., a document or a query) into their embeddings of size d. Φ represents the parameters of the LLM (i.e., a neural network), and d is the

[1] A token is a sequence of characters. Thus, it can be a word or its fragment, and it can contain punctuation and whitespace. The vocabulary V depends on the used LLM.

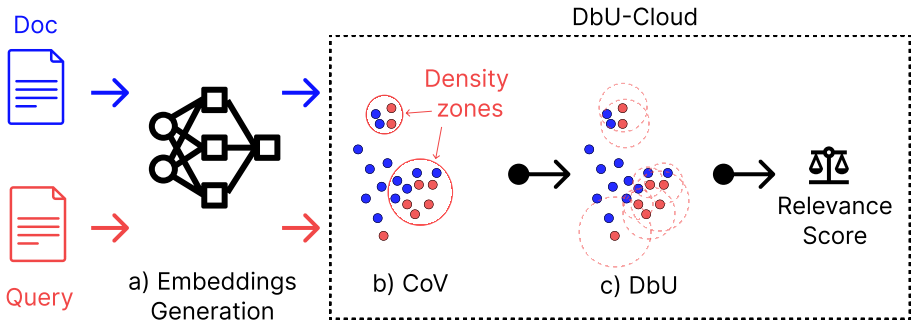


FIGURE 11.1: Overview of DbU-Cloud. (a) A document and a query are each processed through an LLM or any embedding generator to produce their respective sets of dense embeddings. From these two sets, (b) we define the Cloud of Vectors (CoV) framework, and (c) the DbU Relevance Score that leverages CoV. The combination of CoV and DbU is DbU-Cloud.

fixed size of the embeddings, which depends on the hyper-parameters of the chosen LLM.

The reference approach in the literature is to pool the documents' and queries' embeddings into single-vector representations. As such, the Relevance score is

$$sim(agg(LLM(D_i, \Phi)), agg(LLM(Q, \Phi))),$$

where *sim* is a generic similarity function (often the cosine similarity or dot product) and *agg* is a generic aggregation function that pools all the embeddings generated from the LLM into a single representation. When *agg* is the mean function, we refer to the baseline as MoE. When *agg* is the max function, we refer to the baseline as XoE.

11.1.2 Clouds of Vectors Model

This Section formally defines and introduces the novel Cloud of Vectors (CoV) reference model, as seen in Figure 11.1b. This definition outlines the theoretical framework and mathematical foundations underpinning CoV, highlighting how it preserves the rich semantics and contextual integrity of term embeddings.

Introducing this novel reference model represents a paradigm shift in the use of embeddings, laying the groundwork for future algorithms that can more effectively address the query-document ranking task. In the following, we discuss the model's definition (see Definition 7) and the related property (see Property 1).

Our method assumes to have as input an embedding matrix $\mathbf{E}$ computed by an LLM, which we formalize as $LLM[\langle t_1, \ldots, t_l \rangle, \Phi]$. The r^{th} row of $\mathbf{E}$ contains the embedding $\mathbf{e}_r \in \mathbb{R}^d$ of the r^{th} token computed by the LLM. Since we work in an unsupervised setting, the Φ parameters remain unchanged and we assume that the LLM is already pre-trained. Thus, from now on, we can refer to the function $LLM[\langle t_1, \ldots, t_l \rangle, \Phi]$, by omitting Φ, as $LLM[D]$ or $LLM[Q]$ accordingly with the considered input sequence (i.e., D or Q). All the formal notations used in this work are provided in Table 11.1 for reference.

The formal definition of the Cloud of Vectors is introduced as follows:

Notation	Meaning
V	Vocabulary of Tokens
t_i	Token $t_i \in V$
$\langle t_1, \ldots, t_l \rangle$	Ordered sequence of l tokens
$D = \langle t_1, \ldots, t_l \rangle$	Document (made of l tokens)
$Q = \langle t_1, \ldots, t_q \rangle$	Query (made of q tokens)
$\mathbb{C} = \{D_1, \ldots, D_c\}$	Corpus of c documents
$LLM[\langle t_1, \ldots, t_l \rangle, \Phi]$	LLM function
Φ	Parameters of the LLM
e_r	Embedding of the r^{th} token
$\mathbf{E}$	Embedding matrix
d	Size of the embeddings
$\mathcal{C}_\mathcal{D}$	Cloud of Vectors of Document D

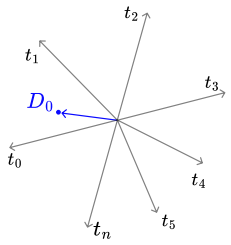
TABLE 11.1: Formal notation used in this work.

Definition 7 (Cloud of Vectors)**.** Given an ordered sequence of l tokens

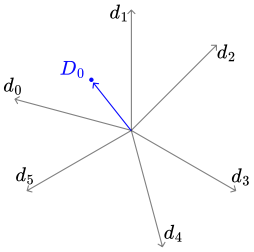
$$D = \langle t_1, \ldots, t_l \rangle$$

, and a function $LLM[\langle t_1, \ldots, t_l \rangle, \Phi] \rightarrow \mathbf{E} : V^l \rightarrow \mathbb{R}^{l \times d}$, a **Cloud of Vectors** $\mathcal{C}_\mathcal{D}$ is the unordered set $\mathcal{C}_\mathcal{D} = \{\mathbf{e}_1, \ldots, \mathbf{e}_l\} \subseteq \mathbb{R}^d$, where the vectors $\mathbf{e}_i$ are able to capture the semantics present in D. We denote the size of a given **Cloud of Vectors** $\mathcal{C}_\mathcal{D}$ as $|\mathcal{C}_\mathcal{D}|$.

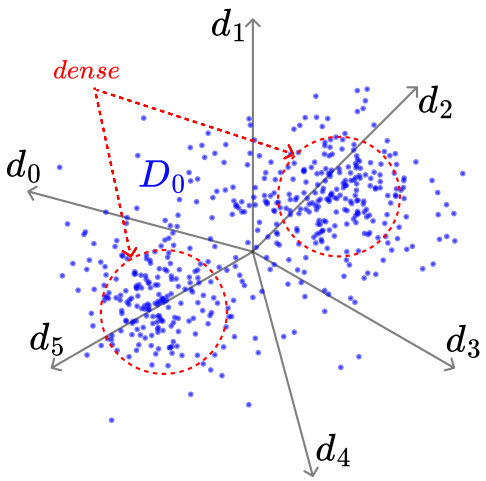
Now that we formally introduced the definition of the Cloud of Vectors model, in Figure 11.2 we visualize its characteristics and its difference with the Vector Space Model (VSM) and the more recent Mean of Embeddings approach.
The Subfigure 11.2a depicts the classic VSM, where documents are represented as only one high-dimensional vector based on the term frequency-inverse document frequency (TF-IDF) [219]. This approach relies on word occurrence statistics and lacks semantic depth, limiting its effectiveness in capturing the text's true meaning.
The second Subfigure 11.2b shows the mean embedding approach, an advancement over VSM. Here, documents are represented by averaging the vectors of individual words using word embeddings. While this method improves semantic representation compared to VSM, it still leads to the loss of individual word semantics by squeezing all the tokens into just one vector.
The third Subfigure 11.2c presents our proposed Cloud of Vectors model. Unlike the previous models, this approach retains the rich semantics produced by large language models (LLMs) by representing documents as a Cloud of Vectors. Areas of high embedding density, highlighted in red, correspond to semantic clusters.
Moreover, as we discussed in Section 10.2, modern LLMs are able to embed the contextual semantics of the tokens, i.e., the embedding of each token of a sequence can be influenced by the embeddings of the other tokens of the same sequence. This ability has a twofold effect. A specific token can have a different embedding in different sequences (e.g., the embedding of the token a in $\langle y, a, g \rangle$ is different from the one that a have in $\langle c, y, a \rangle$, where $a, c, g, y \in V$). On the other hand, during the training phase, the embeddings of the token of a sentence tend to be modified to respect the overall semantics of the sentence (e.g., two synonyms expose similar embeddings if they are replaced in the same sentence). Considering the above capabilities of large language models (LLMs) in conjunction with the introduced Cloud of Vectors model, we can formally present the following property that arises from their co-presence:



(A) Document D_0 represented in the classical *Vector Space Model (VSM)* where the number of the dimension is coincident with $|V| = n$.



(B) Document D_0 represented by its *Mean Of Embeddings* in an embedding space of size $d = 6$.



(C) Document D_0 represented as the *Cloud of Vectors* in an embedding space of size $d = 6$.

FIGURE 11.2: Paradigm Shift in Document Representation visually explained. 11.2a *Classic Vector Space Model*: Documents/Queries are vectors based on TF-IDF in the terms space. 11.2b *Currently used Mean of Embedding model*: Documents/Queries are represented by the average of word embeddings in the embeddings space. 11.2c *Proposed Cloud of Vectors model*: Documents/Queries are represented as a cloud of vectors in the embedding space; semantic dense zones are also highlighted (see Property 1).

Property 1 (Clouds' Densities)**.** If a document D – represented by an ordered sequence of l tokens $\langle t_1, \ldots, t_l \rangle$ – contains a semantic concept expressed multiple times using different tokens and sentences, the Cloud of Vectors $\mathcal{C}_\mathcal{D}$ of D will exhibit a region in the d-dimensional space which exposes a higher density (i.e., areas with higher concentrations of vectors within a given hyper-volume).

The presented Property 1 directly relates to the ability of large language models (LLMs) to encode semantic information. However, this property cannot manifest without adopting the Cloud of Vectors model as the reference framework. In other words, this property is not present or observable if the vectors are not word embeddings and the Cloud of Vectors is not used. To better understand this property, we visualize it by highlighting two denser zones in the illustrative example of Subfigure 11.2c. These examples might correspond to a document that conveys two semantic concepts, for instance, a document discussing the impact of political decisions on the environment contains two main topics: *politics* and *environment*. If these concepts are expressed multiple times with varying nuances, the related Cloud of Vectors will reveal two distinct zones of higher density, as depicted in Subfigure 11.2c, since the Cloud of Vectors $\mathcal{C}_\mathcal{D}$ of D will exhibit two regions in the d-dimensional space with higher density.

11.1.3 DbU Relevance Score

Hereafter, we present the DbU Relevance Score, which, in conjunction with the CoV model discussed in Section 11.1.2 , forms the backbone of the DbU-Cloud methodology (refer to Figure 11.1c). Inspired by the density definition provided in [28], DbU considers the most similar densities of the document to the input query.

The DbU method evaluates document relevance by assessing the degree of isolation of a document's embeddings relative to those of the input query. In contrast to traditional approaches that aggregate information into a single point, DbU assigns a score that captures the extent to which a document aligns with the query. This is achieved by comparing the local density of a query term — estimated based on its similarity to document embeddings — with the densities of neighboring embeddings. Documents that exhibit locally denser regions in proximity to the query are assigned higher scores, indicating higher potential relevance. This method is particularly effective in datasets characterized by clusters of varying densities, where conventional pooling-based mechanisms may fail to accurately capture semantic variance.

The first concept we define is the *akin function*, which is intuitively the similarity between an embedding and the k^{th} most similar vector in a set (according to a given similarity metric). This constitutes the foundational measure of the local neighborhood, dynamically adapting to the density surrounding each point (i.e., embedding). It effectively addresses situations in which multiple neighbors are equidistant and ensures that the resulting neighborhoods accurately capture local density variations.

Definition 8 (Akin function)**.** $akin(e, \mathcal{C}, k) : \mathbb{R}^d \times \mathbb{R}^{l \times d} \times \mathbb{N} \to \mathbb{R}$ is defined as:

$$akin(e, \mathcal{C}, k) = similarity(e, sorted\_set(e, \mathcal{C}, similarity)[k]),$$

where *similarity* is a given similarity function of choice between two embeddings, $sorted\_set(e, \mathcal{C}, similarity)$ are the embeddings of Cloud of Vectors $\mathcal{C}$ sorted in descending order accordingly to their similarity to a given embeddings $\mathbf{e}$, and $[k]$ indicates the k^{th} element of this ordered set.

For example, if we choose *Cosine Similarity* as the similarity metric:
$akin(e, \mathcal{C}, k) = cos(e, sorted\_set(e, \mathcal{C}, cos)[k])$, meaning the *cosine similarity* between $\mathbf{e}$ and the k^{th} most similar vector to $\mathbf{e}$ in $\mathcal{C}$.

Next, we define the *akin neighborhood*, which represents the sub-cloud of vectors we take into account for computing the relevance of the document. The rationale here is to establish the relevant context for comparing e to the document's cloud $\mathcal{C}_D$.

Definition 9 (Akin neighborhood)**.** Let $\mathcal{C}_D$ be the cloud of vectors of a sequence of tokens D, and let $k \leq |\mathcal{C}_D|$ be a natural number. Let $\mathbf{e_i}, \mathbf{e_j} \in \mathbb{R}^d$ be vectors in the same embedding space of $\mathcal{C}_D$. Let $sim(\mathbf{e_i}, \mathbf{e_j}) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}[0, 1]$ be a similarity function between two vectors $\mathbf{e_i}$ and $\mathbf{e_j}$. We define the akin neighborhood $\mathcal{A}(\mathbf{e_i}, \mathcal{C}_D, k)$ of $\mathbf{e_i}$ induced by k in $\mathcal{C}_D$ as:

$$\mathcal{A}(\mathbf{e_i}, \mathcal{C}_D, k) = \{\mathbf{e_j} \in \mathcal{C}_D \setminus \{\mathbf{e_i}\} \mid sim(\mathbf{e_i}, \mathbf{e_j}) \geq akin(\mathbf{e_i}, \mathcal{C}_D, k)\}, \tag{11.1}$$

Note that the size of the akin neighborhood $|\mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k)|$ is strictly greater than k when two or more vectors in $\mathcal{C}_D$ have a similarity with $\mathbf{e_q}$ equal to $akin(\mathbf{e_q}, \mathcal{C}_D, k)$. We now define the *local density* of the cloud of vectors as a measure directly intertwined with the similarities in the akin neighborhood. This prevents embeddings in very dense areas from having overly small influence (i.e., "dominance") in comparisons. It stabilizes comparisons and avoids exaggeration of density for very close pairs, especially in dense regions.

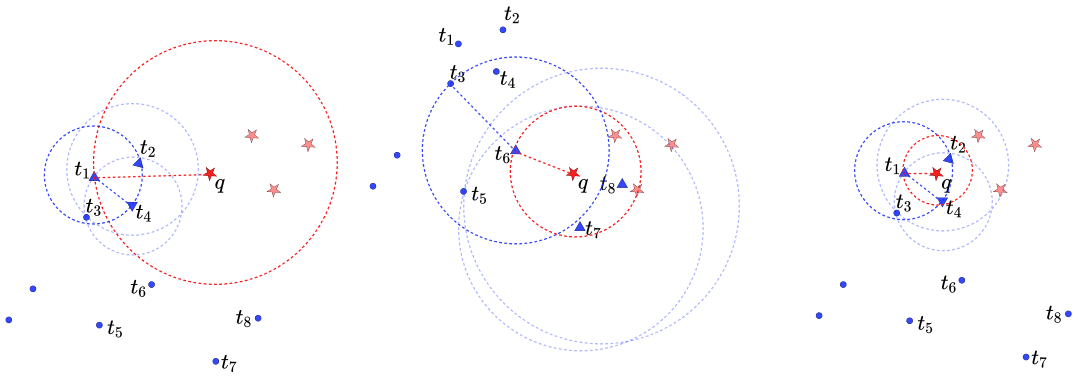
Definition 10 (local density)**.** Let $\mathbf{e_i}, \mathbf{e_j}, \mathcal{C}_D, k, sim(\mathbf{e_i}, \mathbf{e_j})$, and $akin(\mathbf{e_i}, \mathcal{C}_D, k)$ be defined as in Definition 9. The local density, induced by k in $\mathcal{C}_D$, of a vector $\mathbf{e_j}$ with respect to $\mathbf{e_i}$ can be defined as:

$$\text{local-density}(\mathbf{e_i}, \mathbf{e_j}, \mathcal{C}_D, k) = \min\left(\text{sim}(\mathbf{e_i}, \mathbf{e_j}), \text{akin}(\mathbf{e_j}, \mathcal{C}_D, k)\right), \tag{11.2}$$

Based on definitions 9 and 10, we can now define the *DbU* relevance score as the mean *local density*, in the Cloud of Vector $\mathcal{C}_D$ of the document D, of the *akin neighborhood* of the Cloud of Vector $\mathcal{C}_Q$ of the query Q in $\mathcal{C}_D$. Formally:

$$DbU(\mathcal{C}_Q, \mathcal{C}_D, k) = \sum_{\mathbf{e_q} \in \mathcal{C}_Q} \sum_{\mathbf{e_d} \in \mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k)} \frac{\text{local-density}(\mathbf{e_q}, \mathbf{e_d}, \mathcal{C}_D, k)}{|\mathcal{C}_Q| \cdot |\mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k)|}, \tag{11.3}$$

As seen in Equation 11.3, the DbU scoring function does not only depend on the clouds of vectors of query and document, but also by a parameter k, which indicates the k^{th} most similar vector to consider for the *akin* function.



(A) The density of $\mathbf{q}$ is *lower* (bigger red circle) than the density of $\mathbf{t_1}$.

(B) The density of $\mathbf{q}$ is *higher* (smaller red circle) than the density of $\mathbf{t_6}$.

(C) The density of $\mathbf{q}$ is *akin* (comparable circles size) to the density of $\mathbf{t_1}$.

FIGURE 11.3: The density of $\mathbf{q}$ is compared to the density of the akin neighbor $\mathbf{t_i}$ in three prototypical scenarios: 11.3a) lower density, 11.3b) higher density, and 11.3c) akin density. In red and blue are depicted what belongs to the query and to the document, respectively. The clouds of vectors are points, stars, and triangles. The akin neighborhood of $\mathbf{q}$ (i.e., $\mathcal{A}(\mathbf{q}, \mathcal{C}_D, 3)$) are blue triangles, while the dashed circles depict densities.

To better explain how the DbU relevance score works, we refer to the example shown in Figure 11.3, where we depicted the clouds of vectors of a query $\mathcal{C}_Q$(in red stars) and of a document $\mathcal{C}_D$ (all the blue dots and triangles). The densities are depicted as circles for the sake of visual clarity: i.e., a bigger circle depicts a lower density. We discuss the DbU relevance for the query vector $\mathbf{e_q}$ by fixing $\mathcal{C}_Q$ in the embedding space and translating $\mathcal{C}_Q$ in three prototypical scenarios (note that both the clouds $\mathcal{C}_Q$ and $\mathcal{C}_D$ have the same internal structure and scale). To not overload the notation of the picture, we adopted the following convention: $\mathbf{e}. \rightarrow \cdot$, e.g., $\mathbf{e_q} \rightarrow \mathbf{q}$. Accordingly to equation 11.3, the final score depends on the akin neighborhood $\mathcal{A}(\mathbf{q}, \mathcal{C}_D, 3)$, depicted with the blue triangles in the Figure (note that the other query vectors are not part of it). Each vector in the akin neighborhood of $\mathbf{q}$ exposes its local density (the blue dashed circles) that will be compared with the one of $\mathbf{q}$ (e.g., the red dashed circle).

In the first scenario(see Fig. 11.3a), **q** has a lower density (i.e., bigger circle) w.r.t. the density of the akin neighbor. Therefore, the document is less relevant for that query, and the akin neighbor's local density is not considered (see eq. 11.2). On the opposite, if the density of the query vector is higher than the density of the akin neighbor (see Fig. 11.3b), it means that the document is potentially relevant but does not discuss enough of the required information. Ideally, we want the document and query densities to be high to the same extent (see Fig. 11.3c). The latter scenario happens when the document is in the proximity of the query and discusses the required information enough.
Given these observations, DbU-Cloud is designed to reward both semantic proximity and local density, assigning higher scores to documents whose term embeddings not only align with the semantic region of the query but also cluster densely within that space.

11.1.4 DbU-Cloud Relevance Score Assignment

This section discusses the implementation of the relevance score assignment made by DbU-Cloud, as defined in Algorithm 4. Prior describing the computation of the relevance score in detail, we outline the assumptions regarding computational efficiency, as summarized in the following box.

Offline precomputation. This box highlights the offline implementation assumption designed to optimize the computational efficiency of the query-document ranking procedure described in Algorithm 4.
(i) The cloud of vector $\mathcal{C}_D$ of each document D has been precomputed offline prior to the execution of the relevance score. As such, we do not explicitly report this transformation in the Algorithm 4.
(ii) The implementation of the *akin* function leverages a precomputed tensor — constructed offline prior to relevance estimation — that encodes the pairwise cosine similarities among the vector representations of each document within the corpus $\mathbb{C}$.

Given a pre-trained LLM, a corpus $\mathbb{C}$ of documents, a query Q, and an integer parameter k, the output is the relevance score computed by DbU-Cloud for each document in the corpus related to the query.
The algorithm leverages the function $akin(\mathbf{e}, \mathcal{C}, k)$ that, given an embedding $\mathbf{e}$, a cloud of vectors $\mathcal{C}$ and a parameter k, computes — accordingly to Definition 8 — the similarity between $\mathbf{e}$ and each vector in $\mathcal{C}$, sorted in descending order.
Specifically, at Line 1 of Algorithm 4, the cloud of vectors $\mathcal{C}_Q$ of the query Q (see Definition 7) is generated through the given LLM.
Line 4 defines the loop for each embedding e_q in the query's Cloud of Vectors $\mathcal{C}_Q$. Due to the length of queries, the number of embeddings involved is inherently limited, making their computation practically efficient.
At Line 5, we compute the *akin* neighborhood (refer to Definition 9) for e_q against the Cloud of Vectors of document D, namely C_D. Here, the *akin* function is computed between the embedding of the query and the embeddings of each document in the corpus. While this step can be computationally intensive, it can leverage several known speed-up strategies from the literature, which are out of scope for this work.
The *local-density* (see definition 10) and the relevance score (see equation 11.3) are respectively computed at lines 7 and 8. The final relevance score that the document D ($\mathcal{C}_D$) has w.r.t. the query Q ($\mathcal{C}_Q$) is assigned at line 16. The final ranking of the

Algorithm 4 DbU-Cloud relevances' assignment

GIVEN: a $LLM(\cdot)$, a Corpus $\mathbb{C}$, a query Q, and the k parameter.
OUTPUT: $DbU(\mathcal{C}_Q, \mathcal{C}_D, k) \forall D \in \mathbb{C}$

1: $\mathcal{C}_Q := LLM(Q)$
2: **for all** $D \in \mathbb{C}$ **do**
3: $score_{QD} := 0$
4: **for all** $\mathbf{e_q} \in \mathcal{C}_Q$ **do**
5: $\mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k) := \{\mathbf{e_d} \in \mathcal{C}_D \mid sim(\mathbf{e_q}, \mathbf{e_d}) \geq akin(\mathbf{e_q}, \mathcal{C}_D, k)\}$
6: **for all** $\mathbf{e_d} \in \mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k)$ **do**
7: $local{-}density(\mathbf{e_q}, \mathbf{e_d}, \mathcal{C}_D, k) := \min\big(sim(\mathbf{e_q}, \mathbf{e_d}),\ akin(\mathbf{e_d}, \mathcal{C}_D, k)\big)$
8: $score_{QD} : += \frac{local{-}density(\mathbf{e_q}, \mathbf{e_d}, \mathcal{C}_D, k)}{|\mathcal{C}_Q| \cdot |\mathcal{A}(\mathbf{e_q}, \mathcal{C}_D, k)|}$
9: **end for**
10: **end for**
11: $DbU(\mathcal{C}_Q, \mathcal{C}_D, k) := score_{QD}$
12: **end for**
13: **return** sort_desc($\{ DbU(\mathcal{C}_Q, \mathcal{C}_D, k) \mid D \in \mathbb{C} \}$)

relevance scores for all the documents is returned, at line 13, in descending order (i.e., higher values indicate a more relevant document).

11.2 Results and Experimental Evaluation

This section outlines in Section 11.2.1 the experimental protocol and presents a detailed evaluation of the proposed method, DbU-Cloud, highlighting both its strengths and limitations. Specifically, the analysis will be discussed from Section 11.2.2 on, by first examining the sensitivity of DbU-Cloud to the hyperparameter k, as defined in Equation 11.3. In Section 11.2.3, we evaluate the effectiveness of DbU-Cloud in comparison to the baselines—MoE and XoE—across all combinations of LLMs and corpora, following the workflow illustrated in Figure 11.4. These results are consolidated and interpreted in Section 11.2.3. Finally, Section 11.2.4 completes the evaluation by comparing DbU-Cloud to BM25 and conducting an ablation study to assess the effect of embedding dimensionality on performance across all tested models.

11.2.1 Experimental Protocol

Section 11.2.1 introduces the evaluation workflow applied consistently across all experiments, involving DbU-Cloud and baseline models on selected corpora and language models. Sections 11.2.1 and 11.2.1 describe the datasets and large language models (LLMs) used in the study. Section 11.2.1 presents the baseline methods considered for comparison, while Section 11.2.1 defines the evaluation metrics adopted throughout the analysis.

Workflow of the Experiments

Figure 11.4 provides the Experimental Workflow we applied during the experiments to a given corpus and a given LLM. The corpus and the LLM are chosen from a set of selected corpora and LLMs discussed in the Sections 11.2.1 and 11.2.1 respectively. Once the methods (DbU-Cloud and the baselines) were run, their results are evaluated accordingly to the measures discussed in Section 11.2.1.

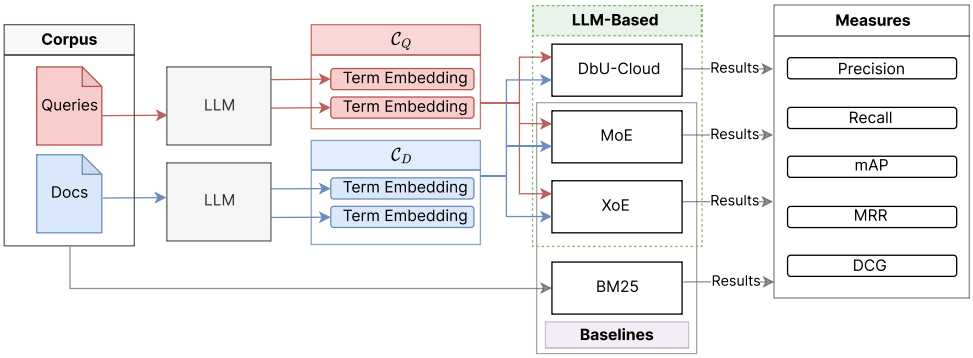


FIGURE 11.4: Experimental workflow adopted in the experimental analysis.

For any corpus (detailed in Section 11.2.1) we consider their documents and queries. We proceed by consistently transforming both documents and queries of each experiment using the chosen LLMs (see Section 11.2.1).

Considering a couple of queries/documents (the pairs are depicted in Figure 11.4 by using red for queries and blue for documents), the output of the transformations are two corresponding Clouds of Vectors, $\mathcal{C}_Q$ for the queries and $\mathcal{C}_D$ for the documents. These clouds will be the input for DbU-Cloud method and the other LLM-based baselines (presented in Section 11.2.1). Finally, to assess their effectiveness, the Ranked lists of documents generated by each baseline are evaluated according to benchmarking measures (see Section 11.2.1), namely Precision, Recall, mAP, MRR and DCG.

All of the experiments we performed are listed in Table 11.2, along with a short description and the Section where they are illustrated.

In the following Sections, we discuss all the Corpora (in Section 11.2.1), the LLMs (in Section 11.2.1), the baselines (in Section 11.2.1), and the benchmarking measures we will use for the comparison (in Section 11.2.1).

Corpora

To evaluate the robustness of DbU-Cloud in different linguistic conditions, we purposefully chose corpora that differ in size (i.e., the number of documents), average documents and query length, content, and language usage, varying from everyday language to specialized medical terminology. Specifically, the CISI[2] corpus uses specialized scientific terminology, while the LISA[2] corpus adopts colloquial language. Both corpora were made by the University of Glasgow and are publicly accessible.

The NFCORPUS[3] was presented as a full-text corpus for Medical Information Retrieval [26], while MS_MARCO[4][58] is a large vernacular corpus widely used in the literature as the reference one for passage retrieval and semantic search. MS_MARCO and NFCORPUS are also available in widely used the ir-datasets library[169]. To cope with the large dimension of MS_MARCO, we uniformly sampled 85 queries from the original corpus. For each query, we uniformly sampled 100 relevant documents, ensuring that each document is relevant for only a query. Then, we sampled 5% of the remaining corpus, ensuring that the chosen documents are not relevant to any query, to finally obtain a corpus of 229,245 documents in total. Similarly,

[2]CISI and LISA are available at https://ir.dcs.gla.ac.uk/resources/test_collections/
[3]The NFCORPUS is available at https://www.cl.uni-heidelberg.de/statnlpgroup/nfcorpus/
[4]MS_MARCO is available at https://microsoft.github.io/msmarco/

Experiment	Section	Description
Study on the parameter k of DbU-Cloud	11.2.2	We employ DbU-Cloud with different parameters k in the range {1,2,3,4,5} to determine the best one in terms of effectiveness.
LLM and Corpus Linguistic Influence Analysis	11.2.3, 11.2.3, 11.2.3, 11.2.3, 11.2.3, 11.2.3	For each LLM (BERT, GPT, ALL-MPNET, Distil- RoBERTa, and DPR), we test each corpus to analyze the effect of different linguistic characteristics on DbU-Cloud and the baselines.
Comparison with BM25	11.2.4	We compare the effectiveness of DbU-Cloud with the best-performing LLM against the classical approach BM25 across all corpora.
Impact of Embedding Size	11.2.4	*We study the impact of the embedding size on DbU-Cloud and the baselines by comparing effectiveness with BERT and BERT-large across all corpora.*
Anecdotical Analysis	11.3	We visualize and discuss the individual term embeddings that compose the query and the top-ranked document of examples from real document ranking scenarios.

TABLE 11.2: List of the experiments we performed for our evaluation.

we sampled the 5% of NFCORPUS queries to make their amount of the same order as the others. The number of documents, queries, and general information of all the corpora adopted for the experimental analysis are reported in Table 11.3. In the table, we employ the same notation introduced in Section 11.1. Namely, $|\mathbb{C}|$ is the number of documents, and $Avg(|D_i|)$, $Avg(|Q_i|)$ are the average document and query size, respectively, for each corpus.

Large Language Models

To evaluate the variation of performances and the impact that the adopted Large Language Model (LLM) might have on the baselines, we selected five LLMs with different characteristics. The details of the LLMs are reported in Table 11.4 where the first column reports the specific LLM with relative reference, the second column reports the dimension of their embeddings, and the last column provides a short description.

BERT-base and GPT were chosen as the most widely known large language models. They differ in terms of the training schema and training dataset. Other than their popularity, we also wanted to challenge the assumption that these LLMs, without the pre-fine-tuning stage, cannot produce embeddings that are usable for the Semantic Search task (which is the closest NLP task to ours). The three other LLMs, namely ALL-MPNET, Distil-RoBERTa, and DPR are employed in their fine-tuned

Corpus	$\lvert\mathbb{C}\rvert$	$\lvert$**Queries**$\rvert$	$Avg(\lvert D_i\rvert)$	$Avg(\lvert Q_i\rvert)$	Linguistic details
CISI[2]	1,440	76	784.76	454.28	Documents and queries containing specialized scientific terminology from Computer Science literature.
LISA[2]	6,000	35	591.03	395.26	It contains documents and queries in colloquial (vernacular) language.
MS_MARCO [4]	229,245	85	335.71	37.80	It contains documents and queries in colloquial (vernacular) language.
NFCORPUS[3]	5,371	75	1518.61	49.16	Documents use specialized medical language and terminology. The queries are in colloquial language.

TABLE 11.3: Corpora used in the experimental analysis. The number of documents $\lvert\mathbb{C}\rvert$ and of the queries $\lvert$**Queries**$\rvert$, the average length of documents $Avg(\lvert D_i\rvert)$ and queries $Avg(\lvert Q_i\rvert)$, and their linguistic details are reported.

state for Semantic Similarity [5]. ALL-MPNET, in particular, is the best-performing fine-tuned Sentence-transformer.
The main difference between DPR and the other Sentence-transformers is that the latter uses a Siamese architecture[209] for training and inference, whereas DPR does not. All the chosen LLMs are employed in the analysis without applying any further fine-tuning supervised or semi-supervised stage. Moreover, all the chosen LLM outputs are of the same embedding size. This was done intentionally to eliminate an additional layer of complexity from the comparison and ensure a consistent embedding size across LLMs by standardizing the representation space. For thoroughness, we specifically investigated the impact of varying the embedding space using BERT-large in Section 11.2.4.

Baselines

To conduct a fair evaluation, we will compare DbU-Cloud with two baselines belonging to the LLM-based method category as DbU-Cloud. These baselines are based on the reference aggregation strategies available nowadays in the literature for embeddings within the settings of sentence transformers. In particular, we considered the *Mean of Embeddings* (MoE) baseline, which ranks the documents accordingly with the Relevance Score obtained by the cosine-similarity between the vectors of the mean of the embeddings of the document and the ones of the query. Likewise, the *maX of Embeddings (XoE)* baseline ranks the documents accordingly with the Relevance Score obtained by the cosine-similarity between the element-wise max embedding vector of the document and of the query. Note that, we kept the LLMs, the embeddings, and the employed corpora consistent in all the experiments. In addition, in Section 11.2.4, we test the widely used *BM25*[212, 60, 211] baseline to have a good indication of the performance of the standard methodologies of Information Retrieval w.r.t. DbU-Cloud.

Benchmarking Measures

Hereafter, we report the most widely used measures in the literature that we employed in the experimental analysis. Precisely, for each experiment, we evaluated the result set of each method accordingly with the Precision (at different depths:

[5] The fine-tuned models, including the description of their training process and training datasets, are available at `https://huggingface.co/sentence-transformers`.

LLM	d	Description
BERT-base [79]	768	Pre-trained standard BERT model for inference.
GPT [204]	768	Pre-trained standard GPT model for inference.
ALL-MPNET[5]	768	State-of-the-art LLM for the sentence-based semantic search, on which it is fine-tuned. It is the best-performing version of SBERT to date. It uses Mean pooling as default.
Distil-RoBERTa[5]	768	Distil-RoBERTa fine-tuned on multiple corpora for semantic search. It uses Mean pooling as default.
DPR[5]	768	DPR Model implemented in the sentence-transformer library.

TABLE 11.4: LLMs used in the experimental analysis with their description and the dimensionality of their embeddings d.

1,5,10,15,30), the Recall (@10), the Mean Average Precision, the Mean Reciprocal Rank, and the Discounted Cumulative Gain (@10) following the typical strategy of aggregating the results across the queries. In particular, the two most classical measures used in the literature are the *Precision* and the *Recall* (see [244] for their definitions). By considering a cut-off value Z, precision, recall and DCG can be defined with respect to the top-Z documents in the ranked list. We will refer to these measures as *P@Z* and *R@Z* (note that we avoid naming them with their typical literature name, P@K, and R@K, to avoid confusion with DbU-Cloud's k parameter).
The Mean Average Precision (mAP) is defined as:

$$mAP = \frac{\sum_{q=1}^{|\mathbf{Q}|} \text{Average\_Precision(q)}}{|\ \mathbf{Q}\ |} \tag{11.4}$$

where $|\mathbf{Q}|$ is the number of queries. The mAP represents the mean of the average precision scores for each query.
The Mean Reciprocal Rank MRR, in contrast to precision, assigns more weight to relevant results that were listed higher in the returned documents. Formally:

$$MRR = \frac{1}{|\ \mathbf{Q}\ |} \sum_{i=1}^{|\mathbf{Q}|} \frac{1}{rank_i} \tag{11.5}$$

Discounted Cumulative Gain (DCG), it is similar to the MRR, but instead of just boosting the score of the highly-ranked relevant documents. This measure also penalizes the low-ranked relevant documents by reducing the scores logarithmically accordingly to the position of the document in the result set. The DCG base its score on the position p, thus it can be defined as:

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)} \tag{11.6}$$

where rel_i is either 1, if the document is relevant, or 0 otherwise.
Moreover, DbU-Cloud must use a similarity measure (by the Definition 9). Thus, in the experimental analysis, we used the *Cosine Similarity* because it is the reference measure in the literature for similarity computation between embeddings. Formally

	BERT base				OpenAI GPT				ALL-MPNET				DistilRoBERTa				DPR			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
$k=1$	0.100	0.066	0.466	1.270	0.072	0.047	0.365	0.925	0.218	0.096	0.630	1.733	0.198	0.112	0.597	1.760	0.145	0.126	0.516	1.343
$k=2$	0.078	0.039	0.340	0.866	0.065	0.044	0.357	0.873	0.218	0.095	0.629	1.734	0.200	0.111	0.601	1.761	0.120	0.112	0.490	1.252
$k=3$	0.060	0.037	0.291	0.778	0.058	0.042	0.343	0.831	0.215	0.094	0.621	1.726	0.198	0.109	0.601	1.754	0.108	0.110	0.460	1.138
$k=4$	0.045	0.034	0.270	0.701	0.045	0.040	0.322	0.769	0.210	0.094	0.614	1.728	0.200	0.108	0.609	1.741	0.098	0.094	0.428	1.058
$k=5$	0.040	0.029	0.260	0.635	0.040	0.037	0.314	0.732	0.208	0.094	0.604	1.713	0.198	0.109	0.607	1.736	0.090	0.082	0.393	0.992
UB	0.116	0.084	0.642	1.623	0.100	0.071	0.623	1.427	0.242	0.112	0.784	2.147	0.235	0.132	0.783	2.177	0.175	0.174	0.723	1.808

TABLE 11.5: Analysis of the impact of parameter k on the effectiveness of DbU-Cloud across LLMs. The Upper Bound shows the best possible value obtainable when selecting the best k for each query.

defined as follows:

$$cos\_sim(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{||\mathbf{a}||\ ||\mathbf{b}||} \tag{11.7}$$

Where **a** and **b** are two generic vectors of the same length.

11.2.2 Sensitivity Analysis on parameter k

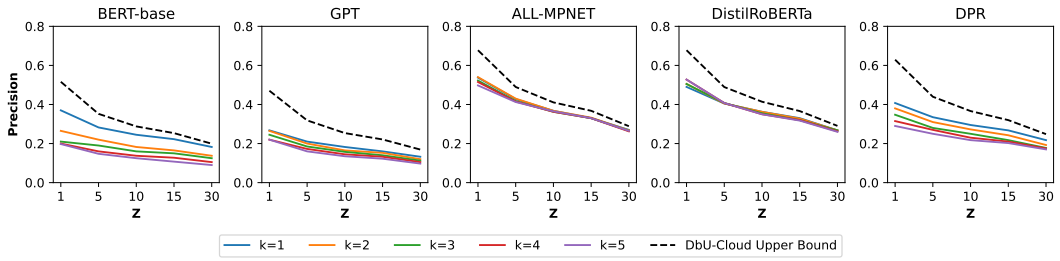


FIGURE 11.5: Analysis of the impact of parameter k on the precision of different LLMs. The values are computed by the mean across all the queries of all corpora. The x-axis depicts the precision thresholds Z.

In this batch of experiments, we test DbU-Cloud on all corpora (see Section 11.2.1) by varying the k in the range $\{1, 2, 3, 4, 5\}$. The aim of this experiment is to determine which value of parameter k performs best on a given LLM. Moreover, to shed light on the potentiality of DbU-Cloud, we introduce the Upper Bound (UB) of DbU-Cloud to show the maximum performance that can be reached hypothetically. The UB is computed by selecting the best k for each query of a certain dataset for each measure. Formally, let $Measure(RankedList)$ be the value of one of the benchmarking measures presented in Section 11.2.1. Then, we define: $UB(Query, Corpus, Measure) = \max_{k \in \{1,2,3,4,5\}} Measure\big(DbU(Query, Corpus, k)\big)$. The UB for any corpus will be the mean value across of all its queries. We report the UB in the future batch of experiments summary to show the performance that DbU-Cloud might hypothetically achieve in that case. Figure 11.5 shows the Precision at depths $Z = 1, 5, 10, 15, 30$ across all corpora for each employed LLM and each k, whereas Table 11.5 depicts MaP, Recall@10, MRR and DCG@10.

Figure 11.5 shows that pre-trained models (BERT, GPT) are more sensitive to the variation of parameter k with respect to fine-tuned models (MPNET, DistilRoBERTa), with the exception of DPR. Specifically, there is an inverse correlation between the constant strategy for parameter k and precision of DbU-Cloud when employing pre-trained models. However, the precision achieved by the UB (depicted in Figure 11.5 as a dotted line) highlights that a constant strategy for k is not optimal for all queries. This suggests that each query should have its own distinct k value to achieve the best results. The values in table 11.5 corroborate these findings. Thus, a possible future

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.020	0.028	0.250	0.438	0.030	0.035	0.172	0.216	0.020	0.012	0.240	0.580	0.000	0.006	0.042	0.049
MoE	0.050	0.050	0.408	0.830	0.060	0.096	0.225	0.501	0.060	0.024	0.454	1.178	0.010	0.014	0.078	0.107
DbU-Cloud	**0.090**	**0.083**	**0.538**	**1.209**	**0.100**	**0.135**	**0.401**	**0.733**	**0.150**	**0.042**	**0.603**	**2.004**	**0.060**	**0.073**	**0.323**	**0.596**

TABLE 11.6: Analysis of the impact of the corpus on the *MaP*, *R@10*, *MRR*, and *DCG@10* of the baselines (MoE, XoE, DbU-Cloud). The values are computed by the mean across all the queries using BERT-base.

direction is to shape a dynamic query-dependent strategy to set the parameter k. Until then, the constant $k = 1$ strategy is the best one for pre-trained models. For this reason, it is the value used in all the experiments pertains to the BERT and GPT. Regarding the fine-tuned models, Table 11.5 and Figure 11.5 show how the sensitivity to parameter $k \in [1, 5]$ is marginal in most cases, as the measures remain stable at all depths Z. The observed stability of DbU-Cloud w.r.t. the parameter k underscores the reliability in real-world unsupervised scenarios where the best k might be unknown. Therefore, adopting a value within this range can serve as a robust and convenient default setting, minimizing the need for extensive hyper-parameter optimization while maintaining consistent precision across various thresholds. For this reason, we employ the median k value (i.e., k=3) for the fine-tuned models during our experimental evaluation.

To sum up, $k = 1$ is the best constant value for pre-trained models in our analyses. While, adopting $k = 3$ for fine-tuned models serves as a robust default setting, minimizing the need for extensive hyper-parameter optimization while maintaining consistent precision. The UB results underscore the potential of DbU-Cloud in both pre-trained and fine-tuned contexts.

11.2.3 Analysis of the impact of the LLM models

In this section, we present the results obtained on each LLM we employ for the experimentation.

Analysis of the impact of BERT

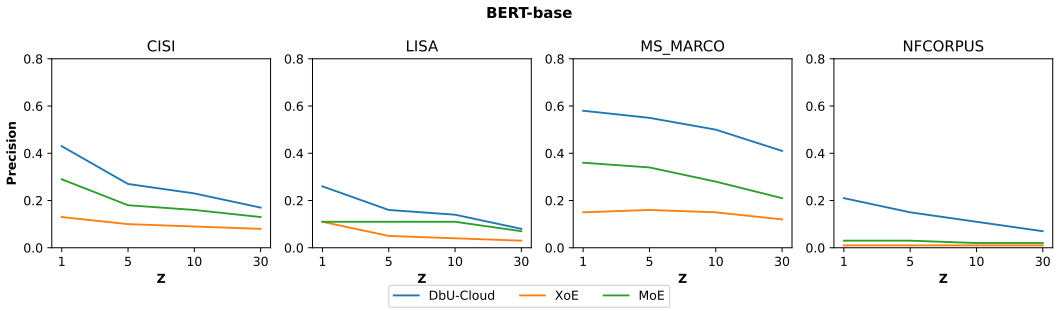


FIGURE 11.6: Analysis of the impact of the corpus on the precision P (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using BERT-base. The cut-off threshold Z is on the x-axis.

Figure 11.6 depicts the precision at different depths Z for all baselines in each corpus when employing BERT-base. Table 11.6 reports the *MaP*, *Recall@10*, *MRR* and

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.040	0.034	0.302	0.636	0.020	0.025	0.130	0.185	0.050	0.020	0.377	0.977	0.020	0.027	0.149	0.244
MoE	**0.070**	0.052	**0.432**	**0.886**	0.060	0.094	0.240	0.499	0.050	0.022	0.414	1.112	0.010	0.024	0.111	0.171
DbU-Cloud	0.060	**0.055**	0.406	0.858	**0.100**	**0.135**	**0.310**	**0.688**	**0.090**	**0.031**	**0.475**	**1.455**	**0.040**	**0.055**	**0.268**	**0.463**

TABLE 11.7: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE), and DbU-Cloud. The values are computed by the mean across all the queries using OpenAI GPT.

DCG@10 for each baseline and corpus. For pre-trained models like BERT, as discussed in Section 11.2.2, we employ k=1 for DbU-Cloud.

DbU-Cloud with BERT significantly outperforms the baselines on all measures and all corpora. This result is consistent regardless of the language features of each individual corpus. However, the best improvement is in MS_Marco, where DbU-Cloud provides a 150% increase on *MaP*, a 29.95% increase on MRR, and a 70.11% increase on DCG@10 with respect to the most effective baseline MoE. However, DbU-Cloud provides vast improvements on other corpora as well, both on *MaP* and DCG@10, implying that DbU-Cloud retrieves relevant documents much more often both within the first results and across the entire ranked list.

Moreover, the results highlight the versatility of DbU-Cloud, showcasing its adaptability to different corpora with varying characteristics. The substantial improvements in *MaP*, MRR, and DCG@10 suggest that DbU-Cloud not only excels in precision but also in the overall ranking and retrieval of documents, contributing to a more comprehensive and accurate retrieval process.

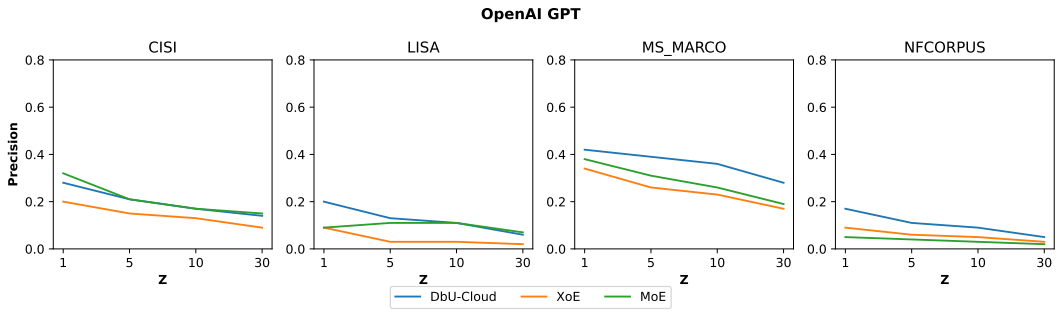


FIGURE 11.7: Analysis of the impact of the corpus on the precision *P* (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using OpenAI GPT. The cut-off threshold *Z* is on the x-axis.

Analysis of the impact of GPT

Figure 11.7 and Table 11.7 depict the selected corpora's precision and other measures when employing GPT. For pre-trained models like GPT, as discussed in Section 11.2.2, we employ k=1. DbU-Cloud outperforms the baseline across all corpora, with bigger improvements in the two datasets available on the most often used in the literature datasets, namely MS_MARCO, and NFCORPUS. DbU-Cloud also outperforms the baselines on corpus LISA, while falling slightly behind the MoE on corpus CISI. However, DbU-Cloud remains the most robust method across all corpora, maintaining consistency across corpora with different language features.

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.150	0.108	0.651	1.613	0.26	0.292	0.532	1.277	0.12	0.041	0.572	1.870	0.11	0.124	0.448	0.930
MoE	0.170	0.124	0.590	1.681	0.300	0.338	0.574	**1.457**	0.12	0.039	0.562	1.806	0.11	0.134	0.473	1.005
DbU-Cloud	**0.200**	**0.127**	**0.657**	**1.874**	**0.340**	**0.364**	**0.598**	1.428	**0.200**	**0.051**	**0.727**	**2.401**	**0.12**	**0.137**	**0.499**	**1.052**

TABLE 11.8: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE), and DbU-Cloud. The values are computed by the mean across all the queries using ALL-MPNET.

Analysis of the impact of ALL-MPNET

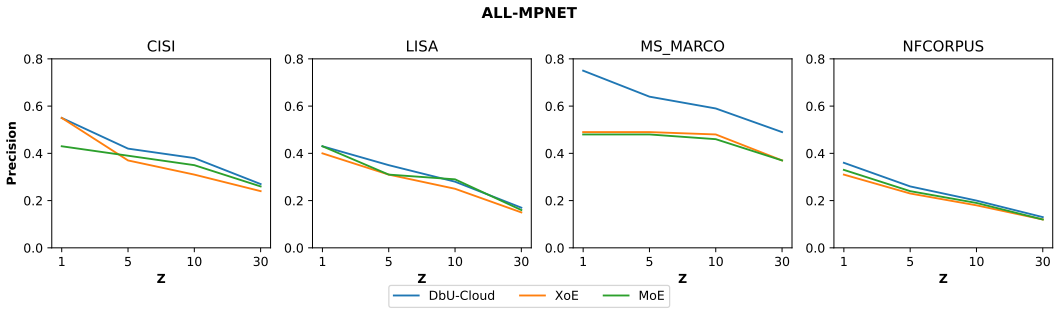


FIGURE 11.8: Analysis of the impact of the corpus on the precision *P* (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using ALL-MPNET. The cut-off threshold *Z* is on the x-axis.

In this Section, we discuss the first considered fine-tuned model, which is ALL-MPNET. As it is possible to notice by Figure 11.8 and Table 11.8, DbU-Cloud confirms the expectation to be, in all the benchmarking measures and, in all the corpora, the best LLM-based unsupervised approach for document ranking when employing ALL-MPNET.

Notice that DbU-Cloud outperforms the baselines on almost all measures and across all corpora, regardless of their specific language features. DbU-Cloud also outperforms the baselines at each Precision depth *Z*, as depicted in Figure 11.8.

In particular, CISI and NFCORPUS present specialized informatics and medical terminology, respectively, with very long queries (please refer to table 11.3). In this setting, the improvement obtained by employing DbU-Cloud is smaller but significant, especially considering the measures of *MaP* and DCG@10, as reported in Table 11.8. LISA, conversely, presents vernacular language but very long queries. DbU-Cloud, leads in *MaP* and MRR, but falls slightly behind MoE in DCG@10. Lastly, MS_MARCO presents vernacular language and shorter queries and is the corpus in which DbU-Cloud has the highest effectiveness. In this setting, DbU-Cloud improves *MaP* by 66.66%, MRR by 27.09% and DCG@10 by 28.39% with respect to the most effective baseline, which in this case is XoE.

These results underscore the practical significance of DbU-Cloud's effectiveness in a setting commonly encountered, for instance, in internet-based Document Retrieval, where shorter queries are matched against longer, generic documents. Moreover, Figure 11.8 shows that DbU-Cloud in MS_MARCO significantly improves on Precision@1, meaning that the first retrieved document is relevant remarkably more often with DbU-Cloud than with the baselines. This is particularly important for Document Retrieval, as end users tend to only interact with the first retrieved documents rather than the full ranked list.

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.140	0.103	0.560	1.401	0.190	0.221	0.480	1.018	0.150	0.046	0.703	2.224	0.100	0.120	0.452	0.931
MoE	0.180	0.133	**0.601**	1.646	**0.260**	0.299	**0.618**	**1.349**	0.170	0.046	0.716	2.242	0.110	0.122	0.463	0.942
DbU-Cloud	**0.200**	**0.147**	0.590	**1.835**	0.250	**0.310**	0.570	1.347	**0.220**	**0.050**	**0.729**	**2.399**	**0.120**	**0.140**	**0.500**	**1.046**

TABLE 11.9: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE), and DbU-Cloud. The values are computed by the mean across all the queries using Distil-RoBERTa.

Analysis of the impact of DistilRoBERTa

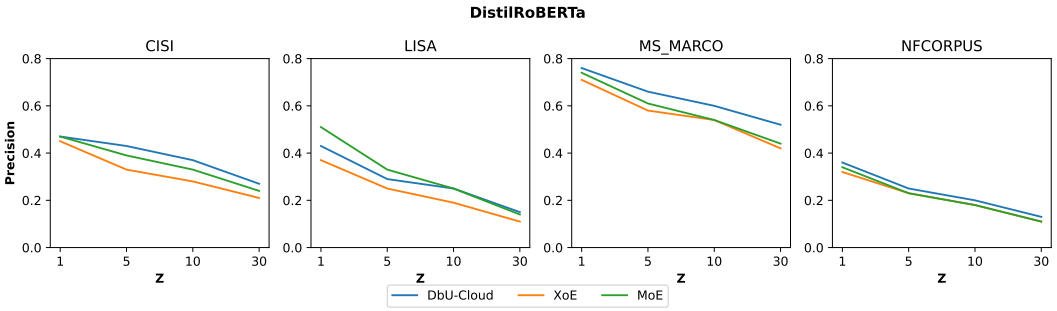


FIGURE 11.9: Analysis of the impact of the corpus on the precision *P* (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using DistilRoBERTa. The cut-off threshold *Z* is on the x-axis.

Figure 11.9 and Table 11.9 show the results when employing the LLM DistilRoBERTa. In this particular setting, DbU-Cloud outperforms all baselines on all other corpora except for LISA and especially on MS_MARCO. Overall, DbU-Cloud is the most consistent approach across all corpora. The most noticeable difference is found, once again, on the MS_MARCO corpus.

Analysis of the impact of DPR

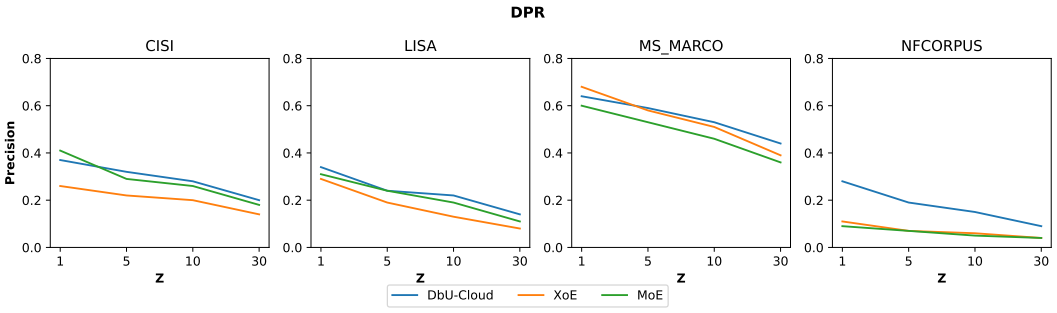


FIGURE 11.10: Analysis of the impact of the corpus on the precision *P* (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using DPR. The cut-off threshold *Z* is on the x-axis.

Figure 11.10 and Table 11.10 show the measures when employing Dense Passage Retriever (DPR). Unlike ALL-MPNET and DistilRoBERTa, DPR does not employ a Siamese architecture, but queries and documents are independently transformed without weight sharing. Despite the slight change in setting, DbU-Cloud remains

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.060	0.054	0.419	0.965	0.130	0.149	0.397	0.758	0.140	0.044	**0.669**	2.126	0.030	0.036	0.188	0.300
MoE	0.100	0.073	**0.552**	1.267	0.200	0.234	0.458	1.004	0.120	0.039	0.612	1.914	0.020	0.038	0.171	0.275
DbU-Cloud	**0.110**	**0.092**	0.499	**1.340**	**0.230**	**0.264**	**0.514**	**1.102**	**0.150**	**0.045**	0.650	**2.137**	**0.090**	**0.101**	**0.403**	**0.793**

TABLE 11.10: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE, DbU-Cloud). The values are computed by the mean across all the queries using DPR.

	BERT base				OpenAI GPT				ALL-MPNET				DistilRoBERTa				DPR			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
XoE	0.018	0.015	0.176	0.355	0.032	0.027	0.240	0.619	0.160	0.082	0.551	1.400	0.145	0.090	0.549	1.519	0.090	0.071	0.418	1.037
MoE	0.045	0.029	0.293	0.705	0.048	0.033	0.298	0.723	0.175	0.087	0.550	1.405	0.180	0.100	0.600	1.610	0.110	0.096	0.448	1.115
DbU-Cloud	**0.100**	**0.066**	**0.466**	**1.270**	**0.073**	**0.047**	**0.364**	**0.925**	**0.215**	**0.094**	**0.622**	**1.727**	**0.198**	**0.109**	**0.601**	**1.754**	**0.145**	**0.125**	**0.517**	**1.343**
DbU-Cloud Upper Bound	0.116	0.084	0.642	1.623	0.100	0.071	0.623	1.427	0.242	0.112	0.784	2.147	0.235	0.132	0.783	2.177	0.175	0.174	0.723	1.808

TABLE 11.11: Analysis of the LLM's impact on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE), DbU-Cloud, and BM25. The values are computed by the mean across all corpora.

the best and most consistent approach overall across all corpora, most notably on NFCORPUS, where it yields a 200% increase in MaP, 180% increase on Recall at 10, 114% increase in MRR and 164.33% increase in DCG@10 concerning the best-performing baseline. DbU-Cloud is consistently able to retrieve relevant documents in the highest positions when employing DPR, as corroborated by the increase in DCG@10 across all corpora.

Comprehensive overview of each LLM's influence across all evaluated corpora

In this setting, we test DbU-Cloud against the baselines Xoe and MoE on each corpus across all LLMs. The experiment follows Section 11.2.1 workflow. For each employed LLM we study the effectiveness (i.e., P@{1,5,10,30}, MaP, R@10, MRR, DCG@10) of DbU-Cloud against the baselines of MoE and XoE on all corpora.
Hereafter, we summarise the impact of each LLM (see Section 11.2.1) across all corpora on the different retrieval approaches (i.e., DbU-Cloud and baselines). Moreover, we report UB as a means of the possible reachable performance of DbU-Cloud.

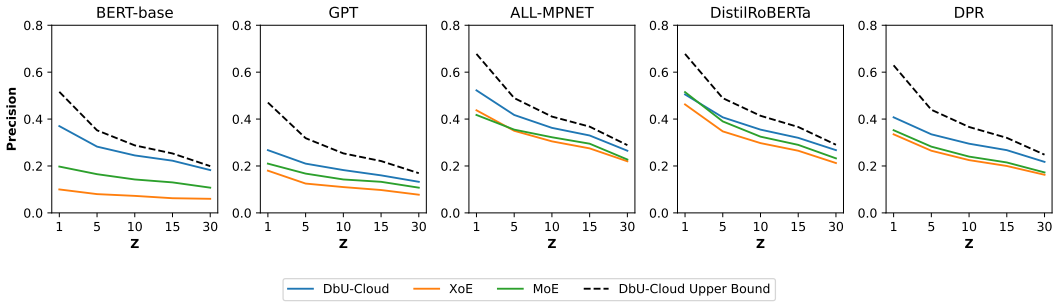


FIGURE 11.11: Analysis of the impact of the LLM on the baselines (Moe,XoE), and DbU-Cloud. In this visualization, the precisions were grouped across all corpora.

Figure 11.11 depicts the trend of the precision, for all the LLMs and all the baselines by the variation of its depth @*Z*, while Table 11.11 reports the *MaP*, the *R@10*, the *MRR*, and the *DCG@10* for all the possible combinations LLM-baseline. The experiment follows the workflow presented in Section 11.2.1, fixing to 1 the k parameter of DbU-Cloud for pre-trained models, and $k = 3$ for fine-tuned models, which regulates the akin neighbourhood and their neighbours (an in-depth analysis of it is conducted in Section 11.2.2). At this stage, we want to capture the general trends

and the impact of the LLMs on the baselines, so we present the aggregation (by the mean) of the benchmarking measures of each corpus.
As it is possible to notice from Figure 11.11 and in Table 11.11, all the LLMs have an impact on the effectiveness of all the LMM-based baselines. In general, XoE is the worst performer among all the baselines using any LLMs, while DbU-Cloud improves every measure on all LLMs. In particular, pre-trained models (namely, BERT-base and OpenAI GPT) improve the most when employing DbU-Cloud instead of a pooling layer. BERT-base, in particular, more than doubles its *MaP* and Recall at 10, while significantly improving MRR and DCG at 10, with respect to the Mean Pooling technique. The precision values depicted in figure 11.11 also denote a significant improvement at all depths Z for pre-trained models.
Focusing now on fine-tuned models (e.g., ALL-MPNET, DistilRoBERTa and DPR), the improvement obtained with DbU-Cloud is smaller with respect to pre-trained models, but still significant across all measures. In particular, MPNET obtains a 20% increase on *MaP*, a 12% increase on MRR, and a 13.5% increase on DCG at 10, with respect to the best-performing baseline, the Mean of Embeddings. Similarly, DistilRoBERTa obtains a 4.7% and 6.14% increase on *MaP* and DCG at 10, respectively, with respect to the Mean of Embeddings.

11.2.4 Comparison with BM25 and Embedding Size Analysis

In this Section, our aim is twofold: *i)* We first compare the effectiveness of DbU-Cloud to a classical Document Retrieval technique, namely Okapi BM25 [212][6]; and *ii)* we then compare the results obtained by employing DbU-Cloud with BERT-base (as we did in our experimental setting, refer to Table 11.4) and BERT-large, whose embeddings have a dimensionality of $d = 1024$ to evaluate the impact of embeddings size on DbU-Cloud measures.

Comparison with Classical Ranking Model

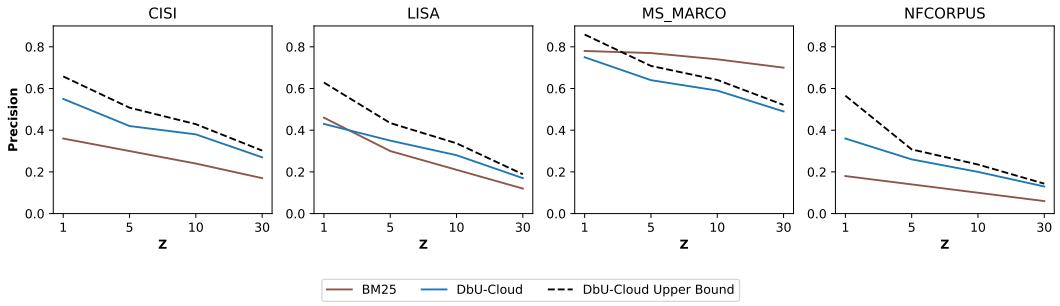


FIGURE 11.12: Analysis of the impact of the corpus on the precision P (y-axis) of DbU-Cloud, its Upper Bound, and the classical retrieval algorithm BM25. The values are computed by the mean across all the queries using ALL-MPNET. The cut-off threshold Z is on the x-axis.

BM25 uses a bag-of-words model with exact matching to compute a query-document Relevance Score and, as such, does not make use of any underlying LLM. It is by far the most popular classical retrieval algorithm.

[6]While effective, BM25 has several limitations: It uses a heuristic saturation function for term frequency, and ignores term context, which can lead to suboptimal rankings. BM25 poorly handles synonyms and polysemy, and its document length normalization may not suit all types.

	CISI				LISA				MS_MARCO				NFCORPUS			
	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
BM25	0.100	0.089	0.524	1.220	0.230	0.257	0.567	1.188	**0.400**	**0.063**	**0.741**	**2.898**	0.050	0.071	0.288	0.541
DbU-Cloud	**0.200**	**0.127**	**0.657**	**1.874**	**0.340**	**0.364**	**0.598**	**1.428**	0.200	0.051	0.727	2.401	**0.12**	**0.137**	**0.499**	**1.052**
DbU-Cloud Upper Bound	0.218	0.144	0.679	1.900	0.396	0.388	0.623	1.443	0.215	0.064	0.919	3.028	0.142	0.160	0.649	1.266

TABLE 11.12: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of DbU-Cloud, its Upper Bound, and the classical retrieval algorithm BM25. The values are computed by the mean across all the queries using ALL-MPNET.

LLM		CISI				LISA				MS_MARCO				NFCORPUS			
		MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10	MaP	R@10	MRR	DCG@10
BERT-base	XoE	0.020	0.028	0.250	0.438	0.030	0.035	0.172	0.216	0.020	0.012	0.240	0.580	0.000	0.006	0.042	0.049
	MoE	0.050	0.050	0.408	0.830	0.060	0.096	0.225	0.501	0.060	0.024	0.454	1.178	0.010	0.014	0.078	0.107
	DbU-Cloud	**0.090**	**0.083**	**0.538**	**1.209**	**0.100**	**0.135**	**0.401**	**0.733**	**0.150**	**0.042**	**0.603**	**2.004**	**0.060**	**0.073**	**0.323**	**0.596**
BERT-large	XoE	0.030	0.028	0.296	0.490	0.060	0.083	0.268	0.331	0.010	0.008	0.205	0.379	0.000	0.004	0.030	0.031
	MoE	0.070	0.065	0.445	1.019	0.130	0.177	0.359	0.753	0.030	0.019	0.410	0.917	0.020	0.021	0.118	0.173
	DbU-Cloud	**0.100**	**0.087**	**0.525**	**1.394**	**0.170**	**0.196**	**0.488**	**0.928**	**0.140**	**0.045**	**0.624**	**2.100**	**0.060**	**0.080**	**0.347**	**0.660**

TABLE 11.13: Analysis of the impact of the corpus on the *MaP*, *R*@10, *MRR*, and *DCG*@10 of the baselines (MoE, XoE), and DbU-Cloud. The values are computed by the mean across all the queries using BERT-base and BERT-large.

Figure 11.12 illustrates the precision at different depths Z, and Table 11.12 presents the *MaP*, R, MRR, and DCG. The comparison between BM25 and DbU-Cloud is performed by the mean across all the queries using ALL-MPNET. In the setting of longer documents and queries, both for vernacular and specialized language (namely CISI, LISA, and NFCORPUS), DbU-Cloud significantly outperforms BM25 on all measures, doubling the MaP on CISI and NFCORPUS. DbU-Cloud also presents a 53.66%, 20.20%, and 94.45% increase on DCG@10 on the corpora CISI, LISA, and NFCORPUS, respectively. However, in the case of MS_MARCO, where BM25 outperforms DbU-Cloud, it's worth noting that this corpus poses unique challenges: MS_MARCO is known for its emphasis on exact term-matching, and the performance of BM25 in this context aligns with its strengths in such scenarios[225]. Nevertheless, our analysis shows that DbU-Cloud significantly narrows the gap between BM25 and plain LLM-based methods, particularly in measures such as MRR and DCG@10, corroborating its competitive effectiveness even in corpora biased towards exact term-matching. When considering its upper bound, the gap between DbU-Cloud and BM25 is even wider, and in the case of MS_MARCO, DbU-Cloud outperforms BM25 on the measures of R@10, MRR, and DCG@10. This implies that, although precision is lower on higher thresholds Z, our approach can retrieve relevant results in the first positions of the ranked list, which has a major impact on document retrieval. Furthermore, to evaluate whether DbU-Cloud can potentially outperform BM25 in the MS_MARCO settings, we present the performance achieved by the Upper Bound. The dotted line in Figure 11.12 highlights that a dynamic k strategy has the potential to surpass BM25 even in this challenging scenario.

Impact of embedding sizes on DbU-Cloud

We now move on to study the impact of embedding sizes on DbU-Cloud. Figure 11.13 depicts the Precision at different thresholds Z of DbU-Cloud and the baselines when employing BERT-base (upper side of the figure) and BERT-large (lower side of the figure). Table 11.13 contains the value of *MaP*, R@10, MRR, and DCG@10 for BERT and BERT-large on each corpus.

It is interesting to note how DbU-Cloud performs best with BERT-large on all corpora. This suggests that the dimensionality of the embeddings is not a concern for the effectiveness of our methods. Moreover, the increase in effectiveness provided

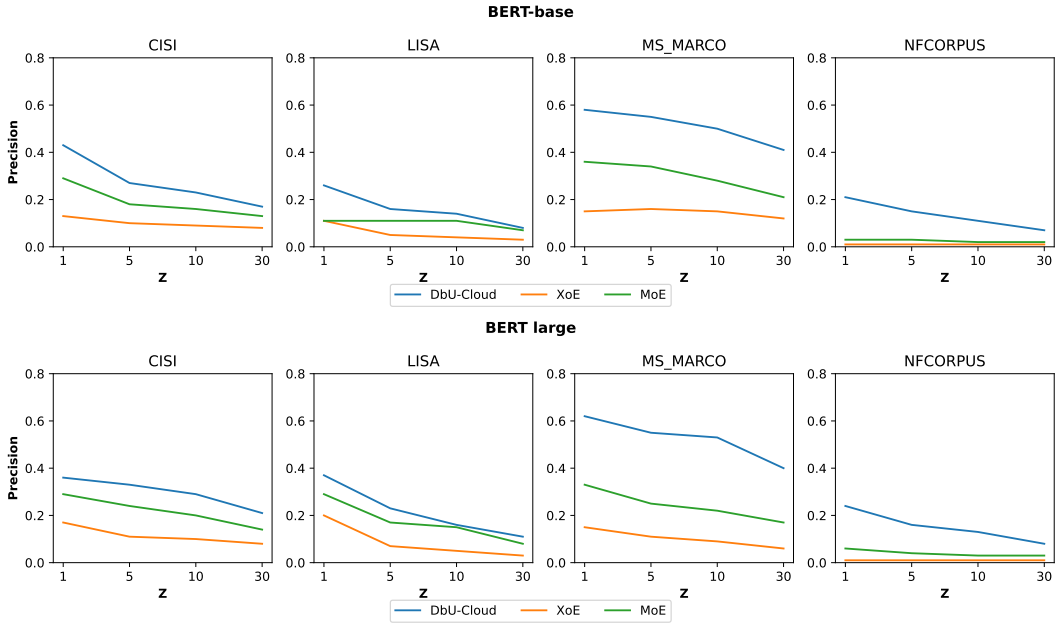


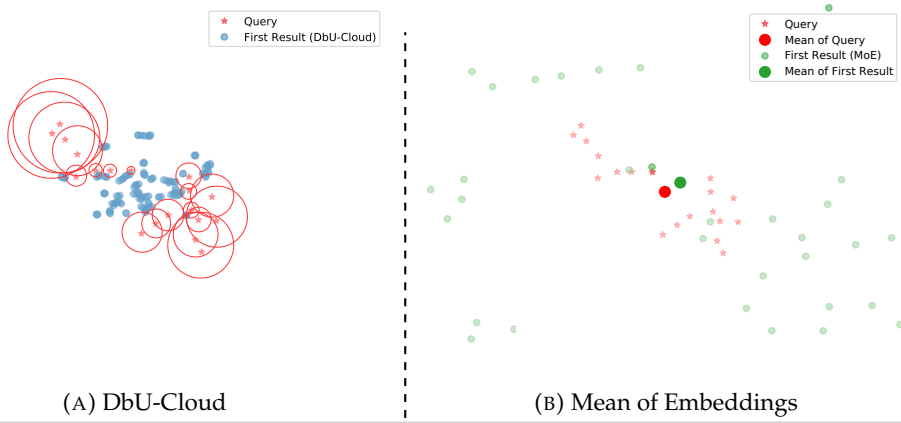
FIGURE 11.13: Analysis of the impact of the corpus on the precision P (y-axis) of the baselines (XoE, MoE), and DbU-Cloud. The values are computed by the mean across all the queries using BERT-base (on the upper side) and BERT-large (on the lower side). The cut-off threshold Z is on the x-axis.

by DbU-Cloud with BERT-large with respect to the baselines XoE and MoE is notable.

11.3 Visual and Qualitative Anecdotal Analysis

To provide a comprehensive understanding of the efficacy of the DbU-Cloud compared to the Mean of Embeddings baseline, this section presents anecdotal examples and visual comparisons of results produced by both methods, using two specific examples: one from the LISA dataset and another from the MS_MARCO dataset. This analysis aims to clearly illustrate the scenarios (i.e., queries and documents) where the DbU-Cloud approach is necessary and where the Mean of Embeddings method fails to deliver accurate results. Given the high-dimensional nature of the data, making these plots interpretable for readers presents a significant challenge.

For the aforementioned reason, to properly visualize how the methods work within the embedding space, we post-process the embeddings of documents and queries as described in [23]. Initially, we use Principal Component Analysis (PCA) [92] to reduce their original dimensionality from 768 to 50. Subsequently, we apply t-SNE [167] to further reduce their dimensionality from 50 to 2. This two-step dimensionality reduction is necessary to obtain results that are both interpretable and stable, as the stochastic nature of t-SNE produces different results if applied directly to the high-dimensional space of 768. The final output can be visualized in a two-dimensional plane.



(A) DbU-Cloud (B) Mean of Embeddings

Corpus	Query	Query Text	Method	First Result Text	Doc ID	Rel
LISA	34	"I should be pleased to receive any information on the evaluation of PRECIS (Preserved Context Indexing System)."	DbU-Cloud	"Where do we go from here? Presents the author's personal opinion that the next quantum leap in information technology will be the development of an active information system. [...]"	3384	Yes
			MoE	"Information Retrieval Research. Papers given at the 1st Joint British Computer Society/Association for Computing Machinery Symposium on Information Storage and Retrieval, Research and Development in Information Retrieval, Cambridge, June 80. [...]"	3388	No

(C) Example of document retrieved by DbU-Cloud and MoE respect to a query of the LISA corpus.

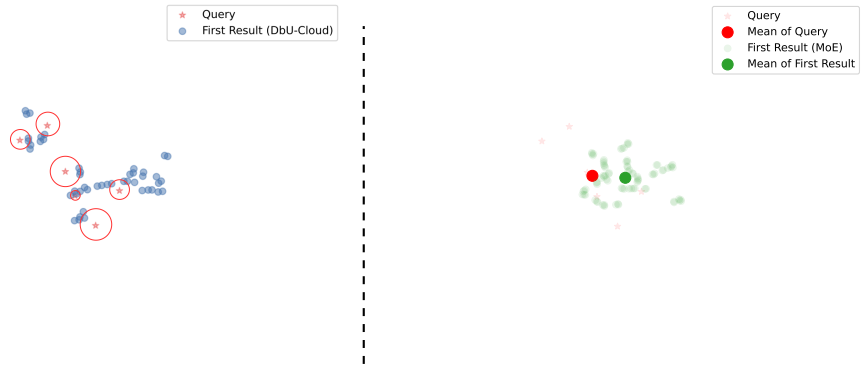
FIGURE 11.14: Visualization of the Clouds of Vectors of the first result obtained via DbU-Cloud (on the left) and MoE (on the right) against the query on corpus LISA.

Figures 11.14 and 11.15 depict the anecdotal examples on corpora LISA and MS_MARCO, respectively. We selected these two corpora to show examples for both a specialized and vernacular language features. In both cases, we employed ALL_MPNET as the reference LLM, as it consistently produced the best overall results. To better understand and show the conditions where DbU-Cloud offers a clear advantage, we selected those queries where the first result returned by DbU-Cloud is considered relevant (P@1 = 1), while the one returned from the MoE is irrelevant (P@1 = 0).

On the left, we visualize the Clouds of Vectors for the first result obtained by our approach, and on the right, the Mean of Embeddings obtained by the pooling mechanism. This visualization visually demonstrates the operational differences between the two approaches. In the DbU-Cloud visualization, the red circles represent the k-neighborhood defined in Section 11.1. Larger circles indicate that the query tensor is more distant from a dense cluster of document embeddings and, thus, less relevant. In the Mean of Embeddings (MoE) visualization, the individual token embeddings are discarded, and the only distance that matters for computing the Relevance Score is the one between the two means (represented by the larger circles). Moreover we have supplemented the visual comparisons with textual explanations. Beneath the visualizations, we provide a table containing the text and IDs of the queries and retrieved documents for each figure. The column "Rel" indicates whether or not the returned document is relevant to the query in the ground truth.

We now proceed to analyze each example, focusing on the corresponding visualizations and the associated document content.

Anecdotal example on LISA. Figure 11.14 presents an anecdotal example for the LISA corpus. The first document retrieved by DbU-Cloud method, as previously stated, is relevant, whereas the first document retrieved using the MoE method is not. This visualization clearly illustrates the Squeezing problem when using mean pooling for all token tensors in a document. All the tensors of the document used by DbU-Cloud method are represented in a dense Cloud of Vectors, while those of the document retrieved by MoE are sparse and distant from the query embeddings. However, due to their distribution, the mean of these embeddings is close to the mean of the query embeddings. Consequently, the document retrieved by MoE is erroneously ranked first. This example is significant as it provides empirical evidence of the *Squeezing Problem* and demonstrates how our approach improves retrieval effectiveness by considering both the density and proximity of the embeddings.



(A) DbU-Cloud (B) Mean of Embeddings

Corpus	Query	Query Text	Method	First Result Text	Doc ID	Rel
MS_MARCO	18711	Another name for Attack on Titan	DbU-Cloud	"Characters Attack on Titan: the military police brigade,the elite force that answers to the king responsible for maintaining the peace, their organization is infamously corrupt and many recruits hope to join in order to take advantage of the safety and comforts it affords. "	8261776	Yes
			MoE	"Percy is the black sheep of the Weasley family. His full name is Percival Ignatius Weasley. Percival is also one of professor Albus Dumbledores names."	8257847	No

(C) Example of document retrieved by DbU-Cloud and MoE respect to a query of the MS_MARCO corpus.

FIGURE 11.15: Visualization of the Clouds of Vectors of the first result obtained via DbU-Cloud (on the left) and the Mean of Embeddings (on the right) against the Query on corpus MS_MARCO.

Anecdotal example on MS_MARCO. Figure 11.15 presents an example from the MS_MARCO corpus. In this scenario, the document ranked first by our method contains embeddings that are semantically related to all query embeddings, while the document retrieved by MoE contains embeddings unrelated to all the query embeddings. As evident from the text reported in Table 11.15c, DbU-Cloud approach correctly identifies the subject of the query and retrieves a very relevant document.

In contrast, MoE retrieves a document that fits the general topic of a fictional TV series or movie but fails to retrieve the requested movie. This example shows that DbU-Cloud method values every query embedding, avoiding the limitation of the classical term-matching behavior while leveraging the semantic embeddings provided by the LLM.

11.4 Discussion

Experimental results. We first conducted a sensitivity analysis on the hyperparameter k, which controls the number of nearest token embeddings considered by DbU-Cloud when computing local density. Our findings reveal that while $k = 1$ yields the highest effectiveness on non-fine-tuned models, performance remains consistently high across all values $k \in \{1, \ldots, 5\}$. This indicates that DbU-Cloud is robust to slight variations of k, reducing the need for extensive hyperparameter tuning in practical deployments.

The analysis in Sections 11.2.3 and 11.2.3 lead to the following observations:

i) DbU-Cloud demonstrates a superior ability to leverage the knowledge embedded in the LLM compared to traditional MoE and XoE approaches.

ii) in fine-tuned models, the margin between the baselines is limited by the pre-fine-tuning stage of the model, which by its nature (see Section 11.2.1) brings closer the embeddings making them more similar.

This is supported by the evidence that MoE and XoE have almost similar effectiveness with it (which implies that the element-wise max of the embeddings is almost equal to the mean, and this happens only if the embeddings are very similar on their own). The presented results are very promising: the density-based approach of DbU-Cloud is able to mitigate the squeezing problem introduced by the classical aggregation techniques and, if coupled with fine-tuned LLMS, obtain state of the art document retrieval effectiveness. Notably, the Upper Bound (UB) of DbU-Cloud outperforms all other methods with any LLM, confirming the potential for improvement of DbU-Cloud.

It is noteworthy that DbU-Cloud outperforms the baselines even when using models like ALL-MPNET, which have been specifically fine-tuned via contrastive learning to ensure that mean-pooled sentence embeddings capture semantic similarity effectively. This result underscores the limits of pooling-based aggregation and highlights the advantage of leveraging fine-grained, density-based approaches like DbU-Cloud.

Summing up, DbU-Cloud outperforms all baselines by a large margin, especially on not fine-tuned models (BERT-base and GPT). The DbU-Cloud Upper Bound increases this gap even more significantly, exhibiting further room for improvement. In Section 11.2.4 we compare DbU-Cloud on the best-performing LLM with BM25, a classical Relevance schema. The results demonstrate that DbU-Cloud significantly outperforms BM25. In particular, DbU-Cloud Upper Bound achieves a substantial performance gain over BM25, especially in terms of MRR and DCG@10 metrics.

The anecdotal analysis in Section 11.3 corroborates the limitation of the Mean (and Max) pooling techniques for Document Ranking. A single document tensor is not able to express the entire semantics of the document, especially when the document contains more than one semantic area. The examples in this section show how the documents retrieved with MoE have a general sense of relevance w.r.t. to the query, while documents retrieved with DbU-Cloud correspond to the query in a more nuanced and precise way.

Finally, in Section 11.2.4, we also perform an ablation study on the Embedding sizes on our methods, using BERT-base and BERT-large. Early results show that DbU-Cloud performs even better with larger embeddings, and the gap with the baselines is even wider. This suggests that DbU-Cloud can fully leverage the richer representations offered by larger models, whereas traditional pooling techniques—like those used in the baselines—struggle even more as dimensionality increases, further cramping diverse semantic signals into a single fixed-size vector. These results reinforce the idea that representing documents with a single pooled embedding is inherently limiting, especially for complex, multi-topic content.

Computational Efficiency. An in-depth evaluation of the efficiency is out of the scope of this work, which focuses on introducing the Clouds of Vectors formalization and a first implementation of DbU-Cloud, a relevance metric for Document Retrieval. While employing every term embedding rather than a pooling is generally more computationally expensive, the field is full of techniques aimed at optimizing similarity measure computations. Researchers and practitioners have developed various strategies to enhance the efficiency of similarity calculations in the context of vectorized representations. It is important to note that many cosine similarity computations can be precomputed and stored before the document ranking phase. This offline computation involves calculating the cosine similarity scores between all pairs of vectors in advance and storing the results in a lookup table or an index. By doing so, real-time computation is avoided during retrieval, leading to a considerable reduction in response time. While using individual term embeddings might introduce some computational overhead, the field offers a plethora of techniques to address this issue.

Experimental Settings. In our Experimental Evaluation, we employed several corpora selected based on their linguistic characteristics. Notably, two corpora (MS MARCO and NFCORPUS) are widely recognized and commonly used in the literature and are available in the ir-datasets library. Additionally, we utilized a variety of LLMs, including BERT, SBERT, DPR, MPNET, and GPT, to cover as many foundational models as possible. Even though expanding the range of corpora could provide a more comprehensive understanding of the proposed method's performance across diverse linguistic contexts, we argue that the selected ones already encompass a full range of available types. Similarly, incorporating a broader set of new LLMs might allow for a more thorough exploration of the model's generalizability and robustness.

11.5 Threats to Validity

This section discusses possible threats to the usage of DbU-Cloud or the validity of our experimental evaluation.

Computational efficiency. An in-depth evaluation of the efficiency is out of the scope of this work, which focuses on introducing the Clouds of Vectors formalization and a first implementation of DbU-Cloud, a relevance metric for Document Retrieval. While employing every term embedding rather than a pooling is generally more computationally expensive, the field is full of techniques aimed at optimizing cosine similarity computations. Researchers and practitioners have developed various strategies to enhance the efficiency of similarity calculations in the context of vectorized representations. It is important to note that many cosine similarity computations can be precomputed and stored offline. This precomputation involves calculating the cosine similarity scores between all pairs of vectors in advance and

storing the results in a lookup table or an index. By doing so, real-time computation is avoided during retrieval, leading to a considerable reduction in response time. While using individual term embeddings might introduce some computational overhead, the field offers a plethora of techniques to address this issue.
Validation on more corpora and LLMs. In our Experimental Evaluation, we employed several corpora selected based on their linguistic characteristics. Notably, two corpora (MS_MARCO and NFCORPUS) are widely recognized and commonly used in the literature and are available in the ir-datasets library. Additionally, we utilized a variety of LLMs, including BERT, SBERT, DPR, MPNET, and GPT, to cover as many foundational models as possible. Even though Expanding the range of corpora could provide a more comprehensive understanding of the proposed method's performance across diverse linguistic contexts, we argue that the selected ones already encompass a wide range of available types. Similarly, incorporating a broader set of new LLMs might allow for a more thorough exploration of the model's generalizability and robustness.

11.6 Conclusions and Future work

In this work, we identified a common limitation arising when using pooling-based LLMs for document ranking and highlighted its negative impact on retrieval effectiveness.
We then presented a new formalization of the Document-Query problem, called Clouds of Vectors, with the aim of leveraging every term embedding produced by the chosen LLM. On the basis of this model, we presented DbU-Cloud, a novel density-based unsupervised method for Document Ranking with LLMs. We thoroughly tested DbU-Cloud against the baselines of the Mean and Max of Embeddings, the most popular pooling techniques. The experimental evaluation, encompassing four corpora and five LLMs with different language features and architectures respectively, shows how DbU-Cloud surpasses the baselines in almost all the settings and corpora. Moreover, corroborating its robustness, we found that DbU-Cloud applied to the best-performing sentence transformer also outperforms classical retrieval methods like BM25.
Regarding future works, it must be noted that DbU-Cloud is the first of many implementations that are possible by employing the Clouds of Vectors model. Future work include expanding the experimentation on other models and corpora, including domain-specific corpora where finding the best possible ranking has significant impact. Moreover, a slightly modified version of DbU-Cloud could also be employed to fine-tune LLMs for Document Ranking. Lastly, optimizing the algorithm and its implementation represent avenues of future work.

Chapter 12

Conclusion

Throughout this thesis, we presented several principled advances on all trustworthiness dimensions recognized by the EU HLEG: lawfulness, ethics and robustness. Our advances targeted specific areas for each dimension. We selected those areas (Machine Unlearning, Fairness, and Explainability) based on their relevance in an era of unprecedented speed of progress for Machine Learning systems.

12.1 Advances in Machine Unlearning

ERASURE. Specifically, for MU, we first proposed ERASURE, a unified framework for comparison and development. ERASURE introduces a unified, extensible platform for MU, addressing a critical need for reproducible research. Using results from two representative methods across four domains as a proof of concept, we demonstrate the framework's ability to support diverse experimental setups at scale. ERASURE's modular design enables the integration of custom datasets, methods, and evaluation measures with minimal overhead. This platform will accelerate progress in the field by providing researchers with the tools needed to build, compare, and refine unlearning methods efficiently.

Benchmark. Building on the ERASURE framework, we built a multi-domain comprehensive benchmark of existing methods in the literature. In this work, we bring order to the landscape of Machine Unlearning methods for classification by providing: *(i).* the most comprehensive benchmark to date, covering 4 data modalities, 12 Unlearners, 8 datasets, and 8 models; *(ii).* the first systematic evaluation on the tabular and graph modalities; *(iii).* a taxonomy of Unlearner methods in the literature, *(iv).* a novel unified metric, LUMA, which quantifies performance as a single value to support hyperparameter tuning and unlearner selection in practice; and *(v).* a publicly available, fully reproducible, and extensible benchmark to facilitate fair comparison in future work.

Our results lead to clear guidelines for the design and deployment of Unlearners, and expose critical shortcomings that can only be revealed through cross-modality analysis. Together, these contributions establish a common ground for evaluating and comparing Machine Unlearning approaches for classification. By ensuring reproducibility and extensibility, our work provides a solid basis for future methods to be rigorously evaluated.

Graphs and Forget Set Identification. We then extended our applicability to the Graph domain and, specifically, to the tasks of node and edge classifications. We highlighted several weaknesses currently present in the literature for graph unlearning, specifically for the evaluation of methods, where small datasets are used to infer general trends and behaviors.

Lastly, we went beyond the state of the art by formalizing a novel problem: the Forget Set Identification. We formalize the ForSId problem, prove its NP-hardness, and design an algorithm for it based on a connection to Red-Blue Set Cover. Extensive experiments attest high performance of the proposed algorithm.

12.2 Advances in Fairness.

Graph Analysis on co-authorship networks. For Fairness, we first produced an empirical study on the structured data available for the Software Engineering Community. We showed how bias in the data can be found, studied, and identified even across graphs. We performed two sets of evaluations: (i) we compared the Italian SE community with the Worldwide SE community via Network Analysis techniques and metrics, (ii) we compared the Italian SE community with the Italian Informatics community to infer differences in the pattern of academic promotions for the different gender groups considering also differences between best performers and mid-low performers in terms of publications and citations.

Debiaser for Multiple Variables. Then, we proposed DEMV, a Debiaser for Multiple Variables, which operates on multiclass classification scenarios. With DEMV, we addressed the problem of bias mitigation in the multi-class classification context with multiple sensitive variables. To the best of our knowledge, DEMV is the first pre-processing method able to handle bias in both binary and multi-class classification problems with any number of sensitive variables. We have exhaustively evaluated our algorithm by comparing it with three established baselines using a heterogeneous set of binary and multi-class datasets, with a different number of sensitive variables, and by employing a heterogeneous set of classification methods.

12.3 Advances in Explainability.

Lastly, for Explainability, we recognized the increasing importance of LLMs, RAGs, and the dense retrievers that make RAGs work. To make these systems robust to deployment environment changes, we need to make sure that they can produce explanations of their decisions and rankings. For this reason, we proposed DbU-Cloud, a novel framework and method for dense document ranking, employing all embeddings from a document (produced by an LLM) rather than averaging them.

In particular, we identified a common limitation arising when using pooling-based LLMs for document ranking and highlighted its negative impact on retrieval effectiveness. We then presented a new formalization of the Document-Query problem, called Clouds of Vectors, with the aim of leveraging every term embedding produced by the chosen LLM. On the basis of this model, we presented DbU-Cloud, a novel density-based unsupervised method for Document Ranking with LLMs. We thoroughly tested DbU-Cloud against the baselines of the Mean and Max of Embeddings, the most popular pooling techniques. The experimental evaluation, encompassing four corpora and five LLMs with different language features and architectures respectively, shows how DbU-Cloud surpasses the baselines in almost all the settings and corpora. Moreover, corroborating its robustness, we found that DbU-Cloud applied to the best-performing sentence transformer also outperforms classical retrieval methods like BM25.

12.4 Future Work

The speed at which Machine Learning is progressing is thought to be worrisome by many. Legal actions and public opinion are struggling to keep up the pace with new technologies. Despite these difficulties, we believe that it is possible to regulate and develop AI in a way that is respectful of the work of people, of the *right to be forgotten*, and in general, preserves privacy, and ultimately meets the standards of trustworthiness.

These standards must be met not only during development, but during the entire lifecycle of the model. For this reason, researching on Machine Unlearning, and how it can impact models in the long run, is crucial. We envision a continuously learning model that updates old information, removes sensitive data, and complies with the right to be forgotten, throughout its entire lifecycle.

We plan to extend the study on the Forget Set Identification (ForSId) Problem to this context. Notably, for future work, researching more solid parameter selection techniques, like bisection, could further improve the results of our algorithm.

In the future, we plan to devise algorithms that tackle ForSId more directly (without passing through proxy problems), deal with approximation properties and algorithms for the proxy Red-Blue ForSId problem, consider different forms of (un)wanted behavior, investigate ad-hoc methodologies for setting the parameters of our method(s), and experimentally test ForSId in additional scenarios and settings. Specifically, although ILP solvers are powerful, they have practical limitations in terms of scalability and computational efficiency.

The extension of MU to Federated learning systems is also very important for privacy preservation. Some decentralized tasks can only be carried out by federated settings, and while MU has been applied in some form, much research remains to be carried out on that specific application.

In general, while the fields of fairness and explainability build upon extensive existing literature, MU is a rapidly emerging topic that brings what was previously missing: a concrete, technically grounded framework for enforcing the right to be forgotten and ensuring legal compliance in modern AI systems.

By combining these fields, we can build ML systems that meet legal, ethical, and technical standards of trustworthiness.

Acknowledgements

I was often advised against pursuing a Ph.D.; however, my choice to ignore that advice proved to be more fruitful than I had imagined.

Over the past three years, I've been surprised by the rate at which I've grown as a person. This would not have been possible without the wonderful people I've met along the way. The list is quite lengthy, and those not named will forgive me; the moments we shared are far more valuable than names on a page that few will read.

Thanks to the people I met in L'Aquila and shared house and office spaces with: Katiuscia, Gennaro, Giordano, Mashal, Alina, Andrea, Diletta and Federico. All of them helped me see things differently when I got stuck.

Thanks to the people I met in Aarhus, and helped me on my stay there: prof. Davide Mottin, Nearchos, Vera, Maria, Chiara, Alberto, Catarina, and Giulia. I will always cherish those Danish memories.

Thanks to the people I barely met, but shared countless conversations with: Inne, Viola, Peppe, Erika, Simeme, Beppe, Jeremy, Martha, and Lorenzo. I could not hope for a better coven.

Thanks to the people I knew even before starting this journey, including Simone, Matteo, Daniele, Susanna, Alice and Giorgio. Some things never change.

I would also like to thank my advisors, professors Giovanni Stilo and Antinisca Di Marco, for guiding me through this journey. Although the ride was not always smooth, we still managed to make it to the finish line.

Bibliography

[1] V.P. Abidha and Pradeesha Ashok. "Red Blue Set Cover problem on axis-parallel hyperplanes and other objects". In: *Information Processing Letters* 186 (2024), p. 106485. ISSN: 0020-0190. DOI: https://doi.org/10.1016/j.ipl.2024.106485. URL: https://www.sciencedirect.com/science/article/pii/S0020019024000152.

[2] Giovanni Abramo, Ciriaco Andrea D'Angelo, and Francesco Rosati. "Gender bias in academic recruitment". In: *Scientometrics* 106.1 (2016), pp. 119–141.

[3] Alekh Agarwal et al. "A Reductions Approach to Fair Classification". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 60–69. URL: https://proceedings.mlr.press/v80/agarwal18a.html.

[4] *Agenzia Nazionale di Valutazione del Sistema Universitario e della Ricerca*. 2024. URL: https://www.anvur.it/attivita/asn/asn-2012-2013/indicatori-e-relative-mediane/.

[5] *AI Fairness 360 - Resources*. 2018. URL: https://aif360.mybluemix.net/resources#guidance.

[6] Sajid Ali et al. "Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence". In: *Information Fusion* 99 (2023), p. 101805. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2023.101805. URL: https://www.sciencedirect.com/science/article/pii/S1566253523001148.

[7] Enrique Amigó et al. "A unifying and general account of fairness measurement in recommender systems". en. In: *Information Processing & Management* 60.1 (Jan. 2023), p. 103115. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2022.103115. URL: https://www.sciencedirect.com/science/article/pii/S0306457322002163 (visited on 11/10/2022).

[8] Avishek Anand et al. *Explainable Information Retrieval: A Survey*. 2022. DOI: https://doi.org/10.48550/arXiv.2211.02405.

[9] **Andrea D'Angelo** et al. "Uncovering gender gap in academia: A comprehensive analysis within the software engineering community". In: ***Journal of Systems and Software*** ***[Q1]*** 217 (2024), p. 112162. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2024.112162. URL: https://www.sciencedirect.com/science/article/pii/S0164121224002073.

[10] Julia Angwin et al. "Machine bias". In: *ProPublica, May* 23.2016 (2016), pp. 139–159.

[11] Pradeesha Ashok, Sudeshna Kolay, and Saket Saurabh. "Multivariate Complexity Analysis of Geometric Red Blue Set Cover". In: *Algorithmica* 79.3 (2017), pp. 667–697.

[12] Sadia Asif and Mohammad Mohammadi Amiri. *OFMU: Optimization-Driven Framework for Machine Unlearning*. 2025. arXiv: 2509.22483 [cs.LG]. URL: https://arxiv.org/abs/2509.22483.

[13] Katherine A Austin, Catherine Martin Christopher, and Darby Dickerson. "Will I Pass the Bar Exam: Predicting Student Success Using LSAT Scores and Law School Performance". In: *HofstrA l. rev.* 45 (2016), p. 753.

[14] Ricardo Baeza-Yates. "Bias on the web". In: *Communications of the ACM* 61.6 (2018), pp. 54–61.

[15] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*. Vol. 463. ACM press New York, 1999.

[16] Ryan S Baker. "Learning Analytics: An Opportunity for Education". In: *XRDS: Crossroads, The ACM Magazine for Students* 29.3 (2023), pp. 18–21.

[17] Ryan S. Baker and Aaron Hawn. "Algorithmic Bias in Education". en. In: *International Journal of Artificial Intelligence in Education* (Nov. 2021). DOI: 10.1007/s40593-021-00285-9. (Visited on 09/21/2022).

[18] Alisha Baskota and Yiu-Kai Ng. "A graduate school recommendation system using the multi-class support vector machine and KNN approaches". In: *2018 IEEE International Conference on Information Reuse and Integration (IRI)*. IEEE. 2018, pp. 277–284.

[19] Barry Becker and Ronny Kohavi. *Adult*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20. 1996.

[20] Andrea Bianchi et al. "A method to comprehensively identify germline SNVs, INDELs and CNVs from whole exome sequencing data of BRCA1/2 negative breast cancer patients". In: ***NAR Genomics and Bioinformatics*** ***[Q1]*** 6.2 (Apr. 2024), lqae033. ISSN: 2631-9268. DOI: 10.1093/nargab/lqae033. eprint: https://academic.oup.com/nargab/article-pdf/6/2/lqae033/57258946/lqae033.pdf. URL: https://doi.org/10.1093/nargab/lqae033.

[21] Sarah Bird et al. *Fairlearn: A toolkit for assessing and improving fairness in AI*. Tech. rep. MSR-TR-2020-32. Microsoft, May 2020. URL: https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.

[22] Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. "Multimodal datasets: misogyny, pornography, and malignant stereotypes". In: *CoRR* (2021).

[23] Angie Boggust, Brandon Carter, and Arvind Satyanarayan. "Embedding Comparator: Visualizing Differences in Global Structure and Local Neighborhoods via Small Multiples". In: *Proceedings of the 27th International Conference on Intelligent User Interfaces*. IUI '22. Helsinki, Finland: Association for Computing Machinery, 2022, pp. 746–766. ISBN: 9781450391443. DOI: 10.1145/3490099.3511122. URL: https://doi.org/10.1145/3490099.3511122.

[24] Jacopo Bonato, Marco Cotogni, and Luigi Sabetta. "Is Retain Set All You Need in Machine Unlearning? Restoring Performance of Unlearned Models with Out-of-Distribution Images". In: *Proc. of European Conf. on Computer Vision (ECCV)*. 2024, pp. 1–19.

[25] Ludovico Boratto and Mirko Marras. "Advances in Bias-Aware Recommendation on the Web". In: *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. WSDM '21. Virtual Event, Israel: Association for Computing Machinery, 2021, pp. 1147–1149. ISBN: 9781450382977. DOI: 10.1145/3437963.3441665. URL: https://doi.org/10.1145/3437963.3441665.

[26] Vera Boteva et al. "A Full-Text Learning to Rank Dataset for Medical Information Retrieval". In: *Advances in Information Retrieval*. Ed. by Nicola Ferro et al. Cham: Springer International Publishing, 2016, pp. 716–722. ISBN: 978-3-319-30671-1.

[27] Lucas Bourtoule et al. *Machine Unlearning*. 2020. arXiv: 1912.03817 [cs.CR]. URL: https://arxiv.org/abs/1912.03817.

[28] M. M. Breunig et al. *LOF: Identifying Density-Based Local Outliers*. 2000.

[29] John R Busenbark et al. "Omitted variable bias: Examining management research with the impact threshold of a confounding variable (ITCV)". In: *Journal of Management* 48.1 (2022), pp. 17–48.

[30] *CA19122-European Network For Gender Balance in Informatics (EUGAIN)*. 2020. URL: https://eugain.eu/.

[31] Gabriel Cadamuro, Ran Gilad-Bachrach, and Jerry Zhu. "Debugging Machine Learning Models". In: 2016. URL: https://api.semanticscholar.org/CorpusID:14629198.

[32] Xavier F Cadet et al. *Deep Unlearn: Benchmarking Machine Unlearning*. 2024.

[33] Toon Calders et al. "Controlling Attribute Effect in Linear Regression". In: *2013 IEEE 13th International Conference on Data Mining*. ISSN: 2374-8486. Dec. 2013, pp. 71–80. DOI: 10.1109/ICDM.2013.114.

[34] Qingqing Cao et al. "DeFormer: Decomposing Pre-trained Transformers for Faster Question Answering". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 4487–4497. DOI: 10.18653/v1/2020.acl-main.411. URL: https://aclanthology.org/2020.acl-main.411/.

[35] Yinzhi Cao and Junfeng Yang. "Towards Making Systems Forget with Machine Unlearning". In: *Proc. of IEEE Symposium on Security and Privacy (S&P)*. 2015, pp. 463–480.

[36] Alberto Caprara, Paolo Toth, and Matteo Fischetti. "Algorithms for the Set Covering Problem". In: *Annals of Operations Research* 98.1 (Dec. 2000), pp. 353–371. ISSN: 1572-9338. DOI: 10.1023/A:1019225027893. URL: https://doi.org/10.1023/A:1019225027893.

[37] Nicholas Carlini et al. "Membership inference attacks from first principles". In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2022, pp. 1897–1914. DOI: 10.1109/SP46214.2022.9833649.

[38] Robert D. Carr et al. "On the red-blue set cover problem". In: *Proc. of ACM-SIAM Symp. on Discrete Algorithms (SODA)*. 2000, pp. 345–353.

[39] Simon Caton and Christian Haas. "Fairness in Machine Learning: A Survey". en. In: *arXiv:2010.04053 [cs, stat]* (Oct. 2020). arXiv: 2010.04053. (Visited on 11/11/2021).

[40] Sungmin Cha et al. *Learning to unlearn: instance-wise unlearning for pre-trained classifiers*. 2024. DOI: 10.1609/aaai.v38i10.28996. URL: https://doi.org/10.1609/aaai.v38i10.28996.

[41] Timothy M. Chan and Nan Hu. "Geometric red-blue set cover for unit squares and related problems". In: *Computational Geometry: Theory and Applications* 48.5 (2015), pp. 380–385.

[42] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[43] Min Chen et al. "Graph Unlearning". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. CCS '22. Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 499–513. ISBN: 9781450394505. DOI: 10.1145/3548606.3559352. URL: https://doi.org/10.1145/3548606.3559352.

[44] Ruizhe Chen et al. "Fast Model DeBias with Machine Unlearning". In: *Proc. of Conf. on Advances in Neural Information Processing Systems (NeurIPS)*. 2023.

[45] Zhenpeng Chen et al. "A Comprehensive Empirical Study of Bias Mitigation Methods for Machine Learning Classifiers". In: *ACM Transactions on Software Engineering and Methodology* 32.4 (). DOI: 10.1145/3583561.

[46] Jiali Cheng and Hadi Amiri. *Mu-bench: A multitask multimodal benchmark for machine unlearning*. 2024.

[47] Jiali Cheng et al. *GNNDelete: A General Strategy for Unlearning in Graph Neural Networks*. 2023. arXiv: 2302.13406 [cs.LG]. URL: https://arxiv.org/abs/2302.13406.

[48] Eli Chien, Chao Pan, and Olgica Milenkovic. *Certified Graph Unlearning*. 2022. arXiv: 2206.09140 [cs.LG]. URL: https://arxiv.org/abs/2206.09140.

[49] Eden Chlamtáč, Yury Makarychev, and Ali Vakilian. *Approximating Red-Blue Set Cover and Minimum Monotone Satisfying Assignment*. 2023. arXiv: 2302.00213 [cs.DS]. URL: https://arxiv.org/abs/2302.00213.

[50] Dasol Choi and Dongbin Na. "Towards machine unlearning benchmarks: Forgetting the personal identities in facial recognition systems". In: *arXiv preprint arXiv:2311.02240* (2023).

[51] Vikram S Chundawat et al. "Can bad teaching induce forgetting? unlearning in deep networks using an incompetent teacher". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. 2023, pp. 7210–7217.

[52] Vikram S Chundawat et al. "Zero-shot machine unlearning". In: *IEEE Transactions on Information Forensics and Security* 18 (2023), pp. 2345–2354. DOI: 10.1109/TIFS.2023.3265506.

[53] Vikram S. Chundawat et al. "Can Bad Teaching Induce Forgetting? Unlearning in Deep Networks Using an Incompetent Teacher". In: *Proc. of AAAI Conf. on Artificial Intelligence (AAAI)*. 2023, pp. 7210–7217.

[54] Kevin A Clarke. "The phantom menace: Omitted variable bias in econometric research". In: *Conflict management and peace science* 22.4 (2005), pp. 341–352.

[55] Paulo Cortez et al. "Modeling wine preferences by data mining from physicochemical properties". In: *Decision support systems* 47.4 (2009), pp. 547–553.

[56] Paulo Cortez et al. *Wine Quality*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C56S3T. 2009.

[57] Marco Cotogni et al. "Duck: Distance-based unlearning via centroid kinematics". In: *arXiv preprint arXiv:2312.02052* (2023). DOI: 10.48550/arXiv.2312.02052.

[58] Nick Craswell et al. *Overview of the TREC 2019 deep learning track*. 2020. arXiv: 2003.07820.

[59] Kate Crawford. *Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence*. Yale University Press, 2022.

[60] Fabio Crestani et al. ""Is This Document Relevant?...Probably": A Survey of Probabilistic Models in Information Retrieval". In: *ACM Comput. Surv.* 30.4 (Dec. 1998), pp. 528–552. ISSN: 0360-0300. DOI: 10.1145/299917.299920. URL: https://doi.org/10.1145/299917.299920.

[61] Philippe Cudre-Mauroux. *Semantic Search*. Cham, 2018. DOI: 10.1007/978-3-319-63962-8_231-1. URL: https://doi.org/10.1007/978-3-319-63962-8_231-1.

[62] Washington Cunha et al. "A Comparative Survey of Instance Selection Methods applied to Non-Neural and Transformer-Based Text Classification". In: *ACM Comput. Surv.* 55.13s (July 2023). ISSN: 0360-0300. DOI: 10.1145/3582000. URL: https://doi.org/10.1145/3582000.

[63] Sergio Currarini, Jesse Matheson, and Fernando Vega-Redondo. "A simple model of homophily in social networks". In: *European Economic Review* 90.C (2016), pp. 18–39. DOI: 10.1016/j.euroecorev.2016. URL: https://ideas.repec.org/a/eee/eecrev/v90y2016icp18-39.html.

[64] Brian d'Alessandro, Cathy O'Neil, and Tom LaGatta. "Conscientious classification: A data scientist's guide to discrimination-aware classification". In: *Big data* 5.2 (2017), pp. 120–134.

[65] Giordano d'Aloisio et al. "Data-Driven Analysis of Gender Fairness in the Software Engineering Academic Landscape". In: *Software Architecture. ECSA 2023 Tracks, Workshops, and Doctoral Symposium [A]*. Ed. by Bedir Tekinerdoğan et al. Cham: Springer Nature Switzerland, 2024, pp. 89–103. ISBN: 978-3-031-66326-0.

[66] Giordano d'Aloisio et al. "Debiaser for Multiple Variables to enhance fairness in classification tasks". In: *Information Processing and Management* 60.2 (2023). All Open Access, Hybrid Gold Open Access. DOI: 10.1016/j.ipm.2022.103226.

[67] Giordano d'Aloisio et al. "Enhancing Fairness in Classification Tasks with Multiple Variables: A Data- and Model-Agnostic Approach". In: *Advances in Bias and Fairness in Information Retrieval*. Ed. by Ludovico Boratto et al. Cham: Springer International Publishing, 2022, pp. 117–129. ISBN: 978-3-031-09316-6.

[68] Giordano d'Aloisio et al. "Enhancing Fairness in Classification Tasks with Multiple Variables: a Data- and Model-Agnostic Approach". In: *To be published at "Proceedings of Third International Workshop on Algorithmic Bias in Search and Recommendation (Bias 2022)"* (2022).

[69] Silvia D'Amicantonio et al. "A Comprehensive Strategy to Bias and Mitigation in Human Resource Decision Systems". In: *Proceedings of the 5th Italian Workshop on Explainable Artificial Intelligence (XAI.it 2024)*. Vol. 3839. CEUR Workshop Proceedings. CEUR-WS, 2024, pp. 11–27.

[70] Andrea D'Angelo and Giordano d'Aloisio. *Public SE Fairness*. 2024. DOI: https://doi.org/10.5281/zenodo.11075506.

[71] Andrea D'Angelo, Giovanni Stilo, and Antinisca di Marco. *A Novel Relevance Score for Unsupervised Retrieval with Large Language Models*. 2024. URL: https://kdd2024.kdd.org/wp-content/uploads/2024/07/paper_11.pdf.

[72] Andrea D'Angelo et al. "ERASURE: A Modular and Extensible Framework for Machine Unlearning". In: *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM 2025)*. ACM, 2025.

[73] Andrea D'Angelo et al. "How to Make Reproducible Research in Machine Unlearning with ERASURE". In: *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*. Ed. by James Kwok. Demo Track. International Joint Conferences on Artificial Intelligence Organization, Aug. 2025, pp. 11025–11029. DOI: 10.24963/ijcai.2025/1255. URL: https://doi.org/10.24963/ijcai.2025/1255.

[74] Andrea D'Angelo, Francesco Gullo, and Giovanni Stilo. "The forget-set identification problem". In: *Machine Learning* 114.11 (2025), p. 247. DOI: 10.1007/s10994-025-06897-9. URL: https://doi.org/10.1007/s10994-025-06897-9.

[75] Quang-Vinh Dang. "Right to Be Forgotten in the Age of Machine Learning". In: *Advances in Digital Science*. 2021, pp. 403–411.

[76] Shib Sankar Dasgupta et al. *Improving Local Identifiability in Probabilistic Box Embeddings*. 2020. arXiv: 2010.04831 [cs.LG]. URL: https://arxiv.org/abs/2010.04831.

[77] Giuseppe Della Penna, Sergio Orefice, and Andrea D'Angelo. "Exploiting spatial relations for grammar-based specification of multidimensional languages". In: *Knowledge and Information Systems* 65.10 (2023), pp. 3995–4020. ISSN: 0219-3116. DOI: 10.1007/s10115-023-01879-6. URL: https://doi.org/10.1007/s10115-023-01879-6.

[78] Christophe Denis et al. "Fairness guarantee in multi-class classification". In: *arXiv:2109.13642 [math, stat]* (Sept. 2021). arXiv: 2109.13642.

[79] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805.

[80] Pengfei Ding et al. "Adaptive Graph Unlearning". In: *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*. Ed. by James Kwok. Main Track. International Joint Conferences on Artificial Intelligence Organization, Aug. 2025, pp. 2767–2774. DOI: 10.24963/ijcai.2025/308. URL: https://doi.org/10.24963/ijcai.2025/308.

[81] Pedro Domingos and Michael Pazzani. "On the optimality of the simple Bayesian classifier under zero-one loss". In: *Machine learning* 29.2 (1997), pp. 103–130.

[82] Yushun Dong et al. "IDEA: A Flexible Framework of Certified Unlearning for Graph Neural Networks". In: *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '24. Barcelona, Spain: Association for Computing Machinery, 2024, pp. 621–630. ISBN: 9798400704901. DOI: 10.1145/3637528.3671744. URL: https://doi.org/10.1145/3637528.3671744.

[83] Vineeth Dorna et al. *OpenUnlearning: Accelerating LLM Unlearning via Unified Benchmarking of Methods and Metrics*. 2025. arXiv: 2506.12618 [cs.CL]. URL: https://arxiv.org/abs/2506.12618.

[84] Finale Doshi-Velez and Been Kim. *Towards A Rigorous Science of Interpretable Machine Learning*. 2017. arXiv: 1702.08608 [stat.ML]. URL: https://arxiv.org/abs/1702.08608.

[85] Cynthia Dwork. "Differential Privacy". In: *Automata, Languages and Programming*. Ed. by Michele Bugliesi et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12. ISBN: 978-3-540-35908-1.

[86] Cynthia Dwork et al. "Fairness through awareness". In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ITCS '12. New York, NY, USA: Association for Computing Machinery, Jan. 2012, pp. 214–226. ISBN: 978-1-4503-1115-1. DOI: 10.1145/2090236.2090255. (Visited on 12/16/2021).

[87] Michael Elkin and David Peleg. "The Hardness of Approximating Spanner Problems". In: *Theory of Computing Systems (TOCS)* 41.4 (2007), pp. 691–729.

[88] Elsevier. *Scopus*. 2023. URL: https://www.scopus.com.

[89] European Commission High-Level Expert Group on Artificial Intelligence. *Ethics Guidelines for Trustworthy AI*. Accessed: 2025-10-30. 2019. URL: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai.

[90] European Parliament and Council of the European Union. *Regulation of the European Parliament and of the Council laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts*. COM/2021/206 final. 2021. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0206.

[91] European Union. *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act) and amending certain Union legislative acts*. Official Journal of the European Union, L 168, 1–85. Accessed 2025-10-04. 2024. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689.

[92] Karl Pearson F.R.S. "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572. DOI: 10.1080/14786440109462720.

[93] Fairlearn. *Fairlearn documentation*. 2022. URL: https://fairlearn.org/main/faq.html (visited on 04/13/2022).

[94] Chongyu Fan et al. "Challenging Forgets: Unveiling the Worst-Case Forget Sets in Machine Unlearning". In: *Proc. of European Conf. on Computer Vision (ECCV)*. 2024, pp. 278–297.

[95] Chongyu Fan et al. "Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation". In: *arXiv preprint arXiv:2310.12508* (2023).

[96] Luyang Fang et al. *Knowledge Distillation and Dataset Distillation of Large Language Models: Emerging Trends, Challenges, and Future Directions*. 2025. arXiv: 2504.14772 [cs.CL]. URL: https://arxiv.org/abs/2504.14772.

[97] Michael P Fay and Michael A Proschan. "Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules". In: *Stat. Surv.* 4.none (Jan. 2010).

[98] Elaine Fehrman et al. "The Five Factor Model of Personality and Evaluation of Drug Consumption Risk". en. In: *Data Science*. Ed. by Francesco Palumbo, Angela Montanari, and Maurizio Vichi. Studies in Classification, Data Analysis, and Knowledge Organization. Cham: Springer International Publishing, 2017, pp. 231–242. ISBN: 978-3-319-55723-6. DOI: 10.1007/978-3-319-55723-6_18.

[99] Michael Feldman et al. "Certifying and Removing Disparate Impact". en. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Sydney NSW Australia: ACM, Aug. 2015, pp. 259–268. ISBN: 978-1-4503-3664-2. DOI: 10.1145/2783258.2783311. (Visited on 11/11/2021).

[100] Wirth F. Ferger. "The Nature and Use of the Harmonic Mean". In: *Journal of the American Statistical Association* 26.173 (1931), pp. 36–40. DOI: 10.1080/01621459.1931.10503148. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1931.10503148. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1931.10503148.

[101] Matthias Fey and Jan Eric Lenssen. *Fast Graph Representation Learning with PyTorch Geometric*. 2019. arXiv: 1903.02428 [cs.LG]. URL: https://arxiv.org/abs/1903.02428.

[102] Luciano Floridi et al. "AI4People—An Ethical Framework for a Good AI Society: Opportunities, Risks, Principles, and Recommendations". In: *Minds and Machines* 28.4 (2018), pp. 689–707.

[103] Jack Foster, Stefan Schoepf, and Alexandra Brintrup. "Fast machine unlearning without retraining through selective synaptic dampening". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 2024, pp. 12043–12051.

[104] Sorelle A Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. "On the (im) possibility of fairness". In: *arXiv preprint arXiv:1609.07236* (2016).

[105] Jerome H Friedman. "Stochastic gradient boosting". In: *Computational statistics & data analysis* 38.4 (2002). Publisher: Elsevier, pp. 367–378.

[106] Rohit Gandikota et al. "Erasing Concepts from Diffusion Models". In: *Proc. of IEEE Int. Conf. on Computer Vision (ICCV)*. 2023, pp. 2426–2436.

[107] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. "CiteSeer: An Automatic Citation Indexing System". In: *Proceedings of the Third ACM Conference on Digital Libraries (DL '98)*. Ed. by I. Witten, R. Akscyn, and F. Shipman III. https://clgiles.ist.psu.edu/papers/DL-1998-citeseer.pdf. New York: ACM Press, 1998, pp. 89–98.

[108] Flavio Giobergia. *IMDb-ID Dataset*. https://huggingface.co/datasets/fgiobergia/imdb-id. Accessed: 2025-05-31. 2023.

[109] Shashwat Goel et al. "Towards adversarial evaluations for inexact machine unlearning". In: *arXiv preprint arXiv:2201.06640* (2022).

[110] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. "Eternal sunshine of the spotless net: Selective forgetting in deep networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 9304–9312.

[111] Aditya Golatkar et al. "Mixed-privacy forgetting in deep networks". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 792–801.

[112] Keltin Grimes et al. *Gone but Not Forgotten: Improved Benchmarks for Machine Unlearning*. 2024.

[113] Michael M Grynbaum and Ryan Mac. "The Times sues OpenAI and Microsoft over AI use of copyrighted work". In: *The New York Times* 27 (2023).

[114] Riccardo Guidotti et al. *A Survey Of Methods For Explaining Black Box Models*. 2018. arXiv: 1802.01933 [cs.CY]. URL: https://arxiv.org/abs/1802.01933.

[115] Marcello Gulli. *AG's corpus of news articles*. http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html.

[116] Chuan Guo et al. "Certified data removal from machine learning models". In: *Proceedings of the 37th International Conference on Machine Learning*. ICML'20. JMLR.org, 2020.

[117] Martin T Hagan, Howard B Demuth, and Mark Beale. *Neural network design*. PWS Publishing Co., 1997.

[118] Sara Hajian, Francesco Bonchi, and Carlos Castillo. "Algorithmic Bias: From Discrimination Discovery to Fairness-aware Data Mining". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. ACM, 2016, pp. 2125–2126.

[119] Moritz Hardt, Eric Price, and Nati Srebro. "Equality of opportunity in supervised learning". In: *Advances in neural information processing systems* 29 (2016), pp. 3315–3323.

[120] Zellig S. Harris. "Distributional Structure". In: *WORD* 10.2-3 (Aug. 1954), pp. 146–162. ISSN: 0043-7956. DOI: 10.1080/00437956.1954.11659520. URL: http://dx.doi.org/10.1080/00437956.1954.11659520 (visited on 03/31/2017).

[121] Jamie Hayes et al. "Inexact unlearning needs more careful evaluations to avoid a false sense of privacy". In: *arXiv preprint arXiv:2403.01218* (2024).

[122] Haibo He et al. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*. IEEE. 2008, pp. 1322–1328.

[123] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[124] Yue Huang et al. *TrustLLM: Trustworthiness in Large Language Models*. 2024. arXiv: 2401.05561 [cs.CL]. URL: https://arxiv.org/abs/2401.05561.

[125] *Hugging Face Sentence Transformer Library*. URL: https://huggingface.co/sentence-transformers.

[126] HuggingFace. *Hugging Face: Natural Language Processing and Machine Learning Tools*. https://huggingface.co. Accessed: 2025-05-01. 2025.

[127] Sonja M. Hyrynsalmi. "How Diversity and Inclusion Are Approached in Software Engineering University-level Teaching". In: *2023 IEEE/ACM 4th Workshop on Gender Equity, Diversity, and Inclusion in Software Engineering (GE-ICSE)*. May 2023, pp. 17–24. DOI: 10.1109/GEICSE59319.2023.00007. URL: https://ieeexplore.ieee.org/abstract/document/10213434 (visited on 05/12/2024).

[128] ICL. *The Performance Application Programming Interface (PAPI)*. https://en.wikipedia.org/wiki/Performance_Application_Programming_Interface. Accessed: 2025-05-01. 2024.

[129] *Italian National Scientific Abilitation*. 2024. URL: https://abilitazione.mur.gov.it/public/index.php.

[130] Cuiqing Jiang et al. "Capturing helpful reviews from social media for product quality improvement: a multi-class classification approach". In: *International Journal of Production Research* 55.12 (2017), pp. 3528–3541.

[131] Harry H. Jiang et al. "AI Art and its Impact on Artists". In: *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*. AIES '23. Montréal, QC, Canada: Association for Computing Machinery, 2023, pp. 363–374. ISBN: 9798400702310. DOI: 10.1145/3600211.3604681. URL: https://doi.org/10.1145/3600211.3604681.

[132] Zhuoran Jin et al. *RWKU: Benchmarking Real-World Knowledge Unlearning for Large Language Models*. 2024.

[133] Faisal Kamiran and Toon Calders. "Data preprocessing techniques for classification without discrimination". en. In: *Knowledge and Information Systems* 33.1 (Oct. 2012), pp. 1–33. ISSN: 0219-1377, 0219-3116. DOI: 10.1007/s10115-011-0463-8. (Visited on 11/11/2021).

[134] Ryo Karakida and Kazuki Osawa. "Understanding Approximate Fisher Information for Fast Convergence of Natural Gradient Descent in Wide Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vancouver, Canada, 2020, pp. 808–819.

[135] Vladimir Karpukhin et al. "Dense Passage Retrieval for Open-Domain Question Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 6769–6781. DOI: 10.18653/v1/2020.emnlp-main.550. URL: https://aclanthology.org/2020.emnlp-main.550.

[136] Sunita Katoch, Sumit S. Chauhan, and Vijay Kumar. "A review on genetic algorithm: past, present, and future". In: *Multimedia Tools and Applications* 80 (2021), pp. 8091–8126. DOI: 10.1007/s11042-020-10139-6. URL: https://doi.org/10.1007/s11042-020-10139-6.

[137] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2004.

[138] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. *The UCI Machine Learning Repository*. https://archive.ics.uci.edu. Acc.: Jan 22, 2025. 2025.

[139] Omar Khattab and Matei Zaharia. "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Virtual Event, China: Association for Computing Machinery, 2020, pp. 39–48. ISBN: 9781450380164. DOI: 10.1145/3397271.3401075. URL: https://doi.org/10.1145/3397271.3401075.

[140] Jyrki Kivinen and Manfred K Warmuth. "Exponentiated gradient versus gradient descent for linear predictors". In: *information and computation* 132.1 (1997), pp. 1–63.

[141] Bryan Klimt and Yiming Yang. "The Enron Corpus: A New Dataset for Email Classification Research". In: *Machine Learning: ECML 2004*. Ed. by Jean-François Boulicaut et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 217–226. ISBN: 978-3-540-30115-8.

[142] Pang Wei Koh and Percy Liang. "Understanding Black-box Predictions via Influence Functions". In: *Proc. of Int. Conf. on Machine Learning (ICML)*. 2017, pp. 1885–1894.

[143] Pang Wei Koh and Percy Liang. *Understanding Black-box Predictions via Influence Functions*. 2020. arXiv: 1703.04730 [stat.ML]. URL: https://arxiv.org/abs/1703.04730.

[144] Ron Kohavi et al. "Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid." In: *Kdd*. Vol. 96. 1996, pp. 202–207.

[145] Alkis Koudounas et al. "" Alexa, can you forget me?" Machine Unlearning Benchmark in Spoken Language Understanding". In: *arXiv preprint arXiv:2505.15700* (2025).

[146] Yekaterina Kovaleva, Ari Happonen, and Eneli Kindsiko. "Designing gender-neutral software engineering program. stereotypes, social pressure, and current attitudes based on recent studies". In: *Proceedings of the Third Workshop on Gender Equality, Diversity, and Inclusion in Software Engineering*. GE@ICSE '22. New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 43–50. ISBN: 978-1-4503-9294-5. DOI: 10.1145/3524501.3527600. URL: https://dl.acm.org/doi/10.1145/3524501.3527600 (visited on 05/12/2024).

[147] Yekaterina Kovaleva, Ari Happonen, and Audrey Mbogho. "Towards gender balance in modern hackathons: literature-based approaches for female inclusiveness". In: *Proceedings of the Third Workshop on Gender Equality, Diversity, and Inclusion in Software Engineering*. GE@ICSE '22. New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 19–26. ISBN: 978-1-4503-9294-5. DOI: 10.1145/3524501.3527594. URL: https://dl.acm.org/doi/10.1145/3524501.3527594 (visited on 05/12/2024).

[148] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. Tech. rep. University of Toronto, 2009. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[149] Damir Krstinić et al. "Multi-label classifier performance evaluation with confusion matrix". In: *Comput Sci Inf Technol* 10 (2020), pp. 1–14.

[150] S. Y. Kung. *Kernel Methods and Machine Learning*. 2014.

[151] Meghdad Kurmanji et al. "Towards unbounded machine unlearning". In: *Advances in neural information processing systems* 36 (2024).

[152] Matt J Kusner et al. "Counterfactual Fairness". In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. (Visited on 12/16/2021).

[153] Yusuke Kuwana et al. "Black-Box Forgetting". In: *Proc. of Conf. on Advances in Neural Information Processing Systems (NeurIPS)*. 2024.

[154] Chanhee Kwak et al. "Let Machines Unlearn - Machine Unlearning and the Right to be Forgotten". In: *Proc. of Americas Conf. on Information Systems (AMCIS)*. 2017.

[155] Tai Le Quy et al. "A survey on datasets for fairness-aware machine learning". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 12.3 (2022), e1452.

[156] Omri Lev and Ashia Wilson. *Faster Machine Unlearning via Natural Gradient Descent*. 2024. arXiv: 2407.08169 [cs.LG]. URL: https://arxiv.org/abs/2407.08169.

[157] Chunxiao Li et al. "An overview of machine unlearning". In: *High-Confidence Computing* 5.2 (2025), p. 100254. ISSN: 2667-2952. DOI: https://doi.org/10.1016/j.hcc.2024.100254. URL: https://www.sciencedirect.com/science/article/pii/S2667295224000576.

[158] Xiang Li, Bhavani Thuraisingham, and Wenqi Wei. *MUBox: A Critical Evaluation Framework of Deep Machine Unlearning*. 2025. arXiv: 2505.08576 [cs.LG]. URL: https://arxiv.org/abs/2505.08576.

[159] Xunkai Li et al. *Toward Scalable Graph Unlearning: A Node Influence Maximization based Approach*. 2025. arXiv: 2501.11823 [cs.LG]. URL: https://arxiv.org/abs/2501.11823.

[160] Xunkai Li et al. "Towards effective and general graph unlearning via mutual evolution". In: *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN: 978-1-57735-887-9. DOI: 10.1609/aaai.v38i12.29273. URL: https://doi.org/10.1609/aaai.v38i12.29273.

[161] Yuyuan Li et al. "UltraRE: Enhancing RecEraser for Recommendation Unlearning via Error Decomposition". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 12611–12625. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/29a0ea49a103a233b17c0705cdeccb66-Paper-Conference.pdf.

[162] Tjen-Sien Lim, Wei-Yin Loh, and Yu-Shan Shih. "A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms". In: *Machine learning* 40.3 (2000), pp. 203–228.

[163] Hengzhu Liu et al. "A survey on machine unlearning: Techniques and new emerged privacy risks". In: *Journal of Information Security and Applications* 90 (2025), p. 104010. ISSN: 2214-2126. DOI: https://doi.org/10.1016/j.jisa.2025.104010. URL: https://www.sciencedirect.com/science/article/pii/S2214212625000481.

[164] Yufang Liu et al. *Unlearning with Fisher Masking*. 2023. arXiv: 2310.05331 [cs.LG]. URL: https://arxiv.org/abs/2310.05331.

[165] Ziwei Liu et al. "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[166] Alexander Ly et al. "A Tutorial on Fisher information". In: *Journal of Mathematical Psychology* 80 (2017), pp. 40–55.

[167] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[168] Sean MacAvaney et al. "Efficient Document Re-Ranking for Transformers by Precomputing Term Representations". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Virtual Event, China: Association for Computing Machinery, 2020, pp. 49–58. ISBN: 9781450380164. DOI: 10.1145/3397271.3401093. URL: https://doi.org/10.1145/3397271.3401093.

[169] Sean MacAvaney et al. *Simplified Data Wrangling with ir_datasets*. July 2021. DOI: 10.1145/3404835.3463254. URL: http://dx.doi.org/10.1145/3404835.3463254.

[170] Jan Machacek et al. *Introducing Inversion of Control*. 2008.

[171] Raghunath Reddy Madireddy and Apurva Mudgal. "A Constant-Factor Approximation Algorithm for Red-Blue Set Cover with Unit Disks". In: *Algorithmica* 85.1 (2023), pp. 100–132.

[172] maharshipandya. *Spotify Tracks Dataset*. https://huggingface.co/datasets/maharshipandya/spotify-tracks-dataset. 2023.

[173] Pratyush Maini et al. *Tofu: A task of fictitious unlearning for llms*. 2024.

[174] Alessandro Mantelero. "The EU Proposal for a General Data Protection Regulation and the roots of the 'right to be forgotten'". In: *Computer Law & Security Review* 29.3 (2013), pp. 229–235.

[175] Kai Marquardt, Ingo Wagner, and Lucia Happe. "Engaging Girls in Computer Science: Do Single-Gender Interdisciplinary Classes Help?" In: *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. 2023, pp. 128–140. DOI: 10.1109/ICSE-SEET58685.2023.00019.

[176] Juliette Marrie et al. *On Good Practices for Task-Specific Distillation of Large Pretrained Visual Models*. 2024. arXiv: 2402.11305 [cs.CV]. URL: https://arxiv.org/abs/2402.11305.

[177] Andrew Kachites McCallum et al. "Automating the Construction of Internet Portals with Machine Learning". In: *Proceedings of the 17th International Conference on Machine Learning (ICML)*. 2000, pp. 327–334.

[178] John H McDonald. *One-way ANOVA*. Vol. 2. sparky house publishing Baltimore, MD, 2009.

[179] L. Meenachi et al. "Multi Class Ensemble Classification for Crop Recommendation". In: *2022 International Conference on Inventive Computation Technologies (ICICT)*. ISSN: 2767-7788. July 2022, pp. 1319–1324. DOI: 10.1109/ICICT54344.2022.9850561.

[180] Ninareh Mehrabi et al. "A Survey on Bias and Fairness in Machine Learning". en. In: *ACM Computing Surveys* 54.6 (July 2021), pp. 1–35. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3457607. (Visited on 11/11/2021).

[181] Lang Mei et al. "Learning Probabilistic Box Embeddings for Effective and Efficient Ranking". In: *Proceedings of the ACM Web Conference 2022*. WWW '22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022, pp. 473–482. ISBN: 9781450390965. DOI: 10.1145/3485447.3512073. URL: https://doi.org/10.1145/3485447.3512073.

[182] Scott Menard. *Applied logistic regression analysis*. Vol. 106. Sage, 2002.

[183] Friederike Mengel, Jan Sauermann, and Ulf Zölitz. "Gender Bias in Teaching Evaluations". In: *Journal of the European Economic Association* 17.2 (Apr. 2019), pp. 535–566. ISSN: 1542-4766. DOI: 10.1093/jeea/jvx057. (Visited on 09/21/2022).

[184] Steven John Metsker. *Design patterns Java workbook*. 2002.

[185] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. URL: http://arxiv.org/abs/1301.3781.

[186] Brent Mittelstadt. "Principles Alone Cannot Guarantee Ethical AI". In: *Nature Machine Intelligence* 1.11 (2019), pp. 501–507.

[187] Corinne A Moss-Racusin et al. "Science faculty's subtle gender biases favor male students". In: *Proceedings of the national academy of sciences* 109.41 (2012), pp. 16474–16479.

[188] Simona Motogna et al. "Retaining women in computer science: the good, the bad and the ugly sides". In: *Proceedings of the Third Workshop on Gender Equality, Diversity, and Inclusion in Software Engineering*. GE@ICSE '22. New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 35–42. ISBN: 978-1-4503-9294-5. DOI: 10.1145/3524501.3527598. URL: https://dl.acm.org/doi/10.1145/3524501.3527598 (visited on 05/12/2024).

[189] United Nations. *THE 17 GOALS | Sustainable Development*. 2015. URL: https://sdgs.un.org/goals (visited on 03/31/2022).

[190] Angela Nebot and Francisco Mugica. "Evolution of the Participation of Women in University Computer Science Studies in Spain and in Europe". In: *2023 IEEE Frontiers in Education Conference (FIE)*. 2023, pp. 1–8. DOI: 10.1109/FIE58773.2023.10343410.

[191] Aviraj Newatia, Michael Cooper, and Rahul Krishnan. "Unlearning Tabular Data Without a "Forget Set"". In: *Proc. of the Third Table Representation Learning Workshop @NeurIPS 2024*. 2024.

[192] M. E. J. Newman. *Networks: an introduction*. Oxford; New York: Oxford University Press, 2010. URL: http://www.amazon.com/Networks-An-Introduction-Mark-Newman/dp/0199206651/ref=sr_1_5?ie=UTF8&qid=1352896678&sr=8-5&keywords=complex+networks.

[193] Mark EJ Newman. "Modularity and community structure in networks". In: *Proceedings of the national academy of sciences* 103.23 (2006), pp. 8577–8582.

[194] Thanh Tam Nguyen et al. "A survey of machine unlearning". In: *arXiv:2209.02299* (2022).

[195] William S Noble. "What is a support vector machine?" In: *Nature biotechnology* 24.12 (2006). Publisher: Nature Publishing Group, pp. 1565–1567.

[196] Alexandra Olteanu et al. "Social data: Biases, methodological pitfalls, and ethical boundaries". In: *Frontiers in Big Data* 2 (2019), p. 13.

[197] J. A. Olvera-López et al. "A review of instance selection methods". In: *Artificial Intelligence Review* 34.2 (2010), pp. 133–143. DOI: 10.1007/s10462-010-9165-y. URL: https://doi.org/10.1007/s10462-010-9165-y.

[198] Organisation for Economic Co-operation and Development. *OECD Principles on Artificial Intelligence*. Accessed: 2025-10-30. 2019. URL: https://oecd.ai/en/ai-principles.

[199] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* (2011).

[200] Joelle Pineau et al. "Improving reproducibility in machine learning research: a report from the NeurIPS 2019 reproducibility program". In: *Journal of Machine Learning Research* 22 (2021).

[201] Preston Putzel and Scott Lee. "Blackbox Post-Processing for Multiclass Fairness". In: *arXiv:2201.04461 [cs]* (Jan. 2022). arXiv: 2201.04461. URL: http://arxiv.org/abs/2201.04461 (visited on 01/27/2022).

[202] Huilian Sophie Qiu et al. "Gender Representation Among Contributors to Open-Source Infrastructure : An Analysis of 20 Package Manager Ecosystems". In: *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. ISSN: 2832-7616. May 2023, pp. 180–187. DOI: 10.1109/ICSE-SEIS58686.2023.00025. (Visited on 05/12/2024).

[203] Youyang Qu et al. *Learn to Unlearn: A Survey on Machine Unlearning*. 2023. arXiv: 2305.07512 [cs.LG]. URL: https://arxiv.org/abs/2305.07512.

[204] Alec Radford and Karthik Narasimhan. *Improving Language Understanding by Generative Pre-Training*. 2018.

[205] Sandro Radovanović et al. "A fair classifier chain for multi-label bank marketing strategy classification". en. In: *International Transactions in Operational Research* (2021). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/itor.13059. ISSN: 1475-3995. DOI: 10.1111/itor.13059. (Visited on 12/01/2021).

[206] Chotirat Ann Ratanamahatana and Dimitrios Gunopulos. "Scaling up the naive Bayesian classifier: Using decision trees for feature selection". In: (2002).

[207] Michael Redmond and Alok Baveja. "A data-driven software tool for enabling cooperative information sharing among police departments". In: *European Journal of Operational Research* 141.3 (2002), pp. 660–678.

[208] Payam Refaeilzadeh, Lei Tang, and Huan Liu. "Cross-Validation". In: *Encyclopedia of Database Systems*. New York, NY: Springer New York, 2016, pp. 1–7. ISBN: 978-1-4899-7993-3. DOI: 10.1007/978-1-4899-7993-3_565-2.

[209] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". 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)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410. URL: https://aclanthology.org/D19-1410.

[210] Ministero dell'Istruzione dell'Università e della Ricerca. *Cerca Università*. 2023. URL: http://cercauniversita.cineca.it/php5/docenti/cerca.php.

[211] Stephen Robertson and Hugo Zaragoza. "The Probabilistic Relevance Framework: BM25 and Beyond". In: *Foundations and Trends® in Information Retrieval* 3.4 (2009), pp. 333–389. ISSN: 1554-0669. DOI: 10.1561/1500000019. URL: http://dx.doi.org/10.1561/1500000019.

[212] Stephen Robertson et al. *Okapi at TREC-3*. Jan. 1995. URL: https://www.microsoft.com/en-us/research/publication/okapi-at-trec-3/.

[213] G.H. Rosenfield and K. Fitzpatrick-Lins. "A coefficient of agreement as a measure of thematic classification accuracy." In: *Photogrammetric Engineering and Remote Sensing* 52.2 (1986), pp. 223–227. URL: http://pubs.er.usgs.gov/publication/70014667.

[214] M Teresa Ruiz and Lois M Verbrugge. "A two way view of gender bias in medicine." In: *Journal of epidemiology and community health* 51.2 (1997), p. 106.

[215] Anwar Said et al. *A Survey of Graph Unlearning*. 2024. arXiv: 2310.02164 [cs.LG]. URL: https://arxiv.org/abs/2310.02164.

[216] Hiroshi Sakiyama, Masaki Fukuda, and Yasushi Okuno. "Prediction of Blood-Brain Barrier Penetration (BBBP) Based on Molecular Descriptors of the Free-Form and In-Blood-Form Datasets". In: *Molecules* 26.24 (Dec. 2021), p. 7428. DOI: 10.3390/molecules26247428.

[217] G. Salton, A. Wong, and C. S. Yang. "A Vector Space Model for Automatic Indexing". In: *Commun. ACM* 18.11 (Nov. 1975), pp. 613–620. ISSN: 0001-0782. DOI: 10.1145/361219.361220. URL: https://doi.org/10.1145/361219.361220.

[218] G. Salton, A. Wong, and C. S. Yang. "A vector space model for automatic indexing". In: *Commun. ACM* 18.11 (Nov. 1975), pp. 613–620. ISSN: 0001-0782. DOI: 10.1145/361219.361220. URL: https://doi.org/10.1145/361219.361220.

[219] Claude Sammut and Geoffrey I. Webb, eds. *TF–IDF*. Boston, MA, 2010. DOI: 10.1007/978-0-387-30164-8_832. URL: https://doi.org/10.1007/978-0-387-30164-8_832.

[220] Victor Sanh et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2020. arXiv: 1910.01108 [cs.CL]. URL: https://arxiv.org/abs/1910.01108.

[221] Priscila Santiesteban, Madeline Endres, and Westley Weimer. "An analysis of sex differences in computing teaching evaluations". In: *Proceedings of the Third Workshop on Gender Equality, Diversity, and Inclusion in Software Engineering*. GE@ICSE '22. New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 84–87. ISBN: 978-1-4503-9294-5. DOI: 10.1145/3524501.3527604. URL: https://dl.acm.org/doi/10.1145/3524501.3527604 (visited on 05/12/2024).

[222] Jari Saramäki et al. "Generalizations of the clustering coefficient to weighted complex networks". In: *Phys. Rev. E* 75 (2 Feb. 2007), p. 027105. DOI: 10.1103/PhysRevE.75.027105. URL: https://link.aps.org/doi/10.1103/PhysRevE.75.027105.

[223] Lesley A Schimanski and Juan Pablo Alperin. "The evaluation of scholarship in academic promotion and tenure processes: Past, present, and future". In: *F1000Research* 7 (2018).

[224] Google Scholar. *Google Scholar*. 2023. URL: https://scholar.google.com/.

[225] Christopher Sciavolino et al. *Simple Entity-Centric Questions Challenge Dense Retrievers*. English (US). 2021.

[226] Oleksandr Shchur et al. *Pitfalls of Graph Neural Network Evaluation*. 2019. arXiv: 1811.05868 [cs.LG]. URL: https://arxiv.org/abs/1811.05868.

[227] Yun Shen et al. "Finding MNEMON: Reviving Memories of Node Embeddings". In: *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. CCS '22. Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 2643–2657. ISBN: 9781450394505. DOI: 10.1145/3548606.3559358. URL: https://doi.org/10.1145/3548606.3559358.

[228] Weijia Shi et al. *Muse: Machine unlearning six-way evaluation for language models*. 2024.

[229] Takashi Shibata et al. "Learning with Selective Forgetting". In: *Proc. of Int. Joint Conf. on Artificial Intelligence (IJCAI)*. 2021, pp. 989–996.

[230] Reza Shokri et al. "Membership inference attacks against machine learning models". In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18. DOI: 10.1109/SP.2017.41.

[231] Sara Nadiv Soffer and Alexei Vazquez. "Network clustering coefficient without degree-correlation biases". In: *Physical Review E* 71.5 (2005), p. 057101.

[232] Alexandra Souly et al. *Poisoning Attacks on LLMs Require a Near-constant Number of Poison Samples*. 2025. arXiv: 2510.07192 [cs.LG]. URL: https://arxiv.org/abs/2510.07192.

[233] Alexandra Souly et al. *Poisoning Attacks on LLMs Require a Near-constant Number of Poison Samples*. 2025. arXiv: 2510.07192 [cs.LG]. URL: https://arxiv.org/abs/2510.07192.

[234] Oumaima Stitini, Soulaimane Kaloun, and Omar Bencharef. "Integrating contextual information into multi-class classification to improve the context-aware recommendation". en. In: *Procedia Computer Science*. 12th International Conference on Emerging Ubiquitous Systems and Pervasive Networks / 11th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare 198 (Jan. 2022), pp. 311–316. ISSN: 1877-0509. DOI: 10.1016/j.procs.2021.12.246. URL: https://www.sciencedirect.com/science/article/pii/S1877050921024856 (visited on 11/11/2022).

[235] W Nick Street, William H Wolberg, and Olvi L Mangasarian. "Nuclear feature extraction for breast tumor diagnosis". In: *Biomedical image processing and biomedical visualization*. Vol. 1905. International Society for Optics and Photonics. 1993, pp. 861–870.

[236] M. S. Suchithra and Maya L. Pai. "Improving the Performance of Sigmoid Kernels in Multiclass SVM Using Optimization Techniques for Agricultural Fertilizer Recommendation System". en. In: *Soft Computing Systems*. Ed. by Ivan Zelinka et al. Communications in Computer and Information Science. Singapore: Springer, 2018, pp. 857–868. ISBN: 9789811319365. DOI: 10.1007/978-981-13-1936-5_87.

[237] Harini Suresh and John V Guttag. "A framework for understanding unintended consequences of machine learning". In: *arXiv preprint arXiv:1901.10002* 2 (2019), p. 8.

[238] Anna Szlavi et al. "Bridging Values: The Inclusion of Young Generations in Computing". In: *Universal Access in Human-Computer Interaction*. Ed. by Margherita Antona and Constantine Stephanidis. Cham: Springer Nature Switzerland, 2023, pp. 154–170. ISBN: 978-3-031-35681-0.

[239] Noshin Tahsin et al. "Can female underrepresentation in information technology be solved through an awareness-based approach?" In: *Proceedings of the Third Workshop on Gender Equality, Diversity, and Inclusion in Software Engineering*. GE@ICSE '22. New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 1–5. ISBN: 978-1-4503-9294-5. DOI: 10.1145/3524501.3527606. URL: https://dl.acm.org/doi/10.1145/3524501.3527606 (visited on 05/12/2024).

[240] Noshin Tahsin et al. "Cognitive Distance and Women in Software Engineering: An Empirical Study in the Context of Bangladesh". In: *2023 IEEE/ACM 4th Workshop on Gender Equity, Diversity, and Inclusion in Software Engineering (GEICSE)*. May 2023, pp. 1–8. DOI: 10.1109/GEICSE59319.2023.00005. URL: https://ieeexplore.ieee.org/abstract/document/10213433 (visited on 05/12/2024).

[241] Jiajun Tan et al. "Unlink to Unlearn: Simplifying Edge Unlearning in GNNs". In: *Companion Proceedings of the ACM Web Conference 2024*. WWW '24. Singapore, Singapore: Association for Computing Machinery, 2024, pp. 489–492. ISBN: 9798400701726. DOI: 10.1145/3589335.3651578. URL: https://doi.org/10.1145/3589335.3651578.

[242] Ayush K Tarun et al. "Fast yet effective machine unlearning". In: *IEEE Transactions on Neural Networks and Learning Systems* (2023). DOI: 10.1109/TNNLS.2023.3266233.

[243] Nandan Thakur et al. *BEIR: A Heterogenous Benchmark for Zero-shot Evaluation of Information Retrieval Models*. 2021. arXiv: 2104.08663 [cs.IR].

[244] Kai Ming Ting. *Precision and Recall*. Boston, MA, 2010. DOI: 10.1007/978-0-387-30164-8_652. URL: https://doi.org/10.1007/978-0-387-30164-8_652.

[245] *TorchVision: PyTorch's computer vision library*. https://pytorch.org/vision/. 2024.

[246] Bianca Trinkenreich et al. "An empirical investigation on the challenges faced by women in the software industry: a case study". In: *Proceedings of the 2022 ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society*. ICSE-SEIS '22. New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 24–35. ISBN: 978-1-4503-9227-3. DOI: 10.1145/3510458.3513018. URL: https://dl.acm.org/doi/10.1145/3510458.3513018 (visited on 05/12/2024).

[247] Athanasios Tsanas et al. "Accurate telemonitoring of Parkinson's disease progression by non-invasive speech tests". en. In: *Nature Precedings* (Oct. 2009). Publisher: Nature Publishing Group, pp. 1–1. ISSN: 1756-0357. DOI: 10.1038/npre.2009.3920.1. URL: https://www.nature.com/articles/npre.2009.3920.1 (visited on 02/23/2022).

[248] Zhengkai Tu et al. "Approximate Nearest Neighbor Search and Lightweight Dense Vector Reranking in Multi-Stage Retrieval Architectures". In: *Proceedings of the 2020 ACM SIGIR on International Conference on Theory of Information Retrieval*. ICTIR '20. Virtual Event, Norway: Association for Computing Machinery, 2020, pp. 97–100. ISBN: 9781450380676. DOI: 10.1145/3409256.3409818. URL: https://doi.org/10.1145/3409256.3409818.

[249] UNESCO. *Recommendation on the Ethics of Artificial Intelligence*. Adopted on 24 November 2021. 2021. URL: https://unesdoc.unesco.org/ark:/48223/pf0000380455.

[250] United Nations. *The 17 Goals – Sustainable Development Goals*. Accessed: 2025-12-03. 2025. URL: https://sdgs.un.org/goals.

[251] L. G. Valiant. "A theory of the learnable". In: *Commun. ACM* 27.11 (Nov. 1984), pp. 1134–1142. ISSN: 0001-0782. DOI: 10.1145/1968.1972. URL: https://doi.org/10.1145/1968.1972.

[252] V. N. Vapnik and A. Ya. Chervonenkis. "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities". In: *Measures of Complexity: Festschrift for Alexey Chervonenkis*. Ed. by Vladimir Vovk, Harris Papadopoulos, and Alexander Gammerman. Cham: Springer International Publishing, 2015, pp. 11–30. ISBN: 978-3-319-21852-6. DOI: 10.1007/978-3-319-21852-6_3.

[253] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[254] Sahil Verma and Julia Rubin. "Fairness definitions explained". In: *2018 ieee/acm international workshop on software fairness (fairware)*. IEEE. 2018, pp. 1–7.

[255] Sandra Wachter, Brent Mittelstadt, and Luciano Floridi. "Why a Right to Explanation of Automated Decision-Making Does Not Exist in the General Data Protection Regulation". In: *International Data Privacy Law* 7.2 (2017), pp. 76–99.

[256] Cheng-Long Wang, Mengdi Huai, and Di Wang. "Inductive graph unlearning". In: *Proceedings of the 32nd USENIX Conference on Security Symposium*. SEC '23. Anaheim, CA, USA: USENIX Association, 2023. ISBN: 978-1-939133-37-3.

[257] Song Wang et al. "Knowledge Editing for Large Language Models: A Survey". In: *ACM Computing Surveys (CSUR)* 57.3 (2025), 59:1–59:37.

[258] Yi Wang, Xinyue Zhang, and Wei Wang. "Fundamentalists, Integrationists, & Transformationists: An Empirical Theory of Men Software Engineers' Orientations in Gender Inequalities". In: *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. ISSN: 2832-7616. May 2023, pp. 25–36. DOI: 10.1109/ICSE-SEIS58686.2023.00009. URL: https://ieeexplore.ieee.org/abstract/document/10173887?casa_token=p29o3o734oAAAAAA:E2YGtbdDEjyRo07ltQZMVB520WG85DnT2xqlgNzIC7UJCuf_rYGVn2lax4vep6s5EeUnppJK (visited on 05/12/2024).

[259] Zhenyi Wang et al. "A Comprehensive Survey of Forgetting in Deep Learning Beyond Continual Learning". en. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 47.3 (Mar. 2025), pp. 1464–1483. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2024.3498346. URL: https://ieeexplore.ieee.org/document/10752992/ (visited on 11/21/2025).

[260] Claes Wohlin et al. *Experimentation in Software Engineering*. en. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. DOI: 10.1007/978-3-642-29044-2. URL: http://link.springer.com/10.1007/978-3-642-29044-2 (visited on 03/13/2024).

[261] David H Wolpert. *What does dinner cost?* en. 1999. URL: http://www.no-free-lunch.org/coev.pdf.

[262] Jiancan Wu et al. "GIF: A General Graph Unlearning Strategy via Influence Function". In: *Proceedings of the ACM Web Conference 2023*. WWW '23. Austin, TX, USA: Association for Computing Machinery, 2023, pp. 651–661. ISBN: 9781450394161. DOI: 10.1145/3543507.3583521. URL: https://doi.org/10.1145/3543507.3583521.

[263] Kun Wu et al. "Certified Edge Unlearning for Graph Neural Networks". In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. KDD '23. Long Beach, CA, USA: Association for Computing Machinery, 2023, pp. 2606–2617. ISBN: 9798400701030. DOI: 10.1145/3580305.3599271. URL: https://doi.org/10.1145/3580305.3599271.

[264] Zhenqin Wu et al. "MoleculeNet: a benchmark for molecular machine learning". In: *Chemical Science* 9.2 (2018). Open Access, pp. 513–530. DOI: 10.1039/C7SC02664A. URL: https://pubs.rsc.org/en/content/articlelanding/2018/sc/c7sc02664a.

[265] Heng Xu et al. "Machine Unlearning: A Survey". In: *ACM Computing Surveys (CSUR)* 56.1 (2024), 9:1–9:36.

[266] Zhichao Xu et al. *Counterfactual Editing for Search Result Explanation*. 2023. arXiv: 2301.10389 [cs.IR].

[267] Nacim Yanes et al. "A machine learning-based recommender system for improving students learning experiences". In: *IEEE Access* 8 (2020), pp. 201218–201235.

[268] Chenxiao Yang et al. *Graph Neural Networks are Inherently Good Generalizers: Insights by Bridging GNNs and MLPs*. 2023. arXiv: 2212.09034 [cs.LG]. URL: https://arxiv.org/abs/2212.09034.

[269] Zhe-Rui Yang et al. "Erase Then Rectify: A Training-Free Parameter Editing Approach for Cost-Effective Graph Unlearning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 39.12 (Apr. 2025), pp. 13044–13051. DOI: 10.1609/aaai.v39i12.33423. URL: https://ojs.aaai.org/index.php/AAAI/article/view/33423.

[270] Jingwen Ye et al. "Learning with Recoverable Forgetting". In: *Proc. of European Conf. on Computer Vision (ECCV)*. 2022, pp. 87–103.

[271] Chih-Kuan Yeh et al. "Representer Point Selection for Explaining Deep Neural Networks". In: *Proc. of Conf. on Advances in Neural Information Processing Systems (NeurIPS)*. 2018, pp. 9311–9321.

[272] Lu Yi and Zhewei Wei. *Scalable and Certifiable Graph Unlearning: Overcoming the Approximation Error Barrier*. Poster presented at International Conference on Learning Representations (ICLR) 2025. 2025. URL: https://iclr.cc/virtual/2025/poster/28301.

[273] Charles Yu et al. "Unlearning Bias in Language Models by Partitioning Gradients". In: *Findings of the Association for Computational Linguistics: ACL*. 2023, pp. 6032–6048.

[274] Jiahao Zhang. "Graph Unlearning with Efficient Partial Retraining". In: *Companion Proceedings of the ACM Web Conference 2024*. WWW '24. Singapore, Singapore: Association for Computing Machinery, 2024, pp. 1218–1221. ISBN: 9798400701726. DOI: 10.1145/3589335.3651265. URL: https://doi.org/10.1145/3589335.3651265.

[275] Jing Zhang et al. "On the application of multi-class classification in physical therapy recommendation". en. In: *Health Information Science and Systems* 1.1 (Dec. 2013), p. 15. ISSN: 2047-2501. DOI: 10.1186/2047-2501-1-15. URL: https://doi.org/10.1186/2047-2501-1-15 (visited on 11/14/2022).

[276] Liangwei Zhang, Jing Lin, and Ramin Karim. "Adaptive kernel density-based anomaly detection for nonlinear systems". In: *Knowledge-Based Systems* 139 (2018), pp. 50–63. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2017.10.009. URL: https://www.sciencedirect.com/science/article/pii/S0950705117304707.

[277] Ruqing Zhang et al. "Aggregating Neural Word Embeddings for Document Representation". In: *Advances in Information Retrieval*. Ed. by Gabriella Pasi et al. Cham: Springer International Publishing, 2018, pp. 303–315. ISBN: 978-3-319-76941-7.

[278] Ruqing Zhang et al. "Spherical Paragraph Model". In: *Advances in Information Retrieval*. Ed. by Gabriella Pasi et al. Cham: Springer International Publishing, 2018, pp. 289–302. ISBN: 978-3-319-76941-7.

[279] Kairan Zhao et al. "What makes unlearning hard and what to do about it". In: *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*. 2024.

[280] Xin Zhao and Riley Young. "Workplace Discrimination in Software Engineering: Where We Stand Today". In: *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. ISSN: 2832-7616. May 2023, pp. 188–193. DOI: 10.1109/ICSE-SEIS58686.2023.00026. (Visited on 05/12/2024).