UNIVERISTY OF MACEDONIA
SCHOOL OF INFORMATION SCIENCES
DEPARTMENT OF APPLIED INFORMATICS

TOWARDS GREENER AI FOR PRIVACY PRESERVING RECORD LINKAGE

Bachelor’s Thesis

of

Emmanouil Sokorelis

Thessaloniki, February 2025

TOWARDS GREENER AI FOR PRIVACY PRESERVING RECORD LINKAGE

Emmanouil Sokorelis

Undergraduate Students of Applied Informatics at the University of Macedonia

Bachelor’s Thesis

Submitted for the partial fulfillment of its requirements

BSc in Applied Informatics, Computer Science & Technology

Supervisor
Alexandros Karakasidis

Approved by the three-member examination committee 13/02/2025

Alexandros Karakasidis	Eftychios Protopapadakis	Georgia Koloniari
....	....	....

Emmanouil Sokorelis

....

Abstract

Artificial intelligence systems, especially those based on deep neural networks, require significant energy resources, increasing the need to balance performance with sustainability, a field known as Green AI. This thesis investigates the intersection of accuracy, privacy, and energy efficiency within deep learning-based, privacy-preserving entity resolution process. Entity resolution is a crucial data management task, widely applied in fields like healthcare and e-commerce, which often involve large-scale data processing with significant environmental impacts. The integration of differential privacy and Bloom filters introduces challenges related to maintaining privacy while optimizing energy consumption. This research aims to strike a balance between these competing demands by exploring the energy costs associated with privacy protection and evaluating the trade-offs between model performance, confidentiality, and environmental sustainability. Through a series of experiments, the thesis explores the effects of key parameters on the efficiency of ER workflows, examining how modifications to encoding, differential noise addition, similarity metrics, and deep learning model configurations influence accuracy, privacy, and energy efficiency. The findings provide valuable insights into optimizing ER processes for privacy and sustainability without sacrificing performance, offering new avenues for environmentally responsible AI development.

Keywords: Green AI, Entity Resolution, Differential Privacy, Bloom Filters, Differential Privacy Bloom Filters, Deep Learning, Energy Efficiency, Performance Optimization.

Acknowledgement

I would like to express my gratitude to my professor and supervisor, Dr. Alexandros Karakasidis, for his support and guidance throughout the project. I am deeply thankful for his genuine interest in the subject, as well as his willingness to assist at every step. His support has been a significant source of motivation, and I am truly grateful for the opportunity to work under his mentorship.

Contents

List of Figures

List of Tables

1 Introduction

This chapter presents the introduction of the thesis, beginning with the motivation for the research. The concept of Green AI forms greater discussion. The chapter also frames the relevance of entity resolution, which is an important data management task that has applications in fields which deal with large-scale data with considerable energy demands. The interplay between accuracy, privacy and energy efficiency is discussed in detail. This work specifically considers the aspect of differential privacy, a framework that perturbs data with noise to guarantee confidentiality while allowing useful data analysis. The thesis delves into differential privacy Bloom filters, an extension of traditional Bloom filters, which integrate differential privacy to enhance data security. The chapter concludes by determining what the objectives and scope of the research are. This indicates the aim to assess and optimize the various trade-offs between accuracy, privacy and energy efficiency within a novel deep learning-based, privacy-preserving entity resolution process. Therefore, this chapter is one that introduces the problem at hand but also frames the questions and areas on which the rest of the thesis will dwell.

1.1 Motivation

Artificial intelligence systems, especially those who rely on deep learning, need huge computational resources, which in turn usually result in high energy consumption and environmental impact. For example, training large neural networks can emit carbon dioxide equivalent to the amount produced by several intercontinental flights. These challenges have triggered discussions related to the concept of Green AI that tries to balance environmental sustainability with performance. Entity resolution is a critical data management task which targets areas such as healthcare, e-commerce and data management systems, often requiring large-scale data processing, making its environmental impact a major issue.

When data privacy must be ensured in ER workflows, an additional layer of complexity arises. Differential privacy is one such rigorous framework that helps guarantee the confidentiality of single records while enabling data analysis. One of the popular variants for privacy-preserving ER workflows includes adding differential noise to the data encoding procedures of Bloom filters. However, privacy-preserving techniques such as differential privacy introduce additional computational overhead, which can increase power consumption.

Differential privacy Bloom filters extend the classic Bloom filters by adding differential privacy mechanisms for sensitive data protection while performing efficient similarity comparisons. Differential privacy Bloom filters add noise to the data encoding process to ensure privacy, making them particularly useful in privacy-preserving applications. Optimizing the noise level is crucial to maintain both the balance between the effectiveness of the ER process and the safeguarding of sensitive information.

1.2 Objectives and Scope

The main objective of this thesis is to investigate the convergence of accuracy, privacy and energy efficiency within a novel deep learning-based, privacy preserving entity resolution process. More precisely, this thesis aims to investigate the energy consumption impacts resulting from the integration of differential privacy and deep learning into an ER workflow that involves Bloom filters. The trade-off will be to find an optimal balance between high accuracy, preserving privacy and reducing the environmental footprint of computing resource usage.

The scope of this thesis is confined to analyzing CPU energy consumption and total RAM usage of the investigated process. Additionally, it investigates barriers to energy efficiency and explores potential optimizations to improve sustainability without compromising performance. The key phases under examination include encoding data into Bloom filters, introducing differential noise, computing similarity and difference metrics among filter pairs, and training and evaluating deep learning classifiers using these scores.

1.3 Thesis Structure

The rest of the thesis is organized as follows: In Chapter 2, we review related work in entity resolution, privacy-preserving techniques, and Green AI. Chapter 3 introduces the deep learning-based, privacy preserving entity resolution process, covering the encoding of data using Bloom filters, the application of differential privacy, and the process of generating similarity features for deep learning model training and performance evaluation. In Chapter 4, we outline the methodology used in this study, describing the datasets, performance evaluation approaches, parameters, and the tools used to measure energy consumption. Chapter 5 presents experimental results evaluating the impact of key parameters on Bloom filters and the ER process, exploring the effects of these parameters on accuracy, privacy, and energy efficiency. Chapter 6 discusses further experimental results, including few-shot learning and cross-dataset evaluations, examining how smaller datasets and cross-dataset applications affect energy consumption and performance. Finally, Chapter 7 concludes the thesis by summarizing the findings, discussing limitations, and suggesting future research directions to further optimize the balance between accuracy, privacy, and energy efficiency.

2 Background and Related Work

This chapter offers a summary of the background and related research for this thesis. The first section presents an overview of ER, emphasizing its significance across various industries. This section also discusses the challenges faced in this area. The second section evaluates traditional ER techniques, classifying them into rule-based, supervised learning, unsupervised learning and hybrid methods. It also outlines the benefits and drawbacks of these strategies. The third part concentrates on modern ER methods, including deep learning, semi-supervised learning, probabilistic models, graph-based models, transfer learning and few-shot learning approaches. These techniques are particularly crucial for tackling some of the issues associated with traditional ER methods. The fourth section addresses privacy in entity resolution, a subject that is gaining prominence. Numerous tactics, including data anonymization and privacy-preserving methods have been established to safeguard privacy during the ER processes. Finally, the fifth section emphasizes Green AI and environmentally friendly methods. Green AI aims to reduce the environmental impact of AI technologies while maintaining their efficiency and effectiveness. This section explores strategies to save energy during AI model training, decrease carbon footprints through renewable energy and uphold responsible data management. It also investigates how AI can support environmental sustainability and discusses ethical issues, such as the importance of transparency, fairness and responsible practices in AI development.

2.1 Overview of Entity Resolution

Entity resolution, often referred to as record linkage, duplication, or entity matching, is a critical procedure in data management. The main goal of ER is to guarantee that each unique entity is accurately represented across different databases. This procedure is critical for handling large datasets from multiple sources, as entities may be recorded in different formats due to data entry errors, naming differences, or formatting issues. Entity resolution is significant in numerous fields where preserving data integrity and quality is important. For example, in customer relationship management systems, entity resolution processes help consolidate all records linked to a particular customer into a single profile, improving customer service and operational efficiency. Similarly, in the healthcare sector, entity resolution is essential for precisely connecting patient records from different hospitals or clinics.

The process of entity resolution may consist of several key phases. First, data preprocessing is performed to clean the data. This phase may involve steps such as addressing missing values and fixing data inconsistencies. The next phase is called blocking or indexing, which aims to reduce the computational load by grouping records that are likely to match into blocks. Blocking uses common attributes like name, location, or date of birth to decrease the number of record pairs that need comparison. Following the blocking phase, the system conducts pairwise evaluations to measure the similarity between potential records. Once similarity is assessed, a classification or clustering technique may be used to identify if two records represent the same entity. The final phase includes post-processing and verification, where manual reviews may take place to ensure the quality of the results.

A significant challenge arises from issues related to data quality, such as spelling mistakes or incomplete data. These factors impede the accurate organization of documents. Another major challenge is managing variations in names since the same person may be identified in different ways (e. g., "John A. Smith" vs. "J. A. Smith" or "Bill" versus "William"). Furthermore, ER systems need to be scalable to manage large data volumes, as the number of record comparisons rises significantly with the dataset's size. This scalability is especially important for large datasets, since traditional pairwise comparison methods can be resource heavy. Ultimately, entity resolution frequently includes ambiguity and uncertainty in deciding if two records refer to the same entity. For instance, several individuals may share the same name, complicating the determination of whether they are the same person based on insufficient information.

2.2 Traditional Methods of Entity Resolution

Traditional approaches to ER mainly depend on rule-based approaches and machine learning methods to connect records that signify the same real-world entity. While these strategies perform effectively in many scenarios, they face difficulties when handling large or noisy datasets. These traditional methods can be classified into four main categories: rule-based, supervised learning, unsupervised learning and hybrid approaches.

2.2.1 Rule-Based Approaches

Rule-based techniques in ER workflows apply predefined rules to evaluate records and determine if they refer to the same entity (Li, Li, & Gao, 2014). These rules might be based on precise characteristics, such as names, locations, or dates. Rule-based systems can be quite efficient when data variations are anticipated, such as common typographical mistakes or established abbreviations. Nonetheless, rule-based approaches often face challenges with scalability since they require manual effort, making them less suitable for large datasets.

2.2.2 Supervised Learning Approaches

Supervised learning methods focus on training machine learning models using labeled data to identify if two records correspond to the same entity. These models rely on characteristics obtained from the records, including similarity scores or the presence of shared attributes. Commonly employed supervised learning models for entity resolution include decision trees (Salim & Mathew, 2016), support vector machines (SVMs) (Salim & Mathew, 2016) and logistic regression (Dey, 2008). The models learn from pairs of records labeled as matches or non-matches and once trained, they can classify new pairs of records. While supervised learning models can outperform rule-based systems in handling complex data variations, they generally need large, labeled datasets for effective training.

2.2.3 Unsupervised Learning Approaches

Unsupervised learning techniques for entity resolution do not require labeled data. These approaches typically employ clustering methods to categorize records according to shared characteristics or similarity metrics. For example, algorithms like k-means clustering or hierarchical clustering can be used to classify pairs of records into as matches or non-matches (Kumar, Ram, & Hanmanthu, 2014), (Saeedi, David, & Rahm, 2021). However, these techniques still encounter difficulties in managing noisy or inconsistent data and often necessitate considerable domain knowledge to create effective clustering strategies.

2.2.4 Hybrid Approaches

Hybrid methods integrate elements of rule-based, supervised and unsupervised techniques to improve the effectiveness and adaptability of ER systems. Initially, these methods may use blocking to minimize comparisons, followed by the application of supervised or unsupervised techniques to enhance the outcomes (Papadakis, Skoutas, Thanos, & Palpanas, 2020), (Papadakis G. , Skoutas, Thanos, & Palpanas, 2019). Hybrid techniques can surpass single-method strategies by leveraging the advantages of each approach to tackle different aspects of the ER process.

2.3 Modern Methods of Entity Resolution

The increasing complexity, size and diversity of modern datasets are some of the reasons which have spurred the development of more advanced approaches that make use of more complex techniques. These include deep learning, semi-supervised learning, probabilistic models, graph models, transfer learning and few-shot learning approaches.

2.3.1 Deep Learning Approaches

Deep learning techniques have gained traction in entity resolution recently, as they can automatically learn feature representations from raw data. Particularly, neural networks have been widely employed for ER tasks. For example, deep learning techniques can create embeddings of entity records, placing similar entities nearer in the embedding space, regardless of exact string matches or data formats (Li, Talburt, Li, & Liu, 2021). Strategies such as Siamese networks are especially beneficial for entity resolution tasks since they learn to evaluate the similarity between record pairs and classify them as either matches or non-matches (Lv, Qi, Huo, Wang, & Gao, 2018). These models can efficiently handle noisy or unstructured data, making them ideal for contemporary entity resolution issues. Deep learning models offer significant advantages in handling extensive datasets, as they can learn from large volumes of data and excel at generalizing to new or previously unencountered variations. Nonetheless, these models require large, labeled datasets for training and can be resource-heavy, potentially limiting their application in specific scenarios.

2.3.2 Semi-Supervised Learning Approaches

Semi-supervised learning has become a crucial approach in ER as it can utilize both labeled and unlabeled datasets. This method is particularly advantageous in situations where acquiring labeled data is costly or demands a considerable amount of time. Through the integration of a small, annotated dataset and a significantly larger collection of unannotated records, semi-supervised learning models can attain strong generalization to novel data. Typical methods involve active learning, where an initial model is built using a small, labeled dataset and the system iteratively selects the least confident record pairs for labeling to refine its performance over time (Simonini, et al., 2021). Another common approach is self-training, where an initial model iteratively labels and retrains on high-confidence predictions to improve its accuracy (Hu, Chen, & Qu, 2011). Techniques like reinforcement learning are also being investigated for entity resolution tasks. In RL-based methods, the issue is viewed as a sequence of decision-making stages, in which an agent engages with the data environment to identify matching records. The agent receives rewards for accurate matches, allowing it to learn and enhance its performance gradually through experimentation (Guo, Han, Zhang, & Chen, 2022). These emerging and semi-supervised methods lessen the reliance on large, labeled datasets while providing enhanced adaptability and scalability, rendering them useful resources for addressing contemporary entity resolution issues.

2.3.3 Probabilistic and Graph-Based Models

Probabilistic methods, such as Bayesian networks and Markov random fields, have gained popularity in ER due to their ability to manage uncertainty and ambiguity (Aleshin-Guendel & Steorts, 2024), (Singla & Domingos, 2006). These models can assess the likelihood that two records pertain to the same entity by scrutinizing different attributes and their relationships. Probabilistic approaches are particularly beneficial for managing uncertain or incomplete information, as they allow the system to leverage previous knowledge and learn from the evidence at hand. Models based on graphs are often used in modern ER methods as well. These models represent records as nodes in a graph, where edges indicate possible matches. Graph algorithms can be utilized to examine the links between records to determine corresponding entities (Chen, Kalashnikov, & Mehrotra, 2007). Graph-based techniques are especially beneficial when data is highly interrelated, as observed in social networks or relational databases.

2.3.4 Transfer Learning and Few-Shot Learning

Modern approaches like transfer learning and few-shot learning have proven effective in entity resolution tasks, especially when labeled data is limited. Transfer learning allows models to leverage knowledge obtained from one domain or dataset and use it in another, similar domain (Kasai, Qian, Gurajada, Li, & Popa, 2019). This is particularly beneficial when there is a lack of labeled data for ER in a particular context, whereas there is plenty of data available in another domain. For instance, a model created for matching records in one field (e.g., healthcare) can be modified to perform ER in a different field (e.g., finance). In contrast, few-shot learning focuses on teaching models to perform well with a limited number of labeled examples. This method is highly beneficial for ER when labeling extensive datasets is unfeasible or costly. Few-shot learning approaches enable the model to utilize insights from a small number of labeled samples and effectively identify entities in new, unfamiliar situations.

2.4 Privacy in Entity Resolution

Privacy has increasingly become a significant issue in entity resolution. Although ER is essential for improving data quality and supporting decision-making, it also carries privacy risks. A major privacy issue in entity resolution is the potential release of personally identifiable information during the record-linking stage. Additionally, merging information from different sources might lead to breaches of privacy regulations such as GDPR or HIPAA. To address these privacy concerns, researchers have developed various methods to protect data during the ER process.

2.4.1 Data Anonymization and De-identification

Anonymization and de-identification are critical methods to protect sensitive data during entity resolution (Davis & Osoba, 2016). By eliminating or concealing personally identifiable information, the chances of privacy infringements are diminished. These methods usually involve omitting or generalizing identifying characteristics from the data prior to matching or linking. For instance, data can be modified through hashing, generalization, or pseudonymization, making it challenging to connect records with individuals. Nonetheless, these methods must strike a balance, as over-anonymization can remove too much crucial information, damaging the efficiency of the ER processes

2.4.2 Secure Multi-Party Computation

Secure multi-party computation (SMPC) lets multiple parties work together to compute a function using their data, while keeping their individual data private. This method is particularly advantageous in cases where data owners need to work together on entity resolution while maintaining privacy. This approach is especially beneficial in situations where data owners must collaborate on ER while preserving confidentiality. Key mechanisms in SMPC include garbled circuits, which encrypt computations so only the output is revealed, secret sharing, which distributes data across multiple parties such that no single party has complete access and homomorphic encryption, which permits computations on encrypted data without requiring decryption (Zhao, et al., 2019). SMPC offers strong privacy guarantees since raw data is never shared. However, its primary limitations include computational overhead and complexity, especially when applied to large datasets.

2.4.3 Bloom Filters

Bloom filters (BFs) are compact data structures that encode sensitive attributes and allow efficient comparison without exposing raw data. By applying hash functions, data is transformed into fixed-size binary arrays, enabling approximate matches between records. Bloom filters are particularly effective for ER tasks involving large datasets, as they reduce storage requirements and speed up comparisons (Ranbaduge & Schnell, 2020). Additionally, they provide a layer of privacy since the original data is not directly accessible. Despite their advantages, bloom filters have limitations. One of them is that the encoded data could be exposed to inference attacks or reverse engineering, especially if the original data is predictable or the hash functions are weak. This means sensitive information might still be revealed in certain situations, risking privacy.

2.4.4 Differential Privacy

Differential privacy (DP) ensures that the inclusion or exclusion of a single record in a dataset does not significantly affect the outcome of any analysis, thereby protecting individual data. This is achieved by introducing controlled random noise into similarity scores or analysis results, effectively masking the contribution of any single record (Kanyar, 2023). Differential privacy mechanisms also incorporate privacy budgets to regulate the amount of permissible information leakage over multiple queries. While DP provides strong mathematical guarantees of privacy, the added noise can reduce the accuracy of entity matching. Balancing utility and privacy through careful tuning of privacy parameters is critical.

2.4.5 Federated Learning

Federated learning is a decentralized machine learning approach where models are trained collaboratively across multiple devices or organizations without sharing raw data. Each participant trains the model locally on their dataset and only the model updates (e.g., gradients) are aggregated to produce a global model. This method ensures privacy by keeping sensitive data localized and reducing the risk of exposure during the training process. Federated learning is particularly suited for privacy-preserving entity resolution in scenarios involving distributed datasets (Nock, et al., 2018). Challenges in federated learning include ensuring model convergence, handling data heterogeneity across participants and securing communication channels to prevent data leakage during aggregation.

2.5 Green AI and Sustainable AI Practices

The rapid growth of AI has raised concerns about its environmental impact. Green AI and sustainable practices, which aim to reduce the environmental impact of AI without reducing its efficiency or performance, have attracted a lot of attention.

2.5.1 Energy Efficiency in AI Models

Training large AI models, especially deep learning systems, demands significant computational resources, frequently resulting in higher energy consumption. This issue becomes increasingly important as models grow in size and complexity. In response, Green AI aims to improve the energy efficiency of both AI algorithms and hardware (Bolón-Canedo, Morán-Fernández, Cancela, & Alonso-Betanzos, 2024). Techniques such as model pruning, quantization and knowledge distillation can aid in reducing the size and complexity of models, thus lowering the computing load while maintaining performance levels (Neill, 2020). Additionally, enhancing the training procedure by employing more energy-efficient hardware, like specialized processors (e.g., TPUs) or more energy-conserving algorithms, can lower the overall energy expenses linked to training and inference activities (Strubell, Ganesh, & McCallum, 2020).

2.5.2 Carbon Footprint Reduction

The environmental effects of AI extend beyond energy use and encompass the carbon emissions linked to operating and maintaining extensive AI infrastructure. With the expansion of AI, the increasing carbon footprint from training advanced models raises concerns (Liu & Yin, 2024). To tackle this problem, businesses invest in data centers that utilize renewable energy sources like wind, solar, or hydropower. Furthermore, AI developers are encouraged to disclose the carbon footprints of their models, utilizing approaches such as optimizing algorithms to minimize resource consumption and selecting more sustainable energy alternatives.

2.5.3 Sustainable Data Practices

In AI, the quality and quantity of data are vital for training precise models. However, the method of gathering, storing and managing large datasets can also entail considerable environmental costs. Green AI promotes more sustainable data practices, like data minimization, which involves collecting only essential data and reusing existing datasets instead of perpetually gathering new data (Bolón-Canedo, Morán-Fernández, Cancela, & Alonso-Betanzos, 2024). Additionally, data storage and processing can be refined to minimize redundancy and approaches such as federated learning, which enables models to be trained on decentralized data, can lessen the necessity for extensive data transfer and storage, resulting in a reduced environmental impact (Yousefpour, et al., 2023).

2.5.4 AI for Environmental Sustainability

Besides minimizing the ecological footprint of AI technologies, AI significantly contributes to addressing environmental issues and fostering sustainability. It is applied in fields like energy oversight, climate simulation, wildlife preservation and waste disposal. For instance, intelligent grids powered by AI can enhance energy distribution, while machine learning methods can assess environmental data to forecast and reduce the impacts of climate change (Nyangon, 2024). Consequently, AI can enhance sustainable development objectives by encouraging more efficient resource utilization and backing efforts for ecological preservation.

2.5.5 Ethical and Transparent AI Practices

Ethical factors in AI development are strongly connected to sustainable practices. Focusing on transparent decision-making, fairness and equity is crucial to guarantee that the advantages of AI are widely shared, while avoiding negative impacts on society or the environment. Furthermore, organizations ought to establish responsible AI development frameworks, making certain that sustainability objectives are integrated at each phase of AI system design, from inception through deployment and upkeep (Krishnadas, Das, & Das, 2020). This entails creating clear guidelines for the ethical application of AI, as well as guaranteeing that AI systems are designed to be inclusive, fair and resilient to biases that might compromise sustainability initiatives.

2.5.6 Policy and Regulation for Green AI

As the need for AI technologies increases, governments and organizations are starting to create policies and regulations focused on encouraging Green AI. These regulations may encompass guidelines for reducing the environmental effects of AI systems, promoting the use of sustainable technologies and supporting research into energy-efficient AI techniques. Policies may also tackle matters such as the transparency of AI energy use, mandating companies to disclose their carbon footprints and adopt best practices for sustainable AI development (Hacker, 2023). Through these regulatory frameworks, the advancement of Green AI can be backed on a larger scale, ensuring that the swift growth of AI is in harmony with long-term sustainability objectives

2.5.7 Future Directions for Green AI

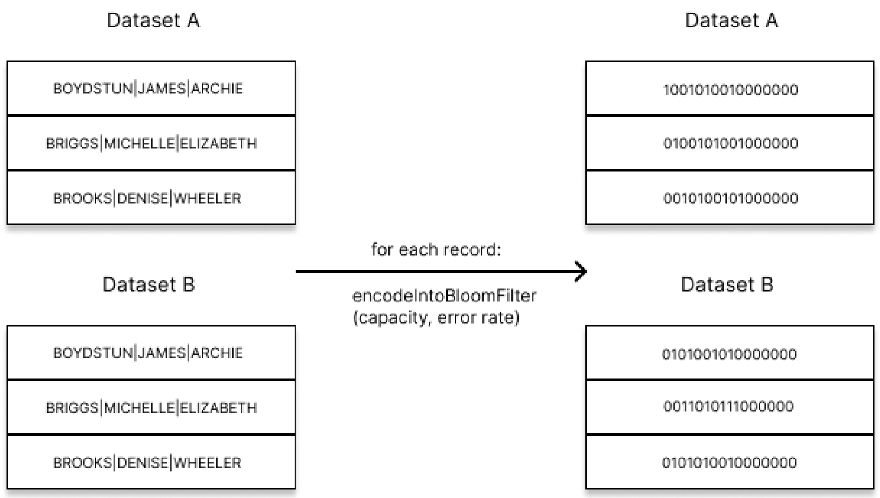
In the future, increased attention to sustainable AI is anticipated as both the research and industry sectors acknowledge the necessity of reducing the environmental effects of AI technologies. Advancements in hardware and software are expected to keep propelling energy-efficient solutions, since AI models will boost sustainability with improved algorithm design and data management methods (Naeeni, 2023). Cooperation between academia, industry and government will be crucial for developing holistic approaches to tackle the environmental issues associated with AI and promote a greener, sustainable future. Embracing Green AI and sustainable AI methods allows us to ensure that the progress of AI technologies fosters both technological innovation and the overarching objectives of environmental protection and sustainable development.

3 Deep Learning-Based Privacy Preserving Entity Resolution

This chapter introduces the workflow for the deep learning-based, privacy preserving entity resolution (DL-PP-ER) process, which is used to identify matching records across two datasets while maintaining data confidentiality. The process begins with records from both datasets being encoded into Bloom filters. To enhance privacy, differential privacy techniques are applied to introduce controlled noise into the filters. Next, all possible encoded record pairs are generated, and their similarity and difference features are computed. These extracted features serve as inputs for deep learning models, which are trained and evaluated to optimize matching accuracy while preserving privacy.

3.1 Data Encoding into Bloom Filters

Figure 1: Encoding Data into Bloom Filters



The first step in the DL-PP-ER process involves encoding the records from two datasets into Bloom filters. The goal is to identify matched records between the two datasets. Before encoding, each record is tokenized into smaller, more manageable components, specifically bigrams. Once the Bloom filters are initialized, these bigrams are added to the filters for further processing.

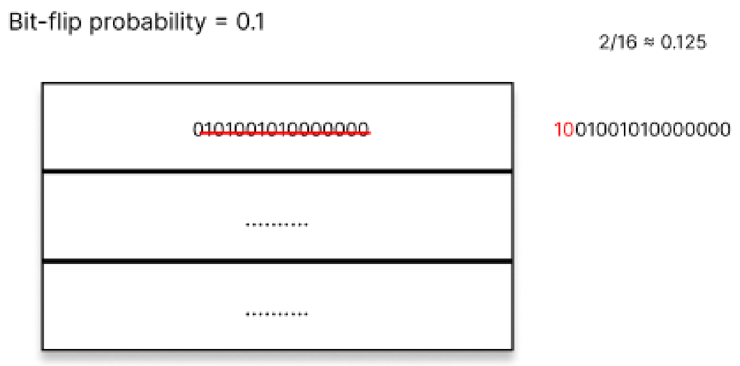
Bloom filters are initialized with two critical parameters: capacity and error rate. Capacity refers to the maximum number of elements the Bloom filter can store, determining how much data can be efficiently encoded. A higher capacity allows the filter to store more elements. The error rate, or false positive rate, represents the probability that an element will be incorrectly identified as part of the set.

These two parameters directly impact the size of the Bloom filter. Specifically, a higher capacity or a lower error rate requires more bits in the Bloom filter to ensure it can handle a larger number of bigrams while maintaining lower error rates. Bloom filters are designed to balance these factors based on application needs, aiming to achieve an optimal trade-off between accuracy and space efficiency.

Figure 1 presents an example showcasing the initial records and the corresponding Bloom filters after transformation. In the example, each row initially represents a record with three columns, and after encoding, a Bloom filter.

3.2 Adding Noise into the Bloom Filters

Figure 2: Introducing Noise into the Bloom filters



After encoding the data into Bloom filters, privacy preservation is further addressed by introducing noise into the filters, using a technique inspired by randomized response, a fundamental method in differential privacy. This noise is introduced through bit flipping, a process in which certain bits within the Bloom filter are randomly altered to obscure the original data.

The probability of bit flipping depends on the privacy budget, a critical parameter in differential privacy. The privacy budget regulates the amount of noise added to the data, balancing privacy with utility. Increased noise makes it more difficult to infer the exact values of the original records, improving privacy. However, excessive noise may compromise the effectiveness of the Bloom filter, as it can introduce too much distortion. When the noise level is too high, the Bloom filter becomes less reliable for matching records, as valid comparisons may be obscured by the random bit flips.

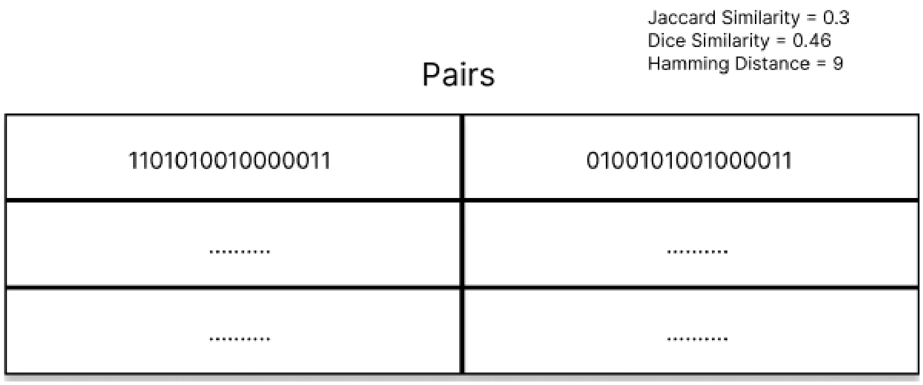
Adjusting the privacy budget requires careful consideration. A smaller privacy budget increases the bit flip probability, leading to a higher introduction of false positives or even false negatives, ultimately reducing the filter's ability to accurately compare data. Therefore, the key challenge is to strike the right balance between enhancing privacy through noise addition and preserving the filter's effectiveness in performing meaningful data comparisons.

In Figure 2, an example of a Bloom filter before and after the introduction of noise is illustrated. The Bloom filter has a size of 16 bits, and the bit flip probability is set to 10%, meaning approximately two of the sixteen bits are randomly altered. The bit flips may vary since the process is based on randomized response. In this specific case, the first two bits change: one flips from 0 to 1, and the other from 1 to 0.

3.3 Generating Pairs and Calculating Features

Next, pairs of records are generated, and features are computed for comparison. This process involves calculating the Cartesian product of the encoded datasets, producing all possible pairs of records. While the Cartesian product is a straightforward approach, it is computationally expensive, especially for large datasets. However, it remains necessary in the absence of blocking techniques that could reduce the number of candidate pairs.

Figure 3: Calculating Similarity and Difference Scores Between Differential Privacy Bloom filters



Jaccard Similarity = 0.3
Dice Similarity = 0.46
Hamming Distance = 9

Pairs

1101010010000011	0100101001000011
....	....
....	....

Once candidate pairs are generated, the next task is to calculate features that quantify the similarity and differences between the records. This process uses three key metrics: Jaccard similarity, Dice similarity, and Hamming distance.

Jaccard similarity is frequently used to evaluate the likeness between two sets. It is defined as the size of the intersection divided by the size of the union of the sets and is expressed as a percentage, showing the proportion of shared elements between the sets relative to the total number of unique elements.

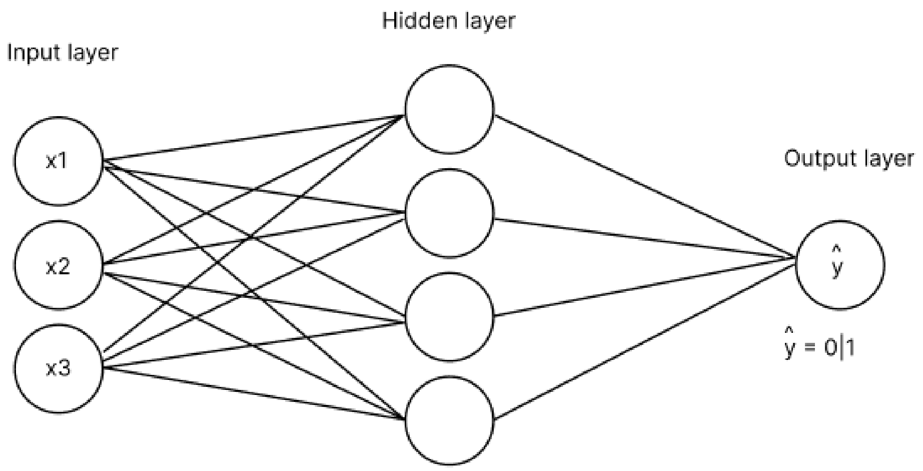
Dice similarity is another metric that measures the overlap between two sets but assigns a higher weight to the intersection relative to the sizes of the sets. This makes Dice similarity particularly useful when the intersection is small compared to the size of the sets. Similar to Jaccard, Dice similarity is presented as a percentage, indicating the extent of overlap between the sets.

Conversely, Hamming distance quantifies the disparity between two strings of the same length. It tallies the positions where the respective characters differ, providing an efficient method to compare binary information or encoded strings. Unlike the first two metrics, Hamming distance is a raw numerical value, representing the absolute number of differing positions between the two strings.

In Figure 3, an example of the calculation of Jaccard, Dice, and Hamming distance between Bloom filters is illustrated. In the example, Jaccard similarity is 0.3, Dice similarity is 0.46, and Hamming distance is 9.

3.4 Training and Evaluation of Neural Models

Figure 4: Training and Evaluation of Neural Models



The deep learning model utilizes these three similarity measures—Jaccard similarity, Dice similarity, and Hamming distance—as features. The training phase involves feeding these computed features into a neural network, enabling the model to learn complex patterns and relationships between similar and dissimilar record pairs.

Model performance is evaluated using standard classification metrics, including precision and recall. Precision measures the proportion of correctly identified matches among all predicted matches, while recall quantifies the proportion of actual matches correctly identified. It is expected that recall achieves high values since Bloom filters typically do not produce false negatives.

An example of a neural network architecture is illustrated in Figure 4. The architecture consists of an input layer, a hidden layer with four neurons, and an output layer, where the output is 0 for non-matched records and 1 for matched records.

4 Methodology

This chapter outlines the methodology used to evaluate the effectiveness of the DL-PP-ER process. It begins with a description of the datasets used in the study, followed by an explanation of the different parameters and configurations employed throughout the process. The chapter also provides an overview of the various approaches used to assess the model's performance. Additionally, the chapter details the tools and metrics used to measure and analyze energy consumption during the various stages of the ER process.

4.1 Datasets Descriptions

Two datasets were used in the study, both containing 10,000 records. The first dataset, dataset_A, had five columns, while the second, dataset_B, contained two columns. To simulate realistic scenarios where discrepancies complicate the matching process, two additional versions of each dataset were created. For dataset_A, the versions dataset_A_1 and dataset_A_5 were introduced, with one fault and five faults added per record, respectively. Similarly, for dataset_B, the versions dataset_B_1 and dataset_B_5 were created, with one and five faults per record, respectively. These varying levels of faults were designed to increase the complexity of the matching process.

4.2 Approaches

The aim of the study was to evaluate both the effectiveness and efficiency of the DL-PP-ER process, as described in Chapter 3, in identifying true matches between original records and their modified counterparts. To achieve this, three different approaches were implemented: exhaustive training, few-shot learning, and cross-dataset evaluation. In the first two approaches, only dataset_A and its modified versions were used, while in the third approach, both datasets, along with their modified versions, were employed.

4.2.1 Exhaustive training

In the exhaustive training approach, a series of experiments were conducted to assess the impact of varying key parameters on the performance of Bloom filters and the overall entity resolution process. Before training the model, the features were split into training, validation, and test sets with a ratio of 60:20:20. The results of these experiments help examine the trade-offs between accuracy, privacy, and energy efficiency, with the goal of optimizing the overall performance of the investigated entity resolution process.

4.2.2 Few-Shot Learning

In the few-shot learning approach, the objective was to assess the entity resolution model's performance when trained on a smaller dataset. This approach aimed to evaluate whether the model could maintain accuracy in identifying true matches while improving efficiency with smaller training sets. To achieve this, we trained the model using sample sizes of 10, 100, and 1,000, ensuring an equal distribution of matched and unmatched labels for each set. These varying sample sizes allowed us to explore the model's ability to perform effectively under different levels of data availability.

4.2.3 Cross Dataset Evaluation

In the cross-dataset evaluation approach, we would like to investigate the possibility of training a model once and reusing it across different datasets to save energy, rather than retraining the model each time. In this approach, we trained the model using dataset_B and its modified versions (dataset_B_1 and dataset_B_5) and tested it on dataset_A and its modified versions (dataset_A_5 and dataset_B_5). The aim was to evaluate how well the model trained on one dataset could identify true matches when applied to a different dataset with potentially distinct characteristics, such as varying numbers of columns.

4.3 Parameters Testing and Configuration

The capacity of the Bloom filter was tested at two values: 100 and 200. A higher capacity allows more elements to be stored, potentially reducing false positives. However, it also requires more memory, which could impact the performance of the entity resolution process.

The error rate, representing the false positive rate of the Bloom filter, was tested at two values: 0.1 and 0.01. A higher error rate increases the likelihood of false positives, while a lower error rate improves accuracy but increases the filter size.

The flip probability, which introduces privacy-preserving noise into the Bloom filter, was varied at three levels: 0, 0.01, and 0.1. Flip probability represents the likelihood that a bit in the Bloom filter will be randomly flipped, adding noise to obscure the original data. A flip probability of 0 means no noise is introduced, while higher probabilities increase privacy but may distort the data, potentially affecting the accuracy of matching.

The neural network architecture followed a sequential design with a single hidden layer. The hidden layer was followed by a batch normalization layer and a ReLU activation function. The final layer did not include a sigmoid activation function, as the loss function used was Binary Cross Entropy with Logits. This loss function operates directly on the raw logits, eliminating the need for a separate sigmoid layer and improving computational efficiency.

To address class imbalance in the dataset, class weights were incorporated into the loss function, giving more importance to the minority class (matching pairs) during training. The model was trained using the Adam optimizer with a learning rate of 0.001, selected for its adaptability and efficient gradient computation.

The number of neurons in the hidden layer was varied at three levels: 4, 8, and 16. Varying the number of neurons affects the model's ability to learn patterns in the data, influencing both accuracy and computational requirements. A smaller number of neurons may lead to underfitting, while a larger number can capture more complex patterns but requires more energy and processing time.

Additionally, the number of epochs was tested at three levels: 5, 10, and 20, with an exception for the few-shot learning approach. The number of epochs determines how many times the model iterates through the entire training dataset. Fewer epochs may result in undertraining, leading to suboptimal performance, while more epochs allow the model to learn more effectively but increase training time and energy consumption. In the few-shot learning approach, training continued until the validation loss stopped decreasing for 20 consecutive epochs, using an early stopping mechanism with a patience of 20.

The training process used a batch size of 8192 for all runs in the standard training process, selected to balance computational efficiency and model performance, with an exception again in the few-shot approach. In the few-shot learning experiments, we did not use batches, as the training samples were much fewer.

4.4 Energy Measurements Tools Used

To assess the energy consumption during the various stages of the entity resolution process, several tools were evaluated and compared. Initially, tools such as Powerstat, Powertop and Scaphandre were explored for measuring power usage.

Powerstat is a tool designed to measure the power consumption of a system, providing an estimate of the system's energy use over time. It is useful for tracking power consumption during runtime, giving an indication of the system’s power efficiency under different workloads. Powertop is another tool designed to monitor and optimize power consumption on Linux-based systems. It provides detailed statistics on power usage by various processes and devices, helping to identify areas where energy efficiency can be improved.

Powertop is widely used for fine-tuning power settings in both desktop and server environments. Scaphandre is a more specialized energy measurement tool that provides real-time power consumption data for various hardware components. It is particularly useful in tracking the energy usage of specific processes and operations during complex workflows like machine learning tasks.

Among the various tools available for energy consumption analysis, PyRAPL emerged as the most suitable for this study. Its simple interface and compatibility with Python-based implementations make it ideal for tracking energy efficiency across different stages of workflow. By enabling fine-grained, per-process energy measurements, PyRAPL provides valuable insights into how specific parameters affect overall energy usage.

The tool measures energy consumption in microjoules for CPU operations and microseconds for RAM usage, providing a comprehensive view of power expenditure. While microjoules quantify the energy required for computational tasks, microseconds indicate the time spent on RAM usage. This dual-level tracking offers a detailed understanding of energy demands, enabling the identification of optimization opportunities and efficiency improvements.

Energy usage was assessed at critical workflow stages to determine how different components and settings influenced consumption. During data encoding, the process of tokenizing and encoding the dataset into Bloom filters, as well as the application of differential privacy noise, was analyzed. Similarly, feature generation was evaluated by measuring the energy required to compute similarity and difference metrics using Jaccard and Dice coefficients and Hamming distance.

Model training was another significant focus, with energy measurements taken during forward passes, backpropagation, and weight updates. These were correlated with parameters such as the number of neurons in the hidden layer and the total epochs. Additionally, power consumption during model evaluation was tracked to account for the energy demands of inference tasks.

5 Experimental Results

This chapter presents the results of exhaustive training experiments conducted to evaluate the impact of varying key parameters on the performance of Bloom filters and the overall ER process. The experiments explored the effects of capacity, error rate, flip probability, the number of neurons in the hidden layer, and the total number of epochs on accuracy, privacy, and energy efficiency.

5.1 Time and Energy Analysis of Feature Generation

Table 1: Relationship Capacity and Error Rate with Bloom Filter Size

Capacity	Error Rate	Bloom Filter Size
100	0.01	959
100	0.1	480
200	0.01	1918
200	0.1	959

As shown in Table 1 above, the size of the Bloom filter is directly influenced by its capacity and error rate. Both an increase in capacity and a decrease in error rate cause the Bloom filter size to grow significantly. For example, at a capacity of 100 and an error rate of 0.1, the Bloom filter size is 480 bits. When the error rate is reduced to 0.01, the size increases to 959 bits. Similarly, at a capacity of 200 and an error rate of 0.01, the filter size grows to 1918 bits. This demonstrates that as the capacity increases or the error rate decreases, the Bloom filter size expands, which leads to a noticeable increase in both time and energy consumption.

The feature generation process consists of two main phases: encoding data into the Bloom filter and calculating features from the encoded data. Both phases are influenced by the size of the Bloom filter, which is determined by its capacity and error rate. Larger Bloom filters require more time and energy to encode, as they involve processing a larger number of bits, thus increasing computational demands. Consequently, an increase in Bloom filter size directly leads to higher encoding time and energy consumption.

Figure 5: Impact of Bloom Filter Parameters on Time and Energy on Encoding Phase

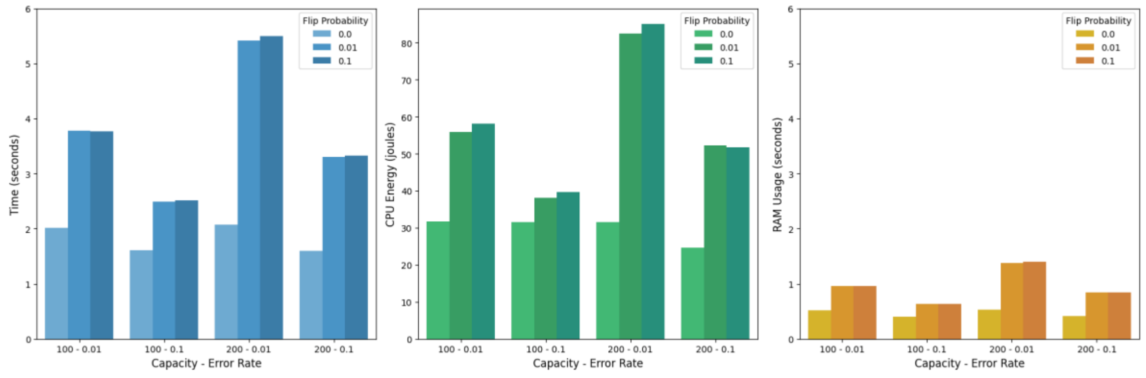


Figure 5 illustrates how variations in Bloom filter parameters affect time and energy consumption during the encoding phase. The x-axis displays different combinations of capacity and error rate, while the y-axis shows the corresponding metrics for time, CPU energy, and RAM usage, presented sequentially from left to right. The varying flip probabilities are represented by different shades of bars.

As shown, larger filters result in increased encoding time, CPU energy, and RAM usage. This means that increasing the capacity or decreasing the error rate leads to longer encoding times and higher energy consumption. Additionally, introducing noise into the filters adds an extra layer of complexity, further increasing both time and energy usage.

Figure 6: Impact of Bloom Filters Parameters on Time and Energy on Calculating Features Phase

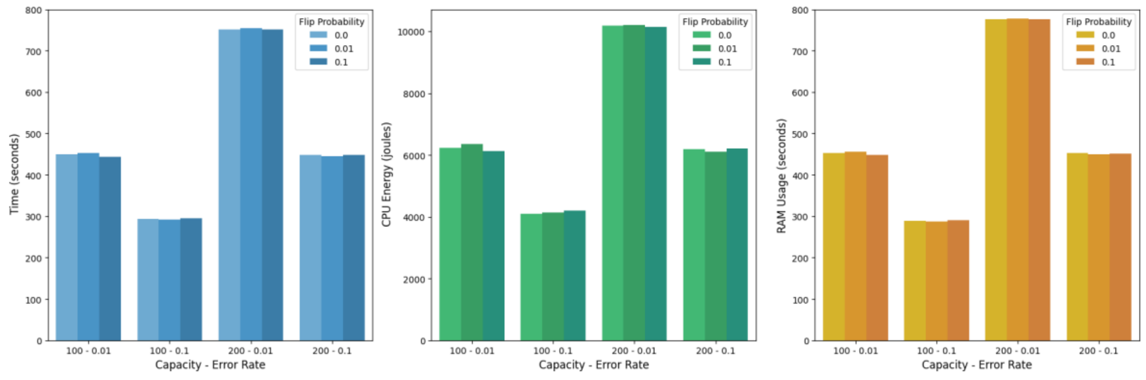


Figure 6 demonstrates how changes in Bloom filter parameters influence time and energy consumption during the feature calculation phase. As with Figure 5, the x-axis represents various combinations of capacity and error rate, while the y-axis shows the corresponding metrics for time, CPU energy, and RAM usage from left to right. The different flip probabilities are indicated by varying shades of bars.

It can be observed that the time and energy required to calculate features is significantly higher than that needed for encoding data. Additionally, as in the encoding phase, the size of the filter has a substantial impact on time and energy consumption. However, in this phase, the calculations are not influenced by the introduced noise.

Figure 7: Impact of Bloom Filters Parameters on Time and Energy on Generating Features Process

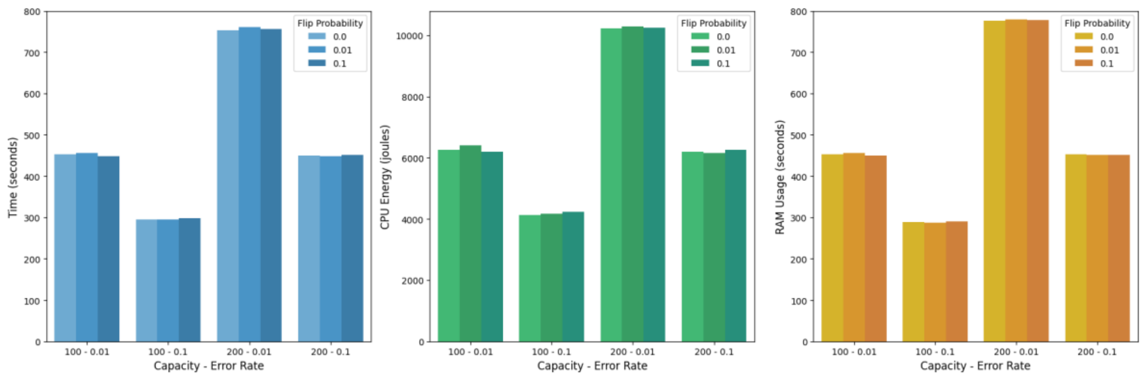


Figure 7 combines the impact of Bloom filter parameters on time and energy consumption across the entire feature generation process, which includes both the encoding and feature calculation phases. Similar to the previous figures, the x-axis shows various combinations of capacity and error rate, while the y-axis presents metrics for time, CPU energy, and RAM usage. The flip probabilities are represented by different shades of bars.

It is evident that while introducing noise into the filters adds complexity to the encoding phase, its effect on overall time and energy consumption is minimal. What has a more significant impact are the values of capacity and error rate, which directly influence the size of the filters.

For instance, at a capacity of 100 and an error rate of 0.01, the total time for the entire process was approximately 450 seconds across all privacy levels. However, when the capacity was increased to 200, the total time rose to around 750 seconds, regardless of the privacy level. In contrast, increasing the error rate to 0.1 reduced the total time to approximately 300 seconds across all privacy levels. CPU energy consumption and RAM usage closely followed these time variations. For example, when the time increased from 300 seconds at a capacity of 100 and an error rate of 0.1 to 450 seconds at a capacity of 200 and an error rate of 0.1, CPU energy consumption rose from around 4,000 joules to 8,000 joules, while RAM usage increased from roughly 300 to 450 seconds.

Therefore, although privacy layers introduce some additional time and energy costs, their impact on the overall feature generation process is relatively minor compared to the effects of the maximum number of elements that can be added to the filter and the probability of false positives.

5.2 Time and Energy Analysis of Training and Evaluation Phase

Table 2: Architecture Details of Neural Network Models

Configuration	Weights (Input to Hidden)	Biases (Hidden)	Batch Norm Parameters	Weights (Hidden to Output)	Bias (Output)	Total Parameters
16	48	16	32	16	1	113
8	24	8	16	8	1	57
4	12	4	8	4	1	29

Table 2 presents the number of weights, biases, batch normalization parameters and the total number of trainable parameters for each configuration of the neural network model.

The architecture of the neural network as mentioned in Chapter 4 followed a sequential design with a single hidden layer. In this model, different numbers of neurons were tested in the hidden layer to assess their impact on model performance. The number of neurons in the hidden layer varied for each configuration (16, 8 and 4 neurons).

The number of weights between the input layer and the hidden layer is calculated by multiplying the number of input features (3) by the number of neurons in the hidden layer. For example, in the configuration with 16 neurons in the hidden layer, there are $3\times16=48$ whereas for 8 neurons, there are $3\times8=24$ weights and for 4 neurons, there are $3\times4=12$ weights.

Each neuron in the hidden layer has an associated bias term. The number of biases in the hidden layer is equal to the number of neurons in that layer. Thus, for 16 neurons, the hidden layer has 16 biases, for 8 neurons, it has 8 biases and for 4 neurons, it has 4 biases. These biases allow the model to adjust the output of the neurons independently of the input data, which helps in learning complex patterns.

Batch normalization introduces two parameters for each neuron (scale and shift). Thus, for 16 neurons, there are $2\times16=32$ batch normalization parameters, for 8 neurons, there are $2\times8=16$ parameters and for 4 neurons, there are $2\times4=8$ parameters.

The number of weights between the hidden layer and the output layer is equal to the number of neurons in the hidden layer. Therefore, for 16 neurons, there are 16 weights, for 8 neurons, there are 8 weights and for 4 neurons, there are 4 weights. The output layer also includes a single bias term, which is the same across all configurations.

The total number of trainable parameters in the network is the sum of all the weights, biases and batch normalization parameters. The configuration with 16 neurons has a total of 113 parameters, the configuration with 8 neurons has 57 parameters and the configuration with 4 neurons has 29 parameters.

Larger models with more neurons have more parameters and typically require more time and energy for training, but they may also have a higher capacity to learn complex patterns. Conversely, smaller models are more computationally efficient but may struggle to capture complex patterns in the data. Additionally, the number of epochs plays a crucial role in determining model performance. Higher numbers of epochs generally lead to better results as the model has more opportunities to learn, but they also significantly increase training time and energy consumption.

Figure 8: Impact of Number of Neurons and Total Epochs on Time and Energy on Training Phase

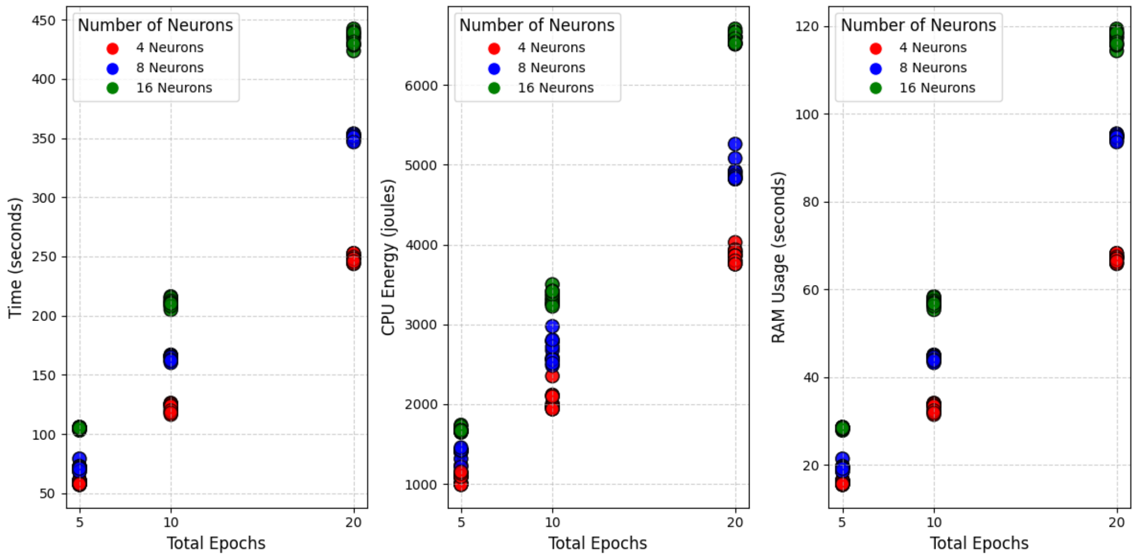
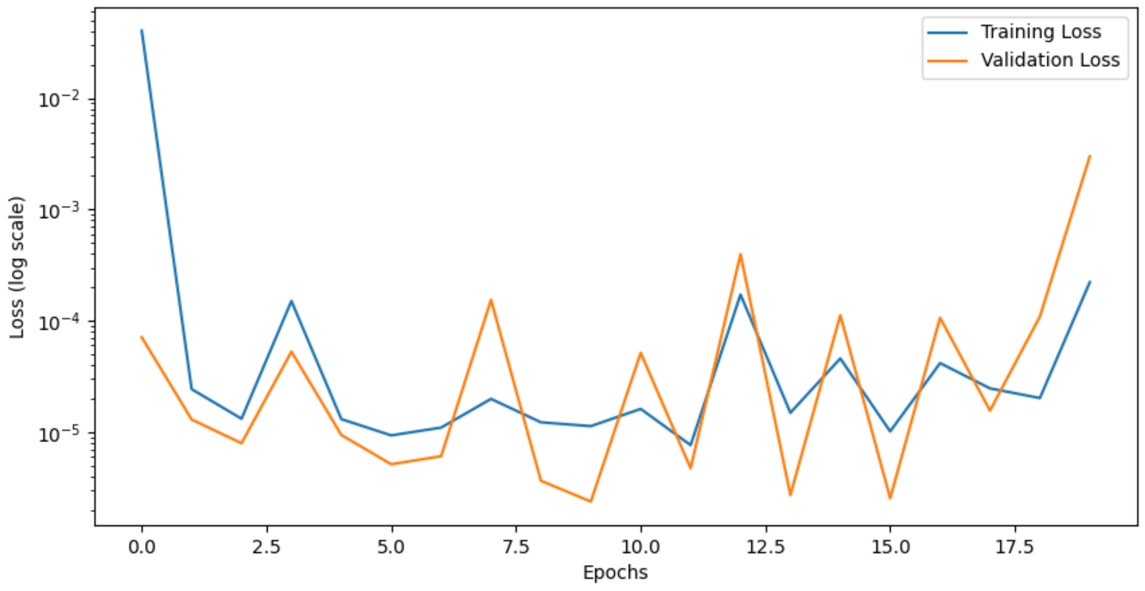


Figure 8 illustrates the impact of the number of neurons and epochs on the time and CPU energy consumption and RAM usage required during the training phase. On the x-axis, the total number of epochs is represented, while on the y-axis, from left to right, are the total time, CPU energy, and RAM usage required for neural network training. Different architectures are represented by different colors.

It is evident that training larger neural networks is more costly in terms of time and energy, which is expected. Larger networks have an increased number of parameters, requiring more computations during both the forward and backward passes of training, leading to higher processing time and energy consumption. For example, for 10 epochs, the neural network with 4 neurons in the hidden layer required approximately 125 seconds to train, with an average CPU energy consumption of around 2200 joules and RAM usage of about 35 seconds. In contrast, the network with 16 neurons required over 200 seconds to train, consuming around 3500 joules of CPU energy and almost 60 seconds of RAM usage.

Similarly, increasing the number of epochs significantly raised the time and energy requirements for neural network training. Each epoch represents a complete pass through the training dataset, so doubling the number of epochs effectively doubles the computational workload. This leads to a proportional increase in the time required to complete training, as well as in the energy consumption of the CPU and the total RAM usage.

Figure 9: Loss vs Val Loss



In Figure 9, right above, the training loss and validation loss are displayed across epochs, with the y-axis represented on a logarithmic scale. During the experiment, in some epochs the validation loss was observed to be lower than the training loss. While this pattern often suggests that the validation dataset is simpler compared to the training dataset, in this case, it can be attributed to the behavior of batch normalization layers. Batch normalization introduces noise during training, which temporarily increases the training loss. This noise is absent during validation, leading to the observed discrepancy between the two losses.

It can be observed that if the training had been stopped early at 5 epochs, the model would have been selected at the epoch with the lowest validation loss, which occurs around epoch 5. Moreover, extending the training to 10 epochs would have resulted in selecting the model at epoch 9, where the validation loss was again at its lowest. Also, if the training was continued to 20 epochs, the best model would have been chosen at epoch 15, where the validation loss reached its minimum for the extended training period.

However, from epoch 15 onwards, the validation loss began to increase, indicating potential overfitting. This suggests that continuing training beyond epoch 15 would result in greater energy consumption without yielding better results, as the model would prioritize fitting the training data at the expense of validation performance.

Figure 10: Impact of Number of Neurons on Time on Evaluation Phase

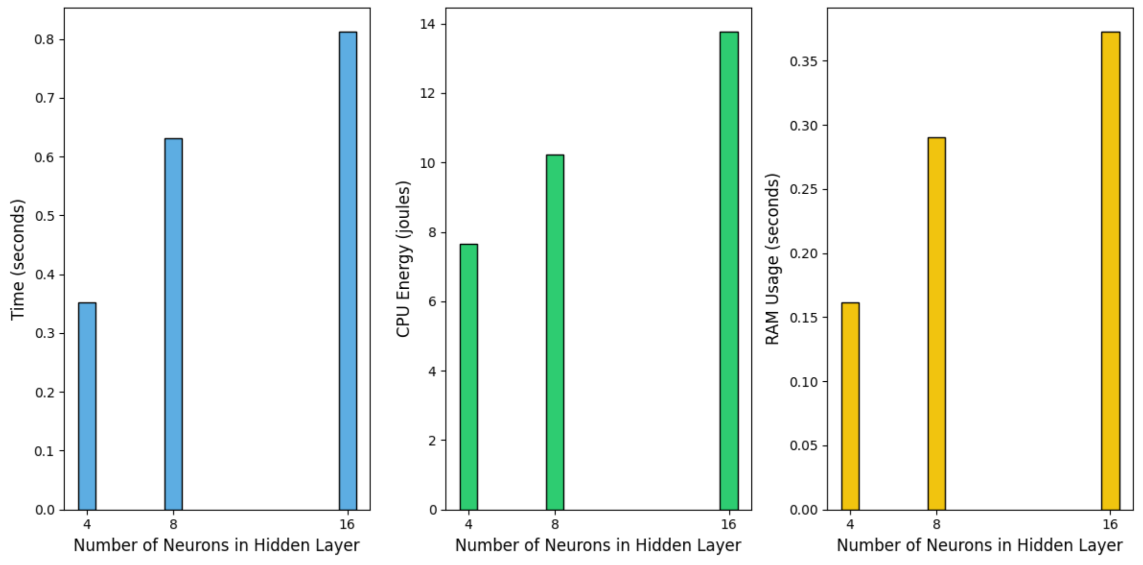


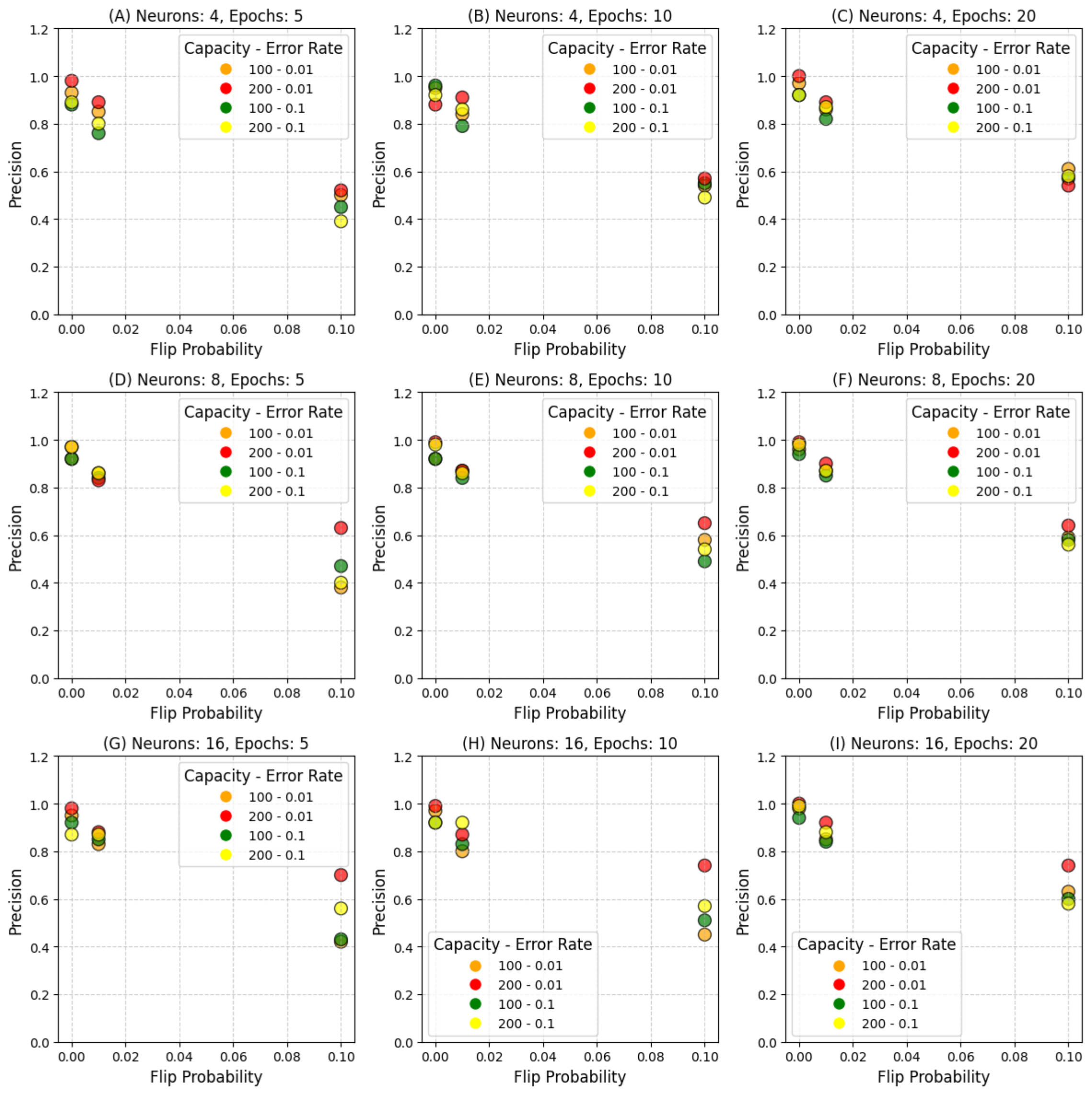
Figure 10 illustrates the impact of the number of neurons in the hidden layer of a neural network on evaluation time, CPU energy consumption, and RAM usage. The first bar chart shows that the evaluation time, measured in seconds, increases as the number of neurons rises, with the highest time observed at 16 neurons. The second chart demonstrates that CPU energy consumption, measured in joules, also grows proportionally with the number of neurons, peaking at 16 neurons. Similarly, the third chart reveals that RAM usage, measured in seconds, follows the same trend, increasing with the number of neurons and reaching its maximum at 16 neurons. These findings indicate that larger neural networks demand greater computational time and energy for evaluation.

5.3 Precision and Recall Analysis for Different System Configurations

In all experiments, data was encoded using Bloom filters, ensuring the absence of false negatives and achieving a recall of 1. However, some pairs were incorrectly predicted as positive when they should have been negative. Figures 11 and 12 illustrate the impact of privacy on precision for different neural network architectures and varying total epochs. Figure 11 focuses on the modified dataset with one fault per record, while Figure 12 examines the modified dataset with five introduced faults.

Each figure contains nine subplots representing different combinations of neurons and total epochs. In each subplot, the x-axis represents flip probability, and the y-axis represents precision. The first row of subplots corresponds to 4 neurons, the second row to 8 neurons, and the third row to 16 neurons. The three columns represent models trained for 5, 10, and 20 epochs. Distinct colors in the diagrams denote different Bloom filter parameter combinations.

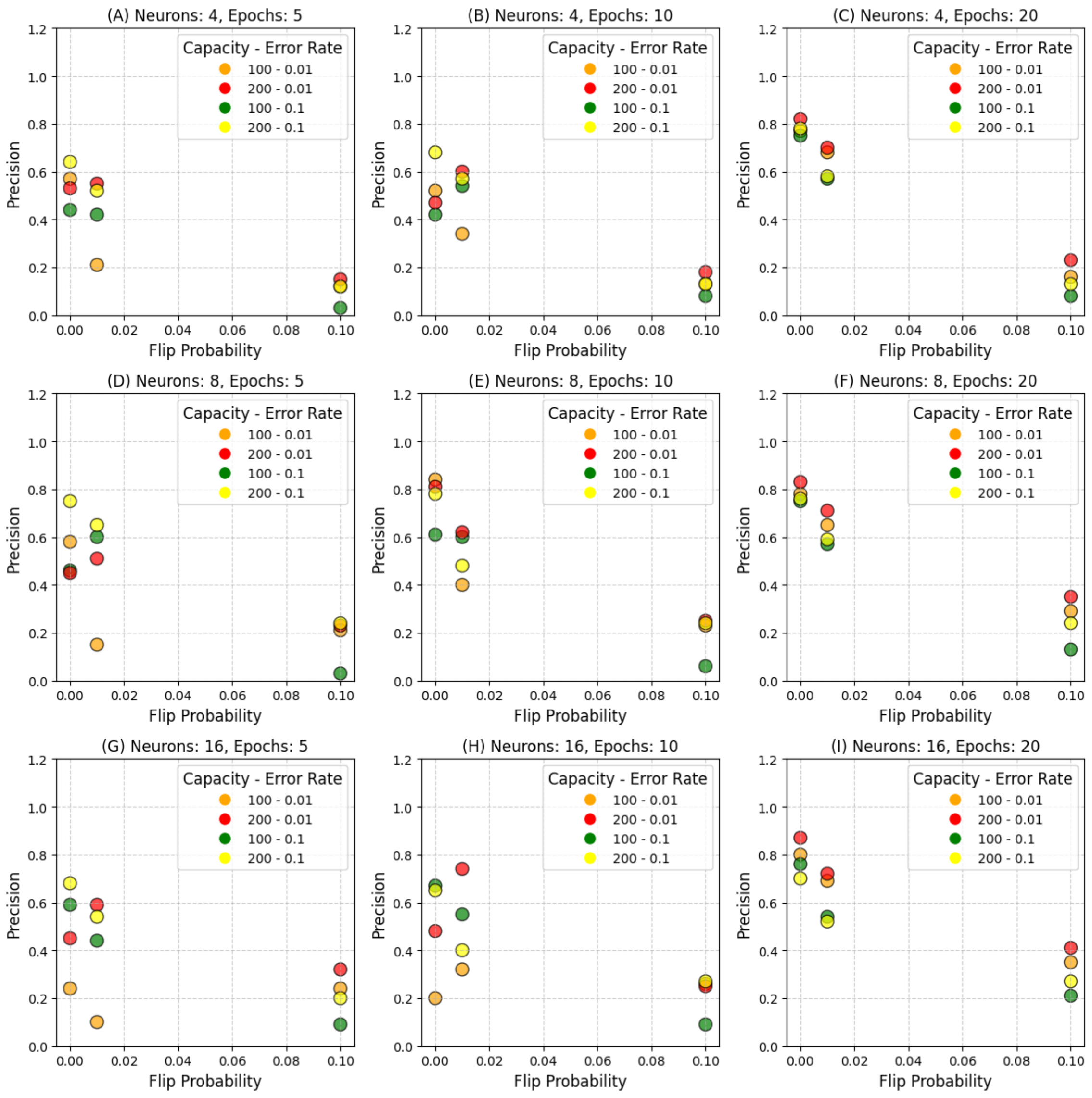
Figure 11: Precision vs Privacy Trade-off with One Fault Introduced in Modified Dataset



In all subplots of Figure 11, it can be observed that higher flip probabilities, which correspond to stronger privacy measures, lead to poorer classification performance. For example, in subplot (I), which corresponds to 16 neurons and 20 total epochs, flipping 10% of the bits in Bloom filters with a capacity of 100 and an error rate of 0.1 caused the precision to drop from nearly 1 to approximately 0.6.

Additionally, in most cases, larger networks with more neurons in the hidden layer achieved better precision. For instance, comparing subplot (G) with (A) and (D), we observe that for Bloom filters with a capacity of 200, an error rate of 0.1, a flip probability of 0.1, and 5 total epochs, the precision was 0.6 with 16 neurons, compared to 0.4 with 4 or 8 neurons.

Figure 12: Precision vs Privacy Trade-Off with Five Faults Introduced in Modified Dataset



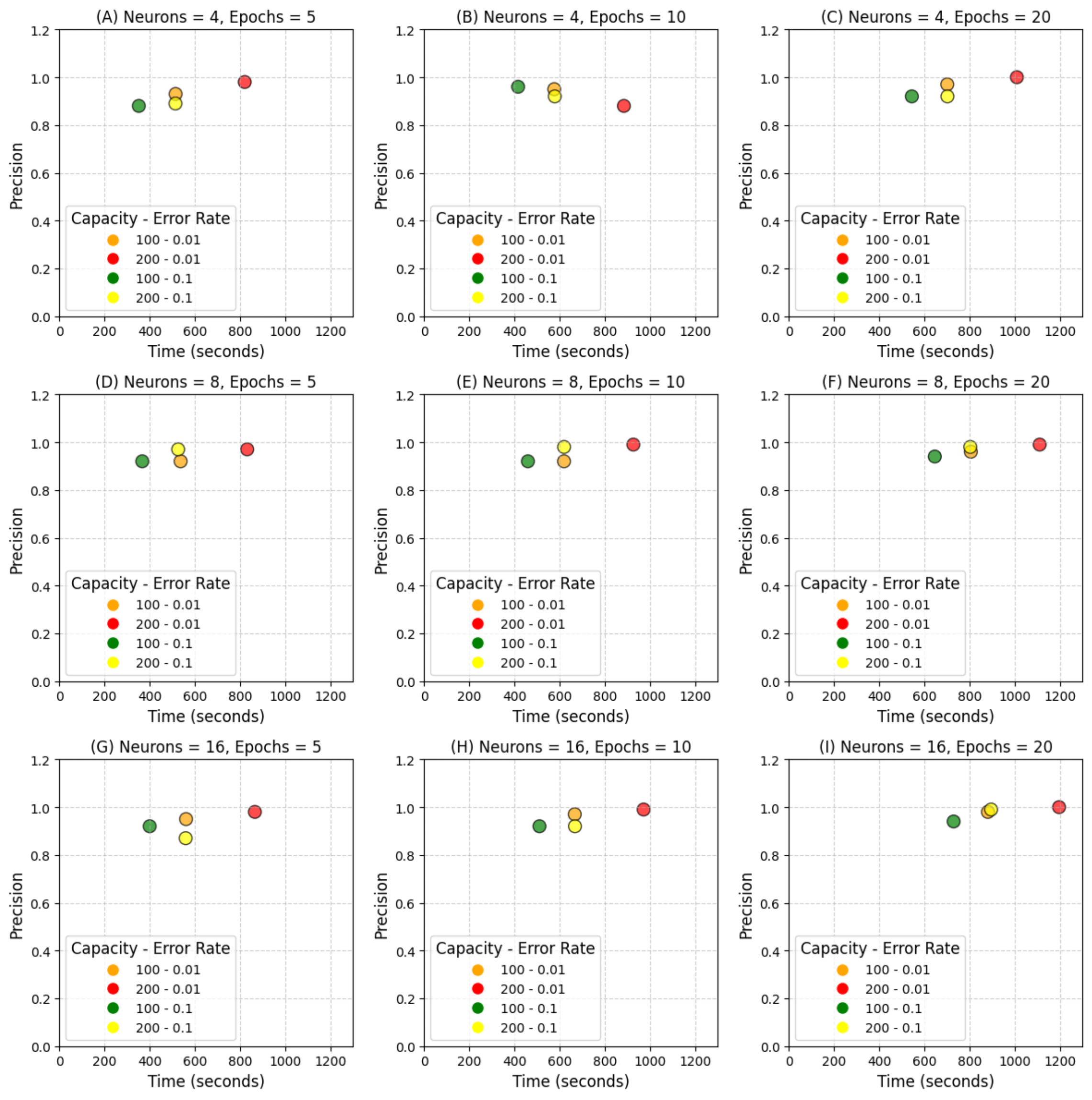
In Figure 12, a noticeable drop in precision is observed as the number of errors in the modified dataset increases. This suggests that recognizing matched pairs becomes more challenging when five faults per record are introduced, compared to just one. For example, when comparing subplot (I) in Figures 11 and 12, for Bloom filters with a capacity of 200 and an error rate of 0.01, the precision for the dataset with one fault was just below 0.8, whereas for the dataset with five faults, it dropped to approximately 0.4.

Furthermore, in Figure 12, it is evident that increasing the number of training epochs generally leads to a slight improvement in precision across most experiments. For instance, comparing subplots (A), (B), and (C), we observe that for most Bloom filter configurations, precision improves as the number of epochs increases.

Lastly, in both Figures 11 and 12, recognizing positive pairs was easier for features generated from pairs of Bloom filters with larger capacities and lower error rates. In subplots corresponding to 5 epochs, and in some cases 10 epochs, this trend might not be as pronounced due to limited training time. However, it becomes more evident in subplots (C), (F), and (I) in both figures, where the improvement is clearer as the model has more opportunity to learn and differentiate between features.

5.4 Time and Energy Analysis for Different System Configurations

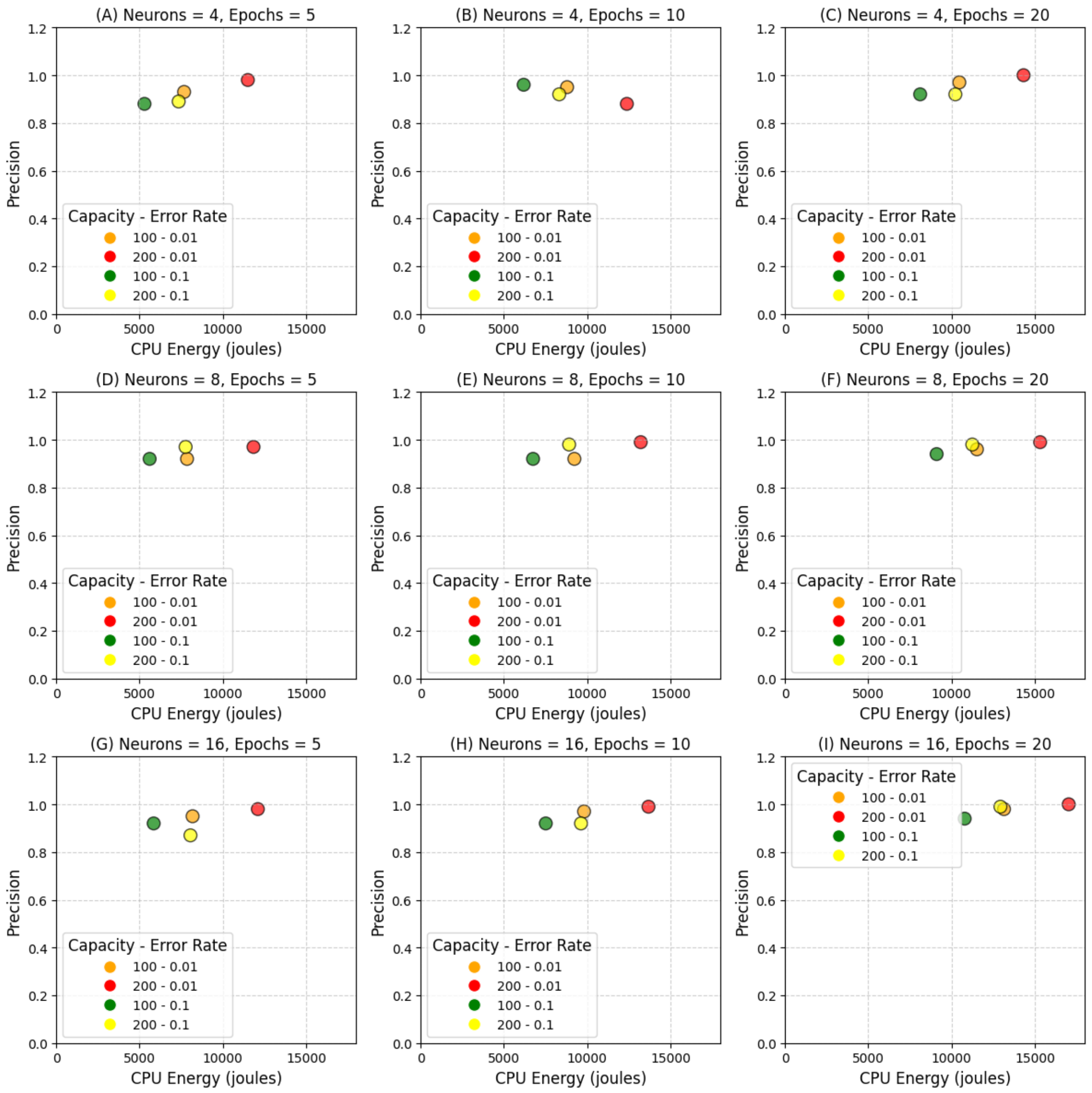
Figure 13: Precision vs Time Energy Trade-Off with One Fault Introduced in Modified Dataset



Figures 13, 14, and 15 illustrate the impact of time and energy on precision for the modified dataset with one fault introduced per record, where no layers of privacy were added. Each figure contains nine subplots, representing different combinations of the number of neurons in the hidden layer and total training epochs.

In all subplots of Figure 13, the x-axis corresponds to the total time spent on the entire process. In Figure 14, the x-axis represents the total CPU energy required, while in Figure 15, the x-axis denotes the RAM usage. Across all figures, the y-axis consistently represents precision.

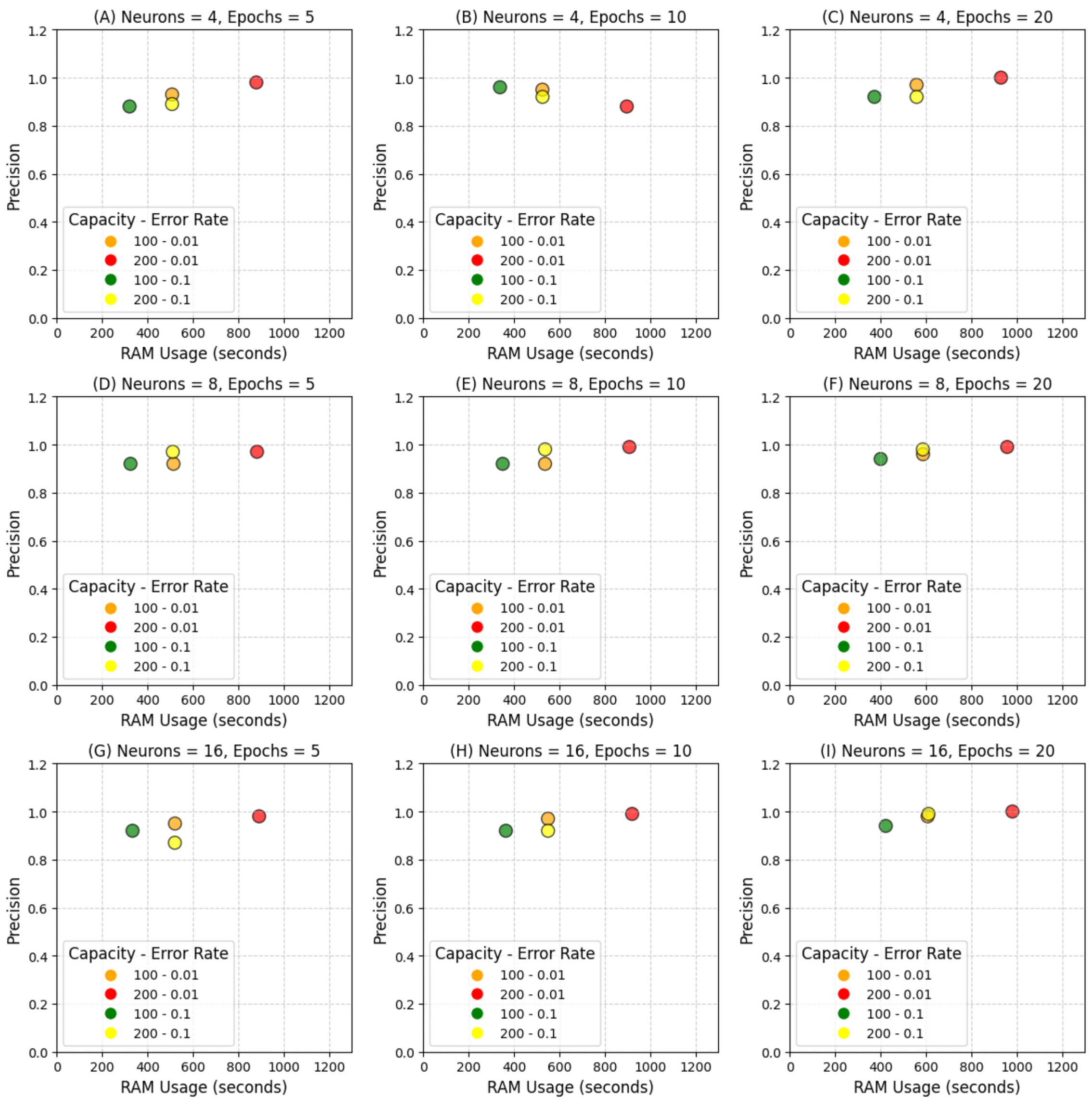
Figure 14: Precision vs CPU Energy Trade-Off with One Fault Introduced in Modified Dataset



The layout of the subplots is organized into rows and columns for clarity. The rows correspond to the number of neurons in the hidden layer: the first row illustrates results for architectures with 4 neurons, the second for architectures with 8 neurons, and the third for architectures with 16 neurons. The columns represent the training epochs, with the first column showing results for models trained for 5 epochs, the second for 10 epochs, and the third for 20 epochs.

In all three figures, it can be observed that capacity and error rate are the parameters that most significantly impact the total time and energy consumption of the ER process. Specifically, as capacity increases and the error rate decreases, the size of the Bloom filter grows, leading to higher time and energy requirements for encoding records and computing similarity and difference metrics.

Figure 15: Precision vs RAM Usage Trade-Off with One Fault Introduced in Modified Dataset



For instance, in subplot (I), which corresponds to a configuration of 16 neurons and 20 epochs, increasing the capacity from 100 to 200 at an error rate of 0.01 causes the total time to rise from approximately 900 seconds to 1200 seconds. Similarly, CPU energy consumption increases from 12,500 joules to 17,500 joules, and RAM usage rises from 600 to 1000 seconds.

The size of the neural network and the total number of epochs also influence the time and energy required during training and evaluation. For example, comparing subplots (C) and (I), increasing the number of neurons in the hidden layer from 4 to 16 causes the total time to rise from approximately 1,000 to 1,200 seconds, CPU energy to increase from 14,000 to 17,500 joules, and RAM usage to grow from 950 to 1,000 MB for a capacity of 200, an error rate of 0.01, and 20 total epochs.

Similarly, comparing subplots (G) and (I), increasing the total number of epochs from 5 to 20 for a capacity of 200, an error rate of 0.01, and 16 neurons results in the total time increasing from 850 to 1,200 seconds, CPU energy rising from 12,000 to 17,500 joules, and RAM usage growing from 850 to 1,000 seconds.

5.5 Accuracy vs Privacy vs Energy Trade-Offs

Higher privacy levels, enforced through increased flip probabilities, introduce noise into Bloom filters, reducing precision and making it harder to differentiate true matches from false positives. However, increasing the Bloom filter's capacity and lowering the error rate can counteract this loss by preserving more relevant information. While privacy enhancements inevitably affect accuracy, careful parameter tuning enables a reasonable trade-off, ensuring that classification performance remains adequate for practical applications. The key lies in adjusting these parameters strategically to maintain a balance between data protection and reliable matching.

Similarly, improving model accuracy often necessitates higher computational costs, leading to greater energy consumption. Larger neural networks with more neurons enhance pattern recognition but also require additional processing power, increasing both training time and resource demands. Extending the number of training epochs further improves model performance but significantly increases energy consumption, making efficiency a crucial consideration. Additionally, the Bloom filter's size plays a crucial role, as larger filters improve accuracy but demand more storage and computational effort. While privacy measures like bit flipping introduce some complexity, their overall energy impact is relatively minor compared to model size and filter adjustments. The ideal configuration ultimately depends on the specific application's priorities, whether privacy, accuracy, or energy efficiency takes precedence.

6 Further Experimental Results

This chapter presents both the results of the few-shot learning experiments and the cross-dataset evaluation. The few-shot learning experiments evaluate the performance of the DL-PP-ER process when trained on smaller datasets, while the cross-dataset evaluation experiments investigate the potential of training a model on one dataset and applying it to others to reduce energy consumption while maintaining performance.

6.1 Few-Shot Learning Results

Table 3: Results of Few-Shot Experiments

Faults per Record	Size of Training Set	Capacity	Error Rate	Flip Probability	Number of Neurons	Precision	Recall
1	10	200	0.01	0.01	16	0.11	1
1	100	200	0.01	0.01	16	0.62	1
1	1000	200	0.01	0.01	16	0.76	1
1	60 M	200	0.01	0.01	16	0.92	1
5	10	200	0.01	0.01	16	0.01	1
5	100	200	0.01	0.01	16	0.32	1
5	1000	200	0.01	0.01	16	0.41	1
5	60 M	200	0.01	0.01	16	0.72	1

The results in Table 3 demonstrate how varying the size of the training set affects the precision and recall of the model. In the exhaustive training approach, the training set consisted of 60 million record pairs, of which 1,500 were matched. In the few-shot learning approach, three different training set sizes were tested: 10, 100, and 1,000 records, each containing an equal number of matched and unmatched pairs. As with the exhaustive training approach, there were no false negatives, so recall remained 1.

The table shows a decrease in precision as the total number of records in the training set decreased, both when the records contained one fault and when they contained five faults. Specifically, as the size of the training set dropped from 60 million to just 10 records, the precision was reduced significantly. This trend was consistent across both fault scenarios. While recall remained at 1, indicating no false negatives, the drop in precision suggests that with fewer training records, the model’s ability to accurately identify true positives diminished.

Interestingly, while a smaller training set required fewer computational resources and energy during the training phase, the training itself was conducted over more epochs. This extended training period continued until the validation loss stopped decreasing, which helped to mitigate some of the effects of the smaller dataset. However, even with the increased number of epochs, the reduced data diversity led to less precise model predictions.

6.2 Cross Dataset Evaluation Results

Table 4: Results of Cross Dataset Evaluation

Training Dataset	**Evaluation Dataset**	**Flip Probability**	**Capacity**	**Error Rate**	**Flip Probability**	**Number of Neurons**	**Precision**	**Recall**
B with 1 error	A with 1 error	0.01	200	0.01	0.01	16	0.98	1
B with 1 error	A with 5 errors	0.01	200	0.01	0.01	16	0.77	1
B with 5 errors	A with 1 error	0.01	200	0.01	0.01	16	0.95	1
B with 5 errors	A with 5 errors	0.01	200	0.01	0.01	16	0.79	1

Table 4 presents the results of a cross-dataset evaluation, examining how well models trained to distinguish between dataset_B and its modified version with one or five faults per record perform when evaluated on other datasets. The goal is to assess the model's generalizability across different datasets.

In the first scenario, where a model trained to distinguish between dataset_B and its modified version with one fault per record was tested on dataset_A with one fault per record, the model demonstrated high precision (0.98) and perfect recall (1), indicating that it generalized well between datasets with similar error characteristics.

In the second scenario, when the model trained to distinguish between dataset_B and its modified version with one fault per record was evaluated on dataset_A with five faults per record, the performance remained strong, with precision and recall at 0.77 and 1, respectively. This suggests that the model still performed well despite the higher number of errors in the evaluation dataset.

When the model was trained to distinguish between dataset_B and its modified version with five faults per record and tested on dataset_A with one fault per record, the model maintained high precision (0.95) and perfect recall (1), indicating it could still handle different error distributions effectively.

Finally, in the case where the model was trained to distinguish between dataset_B and its modified version with five faults per record and evaluated on dataset_A with five faults per record, the recall remained at 1, while precision dropped to 0.79. This indicates that although the model maintained perfect recall, its ability to correctly identify positive instances was affected, suggesting a slight performance degradation. This reduction in precision highlights the challenges the model faced in generalizing when both the training and evaluation datasets contained more complex error patterns.

Overall, while cross-dataset evaluation can save computational resources, it highlights the importance of ensuring that the training dataset reflects the error patterns present in the evaluation dataset. Without this alignment, the model may struggle to maintain consistent performance across different datasets.

7 Conclusion and Future Work

This chapter serves as the conclusion to the study. It begins by providing a summary of the key findings, highlighting the critical insights regarding system parameters and their impact on accuracy, privacy and energy efficiency. The chapter then addresses the limitations of the research, noting the constraints imposed by the experimental design and factors that could influence the generalizability of the results. This section acknowledges areas where further investigation is needed to deepen the understanding of the trade-offs involved in the process. Following this, the broader implications of the study are explored, emphasizing how the findings contribute to the ongoing development of efficient, privacy-preserving machine learning models. The chapter underscores the importance of balancing these three factors—accuracy, privacy and energy efficiency—in real-world applications, particularly in resource-constrained environments. Finally, the chapter concludes with a discussion of potential future directions for research, suggesting several avenues for further exploration that could build upon the insights gained from this study.

7.1 Summary of Findings

This study aimed to investigate the trade-offs between accuracy, energy consumption and privacy in the context of a novel deep learning-based, privacy preserving entity resolution process. The findings reveal a complex interplay between these three factors, with each influencing the overall performance.

A central discovery of the study is that the size of the Bloom filter plays a significant role in both the encoding and feature calculation phases of the DL-PP-ER process. Larger Bloom filters are more effective at representing data, leading to improved classification accuracy. The size of the Bloom filter is influenced by both its capacity and error rate. The larger the capacity, the bigger the filter size and the lower the error rate, the more bits are required to maintain accuracy. Consequently, encoding data into the Bloom filter and calculating features based on that encoded data are the two phases most impacted by Bloom filter size. However, this improvement in accuracy comes with trade-offs: larger filters require more computational resources in terms of both time and energy, as they involve a greater number of bits to process. This demands more time and energy, as the system must process a larger volume of bits during both data encoding and feature calculation.

Furthermore, while the flip probability, which introduces privacy by randomly flipping bits, does add some additional computational cost, its effect on time and energy consumption is relatively minimal compared to the impact of Bloom filter size. The flip probability increases the complexity of the encoding process by adding noise to the data, but this added complexity has only a minor effect on the overall energy and time required. The computational demand induced by flip probability is considerably smaller when compared to the substantial changes in energy consumption or processing time driven by variations in Bloom filter size or the complexity of the deep learning model.

Moreover, the size of the neural network model, particularly the number of neurons in the hidden layers, was found to play an important role in the model's ability to capture complex patterns in the data. Larger models, with more neurons, require more parameters, enhancing the model's learning capacity and improving its performance on complex tasks. However, a larger model demands significantly more computational resources, which in turn leads to higher energy consumption during both training and evaluation. While larger models improve accuracy by allowing the system to capture more detailed patterns, they also require more energy and time to process.

The number of epochs during training was also found to influence both energy consumption and model performance. A higher number of epochs generally improves model accuracy, as it allows the model to learn more from the data. However, more epochs mean more computations, leading to increased energy consumption and longer processing times. This introduces a trade-off between achieving better accuracy through additional training and managing computational costs.

To add up, few-shot learning presents an option for reducing energy consumption during the training phase. By using smaller training sets, fewer resources are required for training the model, reducing computational costs. However, this comes at the cost of accuracy. The fewer the samples in the training set, the more likely the model will struggle to generalize and capture the full complexity of the data. This highlights a trade-off between energy efficiency and model performance, where smaller training sets may save energy, but they may also lead to less accurate models.

Model reuse can be an effective strategy to save on energy and time. A model could be trained once on a dataset and reused across others. This approach can significantly reduce the computational costs associated with training, as the model does not need to be retrained each time. However, if the new datasets differ significantly from the original training data, the model's performance may degrade. The model might not perform consistently across different datasets if the error patterns do not align, potentially leading to lower accuracy in some cases.

Overall, this study demonstrates that the ER process involves a complex balance between accuracy, privacy, and energy efficiency. The optimal choice of model and Bloom filter parameters depends on the specific goals of the user. For instance, prioritizing privacy by increasing the flip probability can reduce model accuracy due to the added noise, while improving accuracy through larger Bloom filters or a more complex neural network model increases energy consumption. Users must weigh these trade-offs based on their objectives, whether that involves maximizing accuracy, preserving privacy, or minimizing energy use. If energy efficiency is a priority, smaller models or optimized Bloom filters may be selected to reduce computational costs, even at the expense of accuracy. Similarly, prioritizing privacy could involve increasing the flip probability, which would typically come at the cost of precision. Thus, the trade-offs between accuracy, privacy, and energy efficiency must be carefully considered to align with the specific requirements of the application. Additionally, adopting few-shot learning or model reuse strategies can provide energy savings, but these come with potential compromises in accuracy and consistency across diverse datasets.

7.2 Challenges in Energy Management

While PyRAPL was selected as the most suitable tool for measuring energy consumption during the entity resolution process as mentioned in Chapter 4, its use was accompanied by several challenges. One major limitation was its reliance on Intel’s Running Average Power Limit interface, which is only supported on Intel processors. This dependency restricted the flexibility of the experimental setup, excluding systems running on other architectures. Additionally, RAPL measures energy consumption for specific components such as the CPU and RAM but does not account for other critical elements like GPUs, which are extensively used during neural network training.

Furthermore, background processes running on the system posed an additional challenge, as they could introduce noise into the energy measurements, making it difficult to isolate the energy consumed solely by the entity resolution workflow. To mitigate this, experiments had to be conducted in controlled environments, which increased the experimental overhead. Moreover, PyRAPL lacks built-in tools for visualization and detailed reporting, requiring additional scripts and tools to process the raw energy data, further complicating the analysis.

Despite these challenges, PyRAPL offered valuable insights into the energy efficiency of the entity resolution process, allowing for the identification of parameter configurations that balance performance and energy use.

7.3 Limitations

While the findings presented in Chapter 5 and 6 demonstrate promising results, this study is subject to several limitations that should be acknowledged. One significant limitation is the constrained range of Bloom filter parameters, such as capacity and error rate, explored during experimentation. These constraints, driven by computational and time limitations, restrict the generalizability of the findings to scenarios outside the tested configurations. Additionally, the neural network architecture employed was relatively simple, consisting of a single hidden layer with varying neuron counts. Although this design facilitated efficient experimentation, it does not represent the complexity of modern deep learning models, which might exhibit different trade-offs between accuracy, privacy and energy efficiency.

Scalability remains a concern, as the experiments focused on moderate-sized datasets and features, leaving uncertainties about the applicability of the methods to high-dimensional or large-scale data. Moreover, the study did not consider the temporal evolution of adversarial techniques or environmental factors that could diminish the effectiveness of privacy mechanisms over time. Finally, the absence of a comprehensive cost-benefit analysis limits the practical applicability of the findings, as decision-makers often need to evaluate the economic implications of adopting specific configurations. These limitations highlight the need for further research to explore these aspects in greater depth and enhance the generalizability and applicability of the proposed methods.

7.4 Broader Implications

The findings of this study extend beyond the immediate scope of differential privacy Bloom filters and neural networks, offering insights into the broader field of privacy-preserving deep learning and energy-efficient systems. As data privacy becomes increasingly critical in a world reliant on large-scale data-driven systems, this research underscores the importance of balancing accuracy, privacy and energy efficiency.

The insights gained also have implications for policymaking and regulatory standards in privacy and sustainability. With the growing push for green computing and stringent privacy regulations like GDPR, methods that quantify and optimize trade-offs are crucial. This study highlights the potential for lightweight privacy-preserving mechanisms, such as Bloom filters, to play a significant role in developing systems that comply with legal and ethical standards while maintaining operational efficiency.

Moreover, this work raises important ethical questions about how to balance the competing priorities of energy efficiency and privacy in contexts where resources are limited or sensitive data is prevalent. In resource-constrained environments, such as edge computing or IoT devices, the proposed methods may enable cost-effective deployment while safeguarding user data. However, it also necessitates careful consideration of the societal consequences of prioritizing one factor over another.

Lastly, the methodology and results offer practical guidance for system designers, encouraging them to incorporate privacy and energy considerations as first-class objectives rather than afterthoughts. As machine learning applications increasingly permeate sensitive domains like healthcare, finance and governance, this research emphasizes the need for adaptive frameworks that evolve alongside technological advancements and adversarial strategies. In doing so, it highlights the transformative potential of interdisciplinary approaches that integrate machine learning, privacy engineering and sustainable computing principles.

7.5 Future Directions

While this study provides valuable insights into the trade-offs between accuracy, privacy and energy efficiency in Bloom filter-based systems, several areas for future research could further enhance the understanding and application of these mechanisms. One promising avenue for future research is improving the robustness of Bloom filter-based systems against adversarial attacks. As privacy-preserving techniques gain adoption, adversaries may attempt to exploit vulnerabilities in Bloom filters through membership inference attacks, adversarial querying, or reverse engineering techniques. Investigating methods to strengthen Bloom filters against such threats—whether through cryptographic enhancements, differential privacy mechanisms, or noise injection—could help ensure secure and reliable applications in privacy-sensitive domains such as healthcare and finance. Additionally, exploring the trade-offs between security, computational overhead, and false positive rates could lead to the development of more resilient Bloom filter-based models.

Energy efficiency is another critical area for improvement. Although this study explored energy consumption during feature generation, training and evaluation, further investigation could examine how energy-efficient hardware or specialized accelerators (such as GPUs, TPUs, or low-power embedded systems) might help mitigate the impact of larger models or more complex privacy-enhancing algorithms. Additionally, exploring energy consumption in more complex or real-time applications, such as edge computing or mobile devices, could provide insights into designing practical privacy-preserving systems for resource-constrained environments.

The few-shot learning and cross-dataset evaluation experiments open further potential for future exploration. Future research could investigate how few-shot learning models can be optimized for larger, more diverse datasets to strike a balance between model generalization and energy consumption. Moreover, future work could explore techniques for improving cross-dataset performance, potentially through domain adaptation or transfer learning.

Finally, the development of standardized evaluation metrics for privacy, accuracy and energy consumption in machine learning models is an important consideration for future research. While this study provides valuable experimental results, the lack of universally accepted benchmarks makes it difficult to compare systems across various research domains. Future work could aim to define clear, reproducible evaluation frameworks that enable cross-study comparisons, fostering more transparent and rigorous assessments of the trade-offs involved in privacy-preserving machine learning systems.

By addressing these research gaps, future studies can pave the way for more efficient, scalable and privacy-respecting machine learning models that can be deployed in real-world applications, contributing to a more secure and sustainable digital ecosystem.

8 References

Aleshin-Guendel, S., & Steorts, R. C. (2024). Convergence Diagnostics for Entity Resolution.

Bolón-Canedo, V., Morán-Fernández, L., Cancela, B., & Alonso-Betanzos, A. (2024). A review of green artificial intelligence: Towards a more sustainable future.

Bolón-Canedo, V., Morán-Fernández, L., Cancela, B., & Alonso-Betanzos, A. (2024). A review of green artificial intelligence: Towards a more sustainable future.

Chen, Z., Kalashnikov, D. V., & Mehrotra, S. (2007). Adaptive graphical approach to entity resolution.

Davis, J. S., & Osoba, O. A. (2016). Privacy Preservation in the Age of Big Data: Insights from Information Theory.

Dey, D. (2008). Entity matching in heterogeneous databases: A logistic regression approach.

Guo, L., Han, Y., Zhang, Q., & Chen, H. (2022). Deep Reinforcement Learning for Entity Alignment.

H Isozaki, H. K. (2002). Efficient Support Vector Classifiers.

Hacker, P. (2023). Sustainable AI Regulation.

Hu, W., Chen, J., & Qu, Y. (2011). A self-training approach for resolving object coreference on the semantic web.

Kanyar, M. N. (2023). Differential Privacy “Working Towards Differential Privacy for Sensitive Text “.

Kasai, J., Qian, K., Gurajada, S., Li, Y., & Popa, L. (2019). Low-resource Deep Entity Resolution with Transfer and Active Learning.

Krishnadas, D. L., Das, R., & Das, A. A. (2020). Sustainable Practices in Design, Application and Use of Artificial Intelligence-Introducing the Heart of AI.

Kumar, B., Ram, B., & Hanmanthu, B. (2014). K-Means Clustering Algorithm based on Entity.

Li, L., Li, J., & Gao, H. (2014). Rule-Based Method for Entity Resolution.

Li, X., Talburt, J. R., Li, T., & Liu, X. (2021). When Entity Resolution Meets Deep Learning, Is Similarity Measure Necessary?

Liu, V., & Yin, Y. (2024). Green AI: exploring carbon footprints, mitigation strategies, and trade offs in large language model training.

Lv, Y., Qi, L., Huo, J., Wang, H., & Gao, Y. (2018). Joint Multi-field Siamese Recurrent Neural Network for Entity Resolution.

Naeeni, S. K. (2023). Sustainability and AI: Prioritizing Environmental Considerations in Tech Advancements.

Neill, J. O. (2020). An Overview of Neural Network Compression.

Nock, R., Hardy, S., Henecka, W., Ivey-Law, H., Patrini, G., Smith, G., & Thorne, B. (2018). Entity Resolution and Federated Learning get a Federated Resolution.

Nyangon, J. (2024). Climate-Proofing Critical Energy Infrastructure: Smart Grids, Artificial Intelligence, and Machine Learning for Power System Resilience against Extreme Weather Events.

Papadakis, G., Skoutas, D., Thanos, E., & Palpanas, T. (2019). A Survey of Blocking and Filtering Techniques for Entity Resolution.

Papadakis, G., Skoutas, D., Thanos, E., & Palpanas, T. (2020). Blocking and Filtering Techniques for Entity Resolution: A Survey.

Ranbaduge, T., & Schnell, R. (2020). Securing Bloom Filters for Privacy-preserving Record Linkage.

Saeedi, A., David, L., & Rahm, E. (2021). Matching Entities from Multiple Sources with Hierarchical Agglomerative Clustering.

Salim, S., & Mathew, L. S. (2016). Decision tree based rules for entity identification.

Simonini, G., Saccani, H., Gagliardelli, L., Zecchini, L., Beneventano, D., & Bergamaschi, S. (2021). The Case for Multi-task Active Learning.

Singla, P., & Domingos, P. (2006). Entity Resolution with Markov Logic.

Strubell, E., Ganesh, A., & McCallum, A. (2020). Energy and Policy Considerations for Modern Deep Learning Research.

Yousefpour, A., Guo, S., Shenoy, A., Ghosh, S., Stock, P., Maeng, K., . . . Mironov, I. (2023). Green Federated Learning.

Zhao, C., Zhao, S., Zhao, M., Chen, Z., Gao, C.-z., Li, H., & Tan, Y.-a. (2019). Secure Multi-Party Computation: Theory, practice and applications.