UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA (CEEI)
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO (PPGCC)

THIAGO PEREIRA DA NÓBREGA

TOWARDS AUDITABLE AND INTELLIGENT PRIVACY-PRESERVING RECORD LINKAGE

Campina Grande
2023

Thiago Pereira da Nóbrega

TOWARDS AUDITABLE AND INTELLIGENT PRIVACY-PRESERVING RECORDLINKAGE

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande, como requisito parcial para a obtenção do grau de Doutor Ciência da Computação.

Orientador: Dr. Carlos Eduardo S. Pires
Dr. Dimas Cassimiro N. Filho

CAMPINA GRANDE
2023

N754t Nóbrega, Thiago Pereira da.
Towards auditable and intelligent privacy-preserving Record linkage / Thiago Pereira da Nóbrega. - Campina Grande, 2023.
219 f. : il. color.

Tese (Doutorado em Ciência da Computação) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2022.
"Orientação: Prof. Dr. Carlos Eduardo Santos Pires, Prof. Dr. Dimas Cassimiro Nascimento Filho."
Referências.

1. Machine Learning. 2. Privacidade. 3. Blockchain. I. Pires, Carlos Eduardo Santos. II. Nascimento Filho, Dimas Cassimiro. III. Título.

CDU 004.056.53:34(043)

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECÁRIA ITAPUANA SOARES DIAS GONÇALVES CRB-15/93

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
POS-GRADUACAO CIENCIAS DA COMPUTACAO
Rua Aprigio Veloso, 882, - Bairro Universitario, Campina Grande/PB, CEP 58429-900

FOLHA DE ASSINATURA PARA TESES E DISSERTAÇÕES

THIAGO PEREIRA DA NÓBREGA

TOWARDS AUDITABLE AND INTELLIGENT PRIVACY-PRESERVING RECORD LINKAGE

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação como pré-requisito para obtenção do título de Doutor em Ciência da Computação.

Aprovada em: 12/12/2022

Prof. Dr. CARLOS EDUARDO SANTOS PIRES, Orientador, UFCG

Prof. Dr. DIMAS CASSIMIRO DO NASCIMENTO FILHO, Orientador, UFAPE

Prof. Dr. CLÁUDIO ELÍZIO CALAZANS CAMPELO, Examinador Interno, UFCG

Prof. Dr. LEANDRO BALBY MARINHO, Examinador Interno, UFCG

Prof. Dr. JAVAM DE CASTRO MACHADO, Examinador Externo, UFC

Prof. Dr. ROBESPIERRE DANTAS DA ROCHA PITA, Examinador Externo, UFBA

Documento assinado eletronicamente por **CARLOS EDUARDO SANTOS PIRES**, **PROFESSOR 3 GRAU**, em 13/12/2022, às 09:33, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Dimas Cassimiro do Nascimento Filho**, **Usuário Externo**, em 13/12/2022, às 10:47, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **Robespierre Dantas da Rocha Pita**, **Usuário Externo**, em 13/12/2022, às 10:58, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **LEANDRO BALBY MARINHO**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 13/12/2022, às 14:03, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **JAVAM DE CASTRO MACHADO**, **Usuário Externo**, em 14/12/2022, às 12:11, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

Documento assinado eletronicamente por **CLAUDIO ELIZIO CALAZANS CAMPELO**, **PROFESSOR(A) DO MAGISTERIO SUPERIOR**, em 14/12/2022, às 22:50, conforme horário oficial de Brasília, com fundamento no art. 8º, caput, da Portaria SEI nº 002, de 25 de outubro de 2018.

A autenticidade deste documento pode ser conferida no site https://sei.ufcg.edu.br/autenticidade, informando o código verificador **2970802** e o código CRC **6F0BEFDE**.

Resumo

A Resolução de Entidade com Garantias de Privacidade (REGP) pretende integrar dados privados/sensíveis de várias fontes de dados mantidas por diferentes partes. A REGP tem por objetivo identificar registros (por exemplo, pessoas ou objetos) que representam a mesma entidade do mundo real em fontes de dados privados mantidas por diferentes custodiantes. Devido a leis e regulamentos recentes (por exemplo, Regulamento Geral de Proteção de Dados), as abordagens PPRL são cada vez mais exigidas em áreas de aplicação do mundo real, como saúde, análise de crédito, avaliação de políticas públicas e segurança nacional.Em cenários práticos, o processo PPRL precisa lidar com problemas de eficiência, eficácia (qualidade de ligação) e privacidade. Por exemplo, o processo de PPRL precisa ser executado sobre grandes fontes de dados (por exemplo, um banco de dados contendo informações pessoais de programas governamentais de distribuição de renda e assistência), com uma classificação precisa das entidades e, ao mesmo tempo, proteger a privacidade de a informação. Nesse contexto, este trabalho propõe melhorias no processo PPRL com o intuito de mitigar alguns dos gargalos do REGP. Particularmente, este trabalho apresenta três grandes contribuições para o processo REGP: i) um protocolo que permite a auditabilidade da computação realizada durante o REGP, ii) uma metodologia não supervisionada que aproveita o conhecimento de conjuntos de dados públicos para treinar classificador baseado em Machine Learning para o REGP, e iii) uma nova representação para os dados PPRL codificados/anonimizados que permitem o uso de novas redes neuro e classificadores de aprendizado profundo no contexto do PPRL. As presentes contribuições aprimoram várias partes do processo PPRL, visando torná-lo mais facilmente utilizado em aplicações do mundo real. Com a contribuição apresentada na tese, esperamos facilitar diversas aplicações do mundo real (por exemplo, estudos médicos, epidemiológicos e populacionais) e reduzir os esforços legais/burocráticos para acessar e processar os dados, tornando a execução dessas aplicações mais simples para empresas e governos.

Palavras-chave: Segurança e Privacidade, Resolução de Entidades, Integração de Dados

Abstract

Privacy-Preserving Record Linkage (PPRL) intends to integrate private/sensitive data from several data sources held by different parties. It aims to identify records (e.g., persons or objects) representing the same real-world entity over private data sources held by different custodians. Due to recent laws and regulations (e.g., General Data Protection Regulation), PPRL approaches are increasingly demanded in real-world application areas, such as health care, credit analysis, public policy evaluation, and national security. In pratical scenarios, the PPRL process needs to deal with efficiency, efficacy (linkage quality), and privacy problems. For instance, the PPRL process needs to be executed over large data sources (e.g., a database containing personal information of governmental income distribution and assistance programs), with an accurate linkage of the entities, and, at the same time, protect the privacy of the information. In this context, this work proposes improvements to the PPRL process intending to mitigate some of the PPRL bottlenecks. Particularly, this work presents three major contributions to the PPRL process: i) a protocol that enables auditability of the computation performed during PPRL, ii) an unsupervised methodology that leverages the knowledge of public datasets to train Machine Learning-based classifier (a Transfer Learning methodology) for the PPRL, and iii) a novel representation for the encoded/anonymized PPRL data that enable the use of novel Neuro Networks and Deep Learning classifiers in the PPRL context. The present contributions improve several parts of the PPRL process, intending to make it more easily used in real-world applications. With the contribution presented in the thesis, we expect to facilitate several real-world applications (e.g., medical, epidemiologic, and populational studies) and reduce the legal/bureaucratic efforts to access and process the data, making these applications' execution more straightforward for companies and governments.

Keywords: Privacy-Preserving Entity Resolution, Data Privacy, Machine Learning, Blockchain, Transfer Learning

Acknowledgments

First, I would like to express my profound gratitude to my advisors, Carlos and Dimas, for their continuous and caring help, advice, and encouragement during my DSc studies at UFCG. Without their support, I would not have come this far.

I am also grateful to my lab colleagues (especially Demetrio, Dimas e Brasileiro) for many valuable discussions which have contributed substantially to this thesis. I would also like to thank all the PPGCC/UFCG professors and staff.

I would also like to thank my parents (Crizeuda and Paulo) for their support. Finally, I want to express my deepest gratitude to my wife (Lêda) and daughter (Maria Beatriz) for their unconditional support and understanding during my DSc.

Contents

List of Symbols

3PAC - *Three-Party Auditable Protocol for PPRL*

ABEL - *Auditable Blockchain-Based PPRL*

AT-UC - *Auto-Tuned Unsupervised Classification step*

BBF - *Balanced Bloom Filter*

BF - *Bloom Filter*

BLIP - *Bloom and Flip*

DBMS - *DataBase Management System*

DM - *Data Matching*

DT - *Decision Tree*

ER - *Entity Resolution*

HBC - *Honest-But-Curious*

LU - *Linkage Unit*

MCC - *Matthews Correlation Coefficient*

ML - *Machine Learning*

PoA - *Proof of Authority*

PoW - *Proof of Work*

PPRL - *Privacy-Preserving Record Linkage*

QID - *Quasi-Identifier*

RL - *Record Linkage*

SBF - *Splitting Bloom Filter*

STTP - *Semi-Trusted Third Party*

TL - *Transfer Learning*

XBF - *Xor Bloom Filter*

List of Figures

List of Tables

Chapter 1

Introduction

In recent times, companies and government significantly increased the amount of the collected data. Much of this data is about personal information, such as shopping transactions, browsing history, telecommunication records, financial information, or electronic health records. This data has been employed in data mining and analytic techniques that can provide relevant information for several areas of knowledge. For instance, personal data can i) be employed to perform crime and fraud detection [176], ii) lead to better patient outcomes or to detect a disease outbreak in the health sector [63], iii) be of vital importance to national security [97] or be a competitive edge to a commercial enterprise[10].

Data mining and analysis often require information from multiple data sources to be integrated in order to enable precise and useful analysis [10]. However, to execute data integration, first, we have to identify and aggregate records that relate to the same entity (e.g., people, restaurants, publications, products, among others) from one or more data sources [28]. This process is known as Record Linkage[1] (RL), Data Matching (DM), or Entity Resolution[2] (ER) [28; 10]. Although the process receives several names in the literature, in this work, we will adopt RL.

A recurring problem that Record Linkage faces is the absence of attributes capable of uniquely identifying entities, which refer to the same entity, in the different data sources.

[1]Record linkage is the term widely used by statisticians, epidemiologists, and social researchers (e.g., economists and anthropologist) to describe the process of merging records from one data source with another that describe the same entity.

[2]In the computer science context, the terms Data Matching and Entity Resolution are also employed to name the RL process.

The absence of a unique identifier, such as an ID, makes the use of simple comparison operations (e.g. SQL joins) impossible, making the linkage to be carried out with sophisticated comparisons involving a set of common attributes to all entities in the different data sources. Such a set of attributes is called quasi-identifiers (QIDs) [30].

Currenttly, Record Linkage not only faces computational and operational challenges intrinsic to the comparison and classification methods, but it also has to address privacy preservation challenges due to recent laws and regulations such as European General Data Protection Regulation (GDPR), Brazilian General Data Protection Law (LGPD) and the US HIPAA Privacy Rule. In this context, Privacy-Preserving Record Linkage (PPRL) emerges, aiming to identify matching entities across private data sources, ensuring that the data's privacy and confidentiality are preserved throughout the linkage process.

The use of PPRL has gained special attention during the COVID-19 pandemic. The demand for applications that assist in public health decision-making increases. As an example, we have an application that estimates the infection rate, vaccination population, and so on. An important problem for these applications is the existence of duplicate entities that can negatively influence the interpretation of data analysis. This problem was noticed by worldwide health authorities - i.e., US[3], UK[4], and BR[5] - when they noticed double-counting covid cases problem. Duplicated COVID cases may occur for different reasons. For instance, we might face this problem when patients leave at state borders and perform tests in two different states, leading the state's health authorities to count the same patient twice. The result of duplicated COVID cases can be problematic and lead health authorities to make wrong decisions, e.g., open or close the local economy at the wrong moment.

Another example of PPRL usage is in the Financial Fraud, Law Enforcement, and Counter-Terrorism applications [176]. These applications need to deal with multiple sensitive databases to identify criminals. As listed in the work of Chen et al. [23] in the aforementioned context at least 13 different sensitive databases need to linking prior to the execution of data mining and machine learning algorithms. In the following, we presented the list of databases listed in the Chen et al. work: i) telemarketing call lists, ii) direct mail mailing lists, iii) airline reservations or travel records, iv) frequent flyer/hotel stay program

[3]https://www.nytimes.com/coronavirus-tests-cdc/

[4]https://www.telegraph.co.uk/thousands-coronavirus-tests-have-double-counted-officials/

[5]https://www.jornaldaparaiba.com.br/duplicidade-em-casos-registrados-de-covid-19/

membership information or activity, v) magazine subscription records, vi) information about purchases made at retailers or over the Internet, vii) telephone calling logs or records, viii) credit or debit card numbers, ix) mortgage or car payment information, x) bank account numbers or balance information, xi) records of birth certificates, xii) marriage licenses and divorce decrees, and xiii) utility bill payment information.

Notice that the number of databases to be linked in this context is significant. Besides the privacy of the database records, the PPRL techniques need to address problems related to the volume of information (a.k.a., very large databases), standardization methods for foreign names and addresses, and the quality of the linkage [30]. For example, suppose the PPRL wrongly classifies an individual, identifying an innocent person as a criminal (false matches). In that case, the consequence could be catastrophic to the innocent person that was wrongly identified as a criminal. Furthermore, these databases are constantly modified, making the same entity be represented by different records [23]. For instance, a person could marry and change her/his name, making it hard to the linkage process due to the modification of the record (information) across the time (a.k.a., Temporal Record Linkage [143]).

As previously presented, PPRL needs to address privacy-related issues. Thus, to ensure the entities' privacy during the process, data source owners (also called PPRL parties or data custodians) need to compare and classify the entities without seeing (in full) the other PPRL parties' data. In this sense, data source owners agree to reveal only selected information about matched records among them or to an external party, such as a researcher. Therefore, PPRL operates over an anonymized (or encrypted) representation based on a fraction of the original data, hampering the linkage process by adding complexity in every step of the process. In summary, we can define the PPRL process as:

Definition 1 (Privacy-preserving record linkage)**.** *Assume that n data sources* $D_1, \cdots, D_n$ *are owned by* $|P|$ *PPRL parties* $P_1, \cdots, P_n$*, respectively. The PPRL parties wish to employ the anonymized QID of the entities* $\hat{e}$ *to identify which of their entities* $\hat{e}_{[1,i]} \in D_1, \hat{e}_{[2,j]} \in D_1, \cdots, \hat{e}_{[m,k]} \in D_m$ *match according to a decision model* $\Xi(\hat{e}_{[1,j]}, \cdots, \hat{e}_{[m,k]})$ *that classifies record pairs into one of following two classes: M of matches or U of nonmatches.*

The basic idea of PPRL is to execute the linkage process in anonymized data (by perturbating the original data with the use of encryption, hash functions, and noise additions),

ensuring that the privacy and confidentiality of the data are preserved during the linkage process. PPRL reveals only a limited amount of information. For instance, a party only knows which of its own records exist in the other party's data source or the number of duplicated entities presented in the datasets used as input to the PPRL process [179]. The steps of the PPRL are illustrated in Figure 1.1.

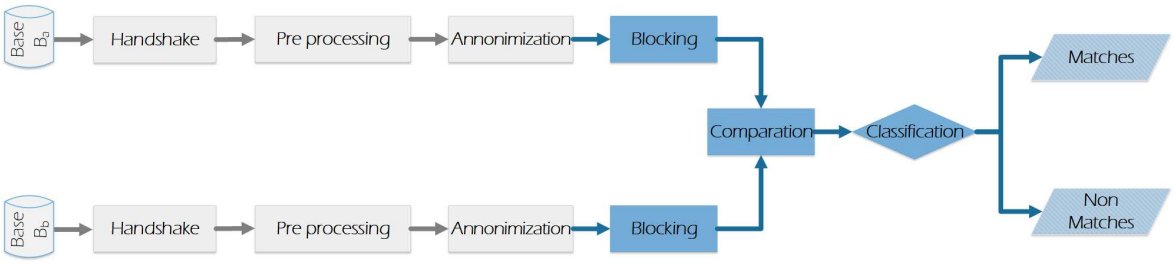


Figure 1.1: Traditional PPRL workflow.

In the *Handshake* step, besides agreeing on data anonymization parameters (e.g., cryptography keys and Bloom filters length), the parties explicitly agree on the attributes (attribute pairing) that will be used to match/link the records during the following steps [128]. An important objective of the *Handshake* is the attribute pairing. Considering that it is not always possible to obtain a common attribute capable of identifying the same entity in multiple databases, for example, the SSN, PPRL uses a set of entity attributes named as quasi-identifiers (QIDs) to perform the necessary comparisons and identify the entities that have a high similarity with entities available at the other participants' datasets.

The *Data pre-processing* step converts the original data into a standard format, previously agreed between the parties. In the *Data anonymization* step, the original is anonymized data, using techniques such as the Bloom Filter (detailed in Section 2.1.4), according to the parameters defined during the *Handshake* step.

The aforementioned steps (*Handshake*, *Data pre-processing* and *Data anonymization*) are conducted by the PPRL parties. Notice that in Figure 1.1 all steps after *Data anonymization* are depicted in a different color. It means that they can be conducted by a third-party or by the PPRL parties; therefore, the usage of anonymized data is mandatory.

The *Blocking* (or indexing) step intends to reduce the number of comparisons needed to be conducted in the following step by pruning of pairs of records that unlikely correspond to matches. For instance, the first letter of a patient's name (blocking key) can be used to group

patient records into a block of candidate pairs. The *Comparison* step applies similarity (or comparison) functions to the anonymized data in order to calculate the similarity between candidate pairs.

The *Classification* step receives the candidate pairs (together with their similarity values) from the Comparison step and classifies each of them into similar (match) or dissimilar (non-match). Finally, in the *Evaluation* step, the efficiency (e.g., execution time, memory consumption, and CPU usage), effectiveness (measured by quality metrics such as Precision, Recall, and F-measure), and privacy of the PPRL process (measured by disclosure risk and information gain) is assessed.

A PPRL solution needs to address three issues (or characteristics): efficiency (or scalability), privacy, and linkage quality. In the following, we outline these PPRL characteristics.

1. **Efficiency**: assume that two data sources to be linked, D_1 and D_2, contain $x = |D_1|$ and $y = |D_2|$ entities. In order to compare every entity in both data sources, we have to perform $x \times y$ comparisons. Therefore, by nature, the naïve linkage process has a quadratic complexity [136], presenting a low efficiency. This problem gets hampered by using big data sources (i.e., data sources with millions or billions of entities) and complex anonymization techniques employed by PPRL. Indexing techniques can be used to overcome this problem [28], as will be presented further in Chapter 2. In summary, PPRL solutions need to be prepared to perform linkage in large data sources;

2. **Privacy**: considering that organizations need to link sensitive information (e.g., medical or credit records) across multiple data sources (D), the PPRL process needs to protect the information's privacy. In order to fulfill privacy-preserving requirements, PPRL solutions employ sophisticated anonymization techniques (e.g., homomorphic encryption and Bloom Filter) to preserve the privacy of the entities at a linkage quality level and an extra computational cost. However, the use of the anonymization techniques do not guarantee information privacy; as demonstrated in Chapter 2, several privacy attacks are able to break the privacy of anonymized data. Therefore, the use of privacy-preserving protocols along with anonymization techniques is required to ensure privacy during the PPRL process;

3. **Linkage Quality**: in general, real-world data sources are 'dirty' [28], which means

they contain errors, typos, variations and values that could be missing. For instance, the name 'Anna Estella' could be entered as 'Ane Stela' by a hospital employee, making it hard to link patient data across different data sources. Therefore, the exact comparison of QID values is not sufficient to achieve accurate linkage results. Thus, to improve the linkage quality, the use of approximate comparison techniques[6], as well as accurate classification techniques, are needed to achieve accurate linkage quality in record linkage applications. These quality problems are exacerbated due to the privacy requirements, i.e., anonymized QIDS. Thus, every PPRL process needs to address the linkage quality issues.

For a PPRL solution to be used in real-world applications, it should address all three characteristics. Furthermore, the solution needs to provide a comprised among privacy, efficiency, and quality according to the needs of the PPRL parties' requirements. There have been many different approaches proposed for PPRL [171; 178; 179]. However, some approaches attempt to address the problem of PPRL fall short in providing a reliable solution, either because they do not provide sufficient privacy capabilities or because they are not able to provide high linkage quality.

1.1 Limitations of Privacy-Preserving Record Linkage

As previously introduced, PPRL needs to address three issues. However, it is worthwhile to mention that Efficiency, Quality, and Privacy are conflicting. In other words, if a PPRL solution prioritizes one of these three characteristics, the other two will suffer.

For instance, if we employ a complex anonymization technique, such as Homomorphic Encryption (HE)[7] [129], we prioritize privacy at an efficiency and quality cost. In other words, by employing this encryption, we add an extra computational cost in every compari-

[6] Approximate comparison techniques return the degree of similarity among two entities, a number between 0 and 1, where 0 means dissimilarity and one total similarity. For instance, if we employ an approximate comparison technique over the 'Anna' and 'Ane' example, it will return a value of .75, indicating that the names are 75% similar, while the exact comparison will indicate that 'Anna' and 'Ane' are not similar.

[7] Homomorphic encryption is an encryption technique that enables arithmetic operations (addition, subtraction, and multiplication operations) over encrypted data. Homomorphic encryption, explained in Chapter 2, can be viewed as an extension of either symmetric-key or public-key cryptography.

son due to the HE operations. Furthermore, we force the linkage process to be carried based only on exact comparisons due to encryption limitations (presented in Chapter 2) [176]. Therefore, the exact comparisons have an impact on the linkage quality because the QID's values need to be the same for a pair of entities to be considered a match; for example, the entities' ana' and 'Ana' are classified as "no match" by exact comparisons techniques.

While PPRL techniques help overcome the privacy-preserving linkage of sensitivity data, they present their own problems. Recent surveys [30; 171; 176; 178; 182] indicate that the main challenges for the extensive use of PPRL are related to the linkage quality, efficiency and privacy issues. In the following, we outline some of the high-level challenges of the PPRL that are marked as open issues by the literature:

- **New adversarial models**: as will be explained in Chapter 2, the parties PPRL need to make assumptions about the behavior of the other parties, and this assumption is named as adversarial models. The currently used adversarial models require that the PPRL parties fully trust other parties [30]. However, this adversarial model is not realistic for real-world applications, mainly because it is hard to find PPRL parties that will not try to learn from the exchange information. Therefore, the need for a more realistic adversarial model is an open issue to the PPRL community;

- **Anonymization techniques**: many of the anonymization techniques used in the PPRL process currently lack evidence that verifies whether these techniques cannot be attacked by an adversary, such as phonetic encoding and generalization techniques [176]. On the other hand, those techniques based on secure multiparty computation and encryption, while probably secure, are currently less scalable to link large data sources. Thus, in order to improve the linkage, novel anonymization techniques are required that are more secure than current approaches while still efficient and accurate, allowing the approximate comparisons of the QIDs values [30];

- **PPRL classification**: most PPRL solutions employ a simple classifier. In order to classify the entity pairs, the PPRL parties define a threshold and compare it against the value that represents the similarity calculated for an entity pair. However, the threshold value definition is a complex task that requires expert operators to "guess" the appropriate value. For instance, if the threshold value is too high (e.g., 0.9 or 1),

PPRL will miss true match entities. On the other hand, if this value is too low, PPRL will likely classify false positive matches. Therefore, novel classification techniques are required in order to help the PPRL operators to classify the entities correctly.

Unless progress is made along with these issues mentioned above, it will not be easy to employ PPRL in real-world data. Next, we present the aims of this thesis.

1.2 Aim of Research

Based on the challenges summarized in Section 1.1, this thesis intends to address the PPRL process's bottlenecks that represent limitations to its extensive use of PPRL. Given the current demands for improvement to the PPRL process, this work's main goal covers improving privacy and the linkage quality of PPRL. The privacy improvements will be concentrated on the anonymization and comparison steps using a novel anonymization technique and auditable data comparison protocols, respectively. The linkage quality improvements are focused on automatic (Machine Learning-based) classifiers to PPRL. Each contribution will be employed to tackle a different bottleneck, detailed in the specific goals section.

1.2.1 Specific goals

Considering the proposed main goal and the fact that the privacy and quality of linkage issues are the most limiting PPRL characteristics to widespread use of real-world applications, this work has the following specific goals:

1. Improve the privacy-preserving capabilities of the Bloom Filter anonymization technique;

2. Propose a novel adversary model that reduces the need of thrust by PPRL parties;

3. Propose a machine learning-based classifier to mitigate the threshold selection during the PPRL Classification step;

4. Propose a novel encoded/anonymized record pair representation that enables the use of novel ML-based classifiers (e.g., deep learning-based classifiers) to improve the linkage of the PPRL process;

1.3 Research Relevance

Data privacy or information privacy has recently gained relevance for individuals, governmental institutions, and corporations. The relevance of data privacy is reflected by the number of laws and regulations presented by different countries worldwide [30]. Due to this regulation, organizations cannot share their data without addressing the privacy of the individuals [176; 178]. In this context, the PPRL process aims to improve the input quality to data applications, such as data mining and analytics.

Identifying duplicated entities across private data sources has an important outcome for any data application. For instance, in the decision-making context, the low quality of the data negatively influences the analyses' interpretation based on these data and, consequently, compromises the decisions. For example, a production chain planning process involving the purchase and stock of raw materials, production, and storage of products. It will most likely be hampered if decisions are made based on reports that do not include duplicated materials stored in a different warehouse.

The previous example was made considering non-private data. Now imagine the consequence of duplicated records in health application. For instance, the existence of duplicated patient records may lead to the wrong conclusion in an investigation if a specific medication is efficient against a disease. In summary, PPRL is an important step to analyze, mine, and process private data sources.

PPRL can be employed in different scenarios besides medical and health applications. As an example, we have anti-terrorist, organized crime, and national security applications. For instance, consider an investigation against the money laundry and corruption investigation. Such investigation needs to use and manage various national databases, from many different sources, including law enforcement agencies, financial institutions, travel history, phone records, and so on [28; 30]. It is obvious that this database is highly sensitive and therefore need to be protected [177]. Thus, the PPRL may facilitate linking the information without all data being given to a criminal investigation unit. In other words, only linked information of suspicious individuals is available to the investigation, reducing privacy and confidentiality breaches [30].

Another important field for PPRL is the Census applications compiled by a national official statistics office, such as the US Census Bureau, the Australian Bureau of Statistics (ABS), the Office for National Statistics (ONS) in the UK, or Instituto Brasileiro de Geografia e Estatística (IBGE) in Brazil. Census data are used for different purposes, and as the most relevant purpose, we can list population estimates, projections for health care and educational planning, and the destination of federal investments. A central task for all statistics offices is to identify duplicated records in Census data to avoid populational double counting and, therefore, avoid incorrect federal investments. However, due to the regulations, privacy concerns arise to the Census databases. These concerns require PPRL techniques that can process a voluminous dataset while preserving the confidentiality of the collected Census data [30].

The need for PPRL solutions is reflected by the wide demand to link real-world private databases. Several countries with different privacy frameworks and legislation are linking some of their sensitive databases. In Brazil, sensitive databases are linking to investigate the consequences of cash welfare and housing programs for Brazil's most poor population concerning their health outcomes [140]. In Germany, information about newborns are linking sensitive to measure the quality of their medical systems. Switzerland, Canada, and Australia standardized their anonymization techniques and created a federal institution to coordinate the linkage of private and sensitive information [30].

Recently laws and regulations (such as the Brazilian LGPD[8], European Union GDPR[9], and USA HIPAA[10]) intend to enhance individuals' control and rights over their sensitive information, enforcing the institutions (governments or private companies) to protect and preserve the privacy of individuals and entities. Moreover, these laws make the right to privacy a basic human right [20]. Therefore, privacy violations are beyond data protection laws and can be characterized as a violation of human rights. Consequently, the penalty for a violation of privacy is severe. In this context, the PPRL task is essential to perform data integration in light of existing privacy laws.

A field that can leverage the PPRL techniques is the Federated Data Linkage application.

[8]https://www.planalto.gov.br/ccivil$_0$3/$_a$to2015 − 2018/2018/lei/l13709.htm

[9]https://gdpr-info.eu/

[10]https://www.hhs.gov/hipaa/index.html

Federated Data Linkage is a data integration task that intends to link data of multiple institutions (e.g., a hospital) that integrates a large network/federation (e.g., all federal hospitals of a country) [74]. In the Federated Data Linkage context, laws (e.g., GDPR and LGPD) and regulations (e.g., HIPAA) require that the privacy of the individuals are preserved during the task execution [15]. For example, during the 2019 pandemic, the German university hospitals [141] made available for researchers a database with information regarding the diagnostic and therapeutic approaches for COVID-19. This database was built over the information available only in isolated silos (comprising all 36 German university hospitals) and inaccessible to external researchers by using a Federated Data Linkage solution (the CODEX[11] project). Therefore, PPRL techniques could link records (e.g., patient records), preserving the privacy of the individual in the Federated Data Linkage applications.

The present work relevance is related to the proposal of methods that attempt to extend the use of PPRL techniques by eliminating bottlenecks. Specifically, this work intends to enhance the linkage quality and propose novel privacy assumptions. The contributions presented in this work can be incorporated into the existing PPRL process and/or can be employed in future researches.

1.4 Research Contributions

In order to illustrate our contributions to the PPRL process, we plotted Figure 1.2. It depicts the PPRL steps, further detailed in Chapter 1, highlighting the steps directly impacted by our contributions. Notice that the figure illustrates a general workflow for two dataset owners. We would like to state that our contribution can be applied in different scenarios, including multi-party PPRL, as explained in the following chapters. For a better understanding, we illustrated our contributions with two dataset owners.

Notice that we propose a contribution to the Anonymization step. This step is critical to the entire PPRL process, impacting the privacy, quality, and efficiency of the PPRL. The majority of the PPRL processes consider the Bloom Filter (BF) anonymization technique, further detailed in Section 2.1.4. BF is able to produce an accurate similarity distance between two entities'. However, recent studies [180; 32; 143; 181; 31;

[11] https://www.netzwerk-universitaetsmedizin.de/projekte/codex

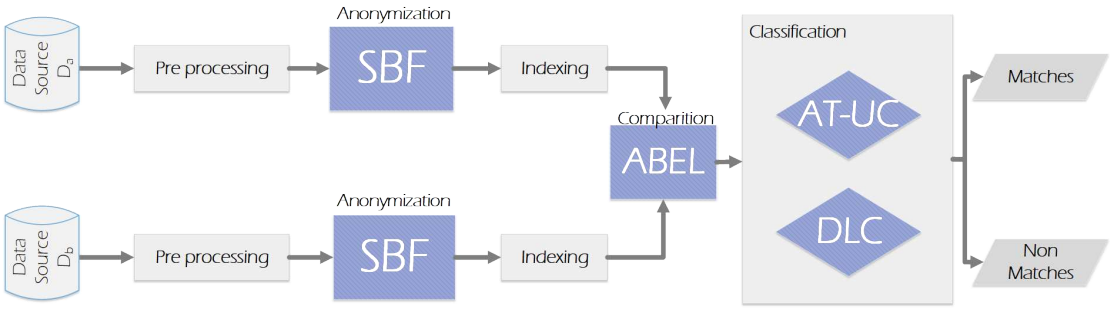


Figure 1.2: Our contributions within the PPRL process.

182] demonstrate that if an attacker has access to a complete database anonymized with this technique, he/she can re-identify the entities, breaking the privacy of the information.

Thus, in this context, we propose the Splitting Bloom Filter (SBF), detailed in Chapter 4. SBF aims to enable an iterative comparison of the entities' similarity by breaking the entities' anonymized representation in splits regarding the BF privacy enhance technique. In other words, SBF modifies the anonymization step's output to enable the auditability in the PPRL comparison step, our second contribution.

In the Comparison step lays our second contribution. A major deficiency in the PPRL context is that the PPRL party needs to consider an unrealistic adversary model. The majority of the PPRL solutions assume an honest-but-curious (HBC) adversary model. This adversary model assumes that all PPRL parties will follow a pre-agreed protocol and will not try to re-identify the anonymized information exchanged during the PPRL. Therefore, having such trust in the PPRL context is unrealistic.

To address the issue mentioned above, we propose the Auditable Blockchain-Based PPRL (ABEL) to provide auditability during the comparison step, eliminating the need to trust the other PPRL parties fully. Moreover, ABEL enables the auditability of the entity's similarity computation using Blockchain technology, with on-chain and off-chain computations. It is worthwhile to mention that the Blockchain stores all processed data on-chain to provide a transparent and temper evident computation. However, this Blockchain characteristic, in a PPRL context, poses a threat to entities' privacy. The usage of off-chain computation by the parties is a fundamental aspect to preserve the privacy of entities during the PPRL execution. A detailed explanation of the ABEL is presented in Chapter 4.

Our third contribution is placed in the Classification step of the PPRL. Due to privacy

limitations, the classification step i) can not be performed or assisted by humans (oracle), and ii) there is no available label data, making it hard to train Machine Learning (ML) classifiers. The majority of the PPRL processes utilize a simple threshold (guessed by a specialist) to define whether an entity pair is a match, or not. It is worthwhile to remark that PPRL is used in law enforcement and medical applications, and an erroneous classification of the PPRL could have a serious outcome to a person. For instance, an innocent man could be flagged as a criminal, or a physician could prescribe the wrong treatment to a patient.

In this context, we propose the Auto-Tuned Unsupervised Classification approach (AT-UC) to provide PPRL with better classifiers; eliminating the need for a specialist to guess a threshold and improve the linkage quality. AT-UC utilizes a Transfer Learning technique to employ non-private datasets for training and modifying a classifier to be executed in a private dataset to tackle the absence of labeled data. Moreover, AT-UC also has to define a proper feature space, select a related dataset, and modify the classifier. AT-UC is presented in detail in Chapter 5.

In the PPRL context, most of the automatic classifiers employ statistical learning techniques (e.g., Support Vector Machine and Logistic Classifiers) [33; 41; 30]. Moreover, these classifiers often employ the similarity measures of the records as input (features). Furthermore, standard similarity measures often do not manage well the heterogeneity of the underlying input data. This requires experts to design and configure such measures manually [105]. Therefore, due to the limitation of the similarity measures employed in PPRL [30; 105], the classifier task of delineating a suitable separation region (e.g., hyperplane or line) between matching and non-matching records gets more challenging.

Our fourth contribution seeks to mitigate the problem of the similarity measures influence over the classifiers employed in PPRL. Our contribution, presented in Chapter 6, employs Deep Learning Classifiers (DLC) to identify patterns that indicate whether an anonymized record pair is a match or not. We also propose a novel representation of the encoded record pair based on a dynamical system representation of the data (Recurrence Plot, detailed in Section 2.8). It is worth noting that the DLC could improve the linkage quality, mitigating the problems of miss classification presented in PPRL.

In summary, this thesis intends to improve the PPRL process in terms of privacy and linkage quality. Moreover, the contributions introduced in this thesis can be employed to

solve problems beyond the PPRL scope. For example, the contribution can be employed to: i) create a Federated Data Linkage solution to integrate multiple sensitive databases (e.g., patient records), providing a tool for epidemiological studies in a country, and ii) adequate data integration tasks to privacy laws (e.g., Brazilian LGPD). The box below summarizes our contributions.

PPRL Open Problems Addressed by our Research

1. Anonymization Technique improvement

Splitting Bloom Filter (Chapter 4)

Addressed Problem:	• The existing Bloom Filter techniques fail to preserve the privacy of the entities when an attacker has access to the anonymized dataset.
Contributions:	✓A technique that enables the approximated comparison using small parts of the original Bloom Filter; ✓Reduce the risk of entities reidentification by an attacker during the Comparison step of PPRL; ✓An improvement to the Bloom Filter anonymization techniques compatible with most used BF hardening techniques.

2. Novel adversary model

Auditable Blockchain-Based PPRL (Chapter 4)

Addressed Problem:	• Most of the existing solutions require that the PPRL parties follow a pre-agreed protocol and, therefore, will trust entirely that all the computations are computed correctly (considering an HBC adversary model).
Contributions:	✓An approach that reduces the level of trust needed to execute the PPRL process; ✓ A high-level protocol to audit the computation performed during the PPRL process; ✓A proof-of-concept implementation of the protocol using Blockchain technology.

3. ML-based classifier

Auto-Tuned Unsupervised Classification (Chapter 5)

Addressed Problem:	• The majority of PPRL solutions consider a simple threshold to classify the entity pairs. Moreover, the threshold value is defined manually (guessed) by a specialist. Furthermore, the entire linkage depends on this threshold value definition, and an improper threshold value could compromise the quality of the linkage process.
Contributions:	✓An unified feature space to PPRL applications; ✓A methodology to select a related training dataset in a privacy-preserving context; ✓A novel semi-automatic unsupervised Classification technique to the PPRL process; ✓Novel implementation of classical similarity metrics to the Bloom Filter.

4. Deep Learning-based classifier

Deep Learning Classifiers (Chapter 6)

Addressed Problem:	• The use of similarity measures as features to PPRL classifiers could result in low linkage quality [105].
Contributions:	✓A novel representation for encoded records pairs; ✓A classifier able to identify matching records using data patterns instead of predefined similarity measurements; ✓Novel Deep Learning architectures for PPRL.

We would like to state that a comprehensive experimental evaluation of our contributions was executed, considering the linkage quality and privacy using multiple real-world and synthetic datasets. We also compare our contributions against several state-of-the-art techniques considering different PPRL scenarios.

1.5 Research Methodology

In order to execute the research, we employ the following research methodology. Based on the work of Kothari [94], we conducted the study in nine steps illustrated in Figure 1.3.

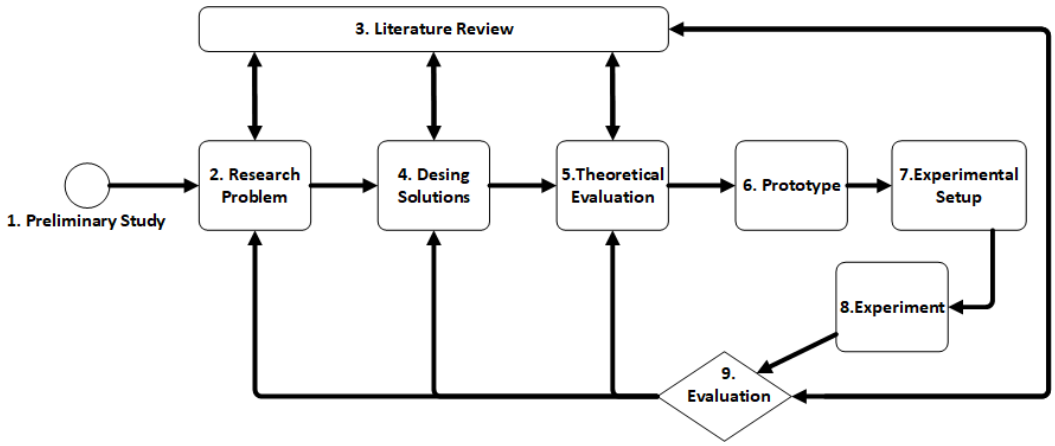


Figure 1.3: Research methodology.

For each contribution, we executed the same methodology. First, we start with a preliminary study. In this step, a basic understanding of the research is gathered by broadly studying the literature, which helped recognize different research problems in the PPRL context. In

the second step, we define the research problem under the previously investigated context. With the research problem defined, we conduct a literature review to understand it better. The knowledge gathered in this step is employed to i) clarify the research problem; ii) propose the solutions; iii) implement prototypes; and iv) present an experimental design.

Once we define the research problem, we propose new approaches to address it in the Design Solution step. When this solution is mature, we start the Theoretical Evaluation step; in this step, we analyze the proposed solution regarding privacy, linkage quality, and efficiency. Notice that the Research problem, Solution Design and Theoretical Evaluation steps are linked to the Literature Review step in order to allow iteration and adjustment in these steps.

In the sixth step, Prototype, the proposed solution is implemented, providing a proof-of-concept solution. In the next step of the process, we propose the setup of experiments to evaluate the proof-of-concept solution. In this setup, we select proper data sources, evaluation metrics, parameters, and state-of-the-art techniques for comparisons. Finally, the experimental results are evaluated concerning efficiency, linkage quality, and privacy to validate the theoretical analysis.

1.6 Document Outline

This thesis is structured as follows. The following chapter provides the background about overall PPRL, Machine Learning, and other concepts related to our research. In Chapter 3, we present a review of the literature on existing PPRL approaches. Chapters 4, 5, and 6 present the Auditable PPRL approach, and the PPRL Unsupervised Classification step, and the Deep Learning Classification step, respectively. Finally, Chapter 7 introduces our final arguments and conclusions.

Chapter 2

Background

This chapter introduces the core concepts underlying the topics discussed in this thesis. We introduce the concepts of data anonymization, Privacy-Preserving Record Linkage (PPRL), PPRL Privacy Attacks, Differential Privacy, Blockchain technology, Recurrence in dynamical system, Transfer Learning, and Deep Learning.

2.1 Data Anonymization

This section provides a brief description of the techniques for encoding and comparing sensitive information based on different dimensions ranging from privacy and technical to practical aspects.

2.1.1 How Anonymization Protects Data Privacy

Individuals have privacy rights (ensured by laws and regulations) regarding who can know or share information their sensitive information. Therefore, in order to protect the sensitive data of individuals, data privacy techniques are employed.

Data privacy

Data privacy refers to the ethical, legal, and technical issues of gathering, manipulating, and sharing data in which identifiable information about individuals is included in the dataset. [61]

In the data privacy domain, a range of techniques intends to remove or mask sensitive information from the dataset. These techniques are named data anonymization.

Data Anonymization is a type of information sanitization - that is, the removal of sensitive information - for privacy protection. It is a procedure to modify a dataset such that the individuals it reflects are anonymous. Typically, this means removing personally identifiable information from datasets so that the identities of individuals in the dataset are anonymous.

Data anonymization

Data anonymization is the process of protecting private or sensitive information by suppressing, masking, or encrypting identifiers that connect an individual to data. In other words, data anonymization is a process of modifying a dataset such that the data can not be used to identify an individual. For example, it is possible to remove the names, social security numbers, and addresses through a data anonymization process that retains the data but keeps the source anonymous. [11]

In summary, data anonymization alters the sensitive information of the individuals in such a way that it can no longer be related back to a given individual. Section 2.4 presents several attacks designed to re-identify the anonymization technique. Some of these attacks are focused on the anonymization technique (e.g., cryptoanalysis [125] and frequency-based attacks [177]). However, other attacks [182] employ non-sensitive data to re-identify the data. For example, using the US census data, a group of researchers could uniquely re-identify 87.1% of the individuals stored in the dataset using only the birth date, gender, and zip code [160].

To preserve the privacy of individuals against attacks over non-sensitive information data de-identification techniques are employed. Data de-identification is a form of dynamic data masking that refers to breaking the link between data and the individual with whom the data is initially associated [49; 193]. Essentially, data de-identification removes or transforms records or personal identifiers to make them statistically de-identified. In other words, data de-identification makes it hard for an attacker to identify an individual from a dataset. On the other hand, data anonymization intends to protect the sensitive data of a record. Data de-identification can be separated into two groups of solutions: Generalizing (e.g., k-anonymization and l-diversity) and Randomizing (e.g., differential privacy and randomized

response).

Data de-identification and anonymization are at different ends of a whole spectrum of data privacy techniques. In this thesis, we work with data anonymization in the context of PPRL. Therefore in the following sections, we provide details of the most used data anonymization techniques in the PPRL process.

2.1.2 Hashing and Encryption Techniques

Hashing is a well-known technique that maps values of arbitrary sizes, such as textual data, to fixed-sized values such as integer numbers that are in a specific range. In other words, a hash function converts its input values into hash values or hash codes. In the PPRL context, one-way hash functions [37] have been used to i) convert a string value into a hash-code (i.e., 'ana' into 'c3ea4f07ff0'); and ii) perform similarity computation over the hashed value.

Algorithms such as Message Digest (e.g., MD5) and Secure Hash Algorithms (e.g., SHA-1 and SHA-256) are the most widely known and used hash algorithms in the PPRL context [176]. However, it is important to remark that, by employing these algorithms as the anonymization method, the PPRL process can only perform exact matching. For example, if we hash the words 'ana' and 'Ana', the result is two distinct hash values, 5BA571 and FA561C, respectively. Furthermore, hashing-based anonymization techniques are susceptible to privacy attacks [30; 32; 181], e.g., Frequency and Dictionary attacks. This topic is better detailed in Section 2.4.

Unlike the hashing-based encoding techniques, encryption techniques allow sensitive data to be encrypted by one party and sent to a linkage unit to be compared in an encrypted form (ciphertext) to generate a result that is also encrypted. This encryption technique is named as Homomorphic Encryption.

Homomorphic Encryption [138] is a family of cryptographic functions that uses two keys: one to encrypt the data and the other to decrypt it. Theses functions enable algebraic operations such as private sum ($\oplus$) and private product ($\otimes$) in encrypted numerical data. In this work, Partially Homomorphic Cryptography (PHC) will be considered. This function allows the private sum of encrypted data and the product of encrypted data by plain constants. Let E_k be a PHC function that encrypts the data, then E_k has the following properties:

$$E_k(x \oplus y) = E_k(x) \oplus E_k(y)$$
$$E_k(x) \otimes c = E_k(x \times c) \tag{2.1}$$

, where c is an unencrypted constant.

PHC functions allow the similarity between non-textual data to be calculated using specialized functions while maintaining the privacy of the data and the comparison result [138; 129]. For example, to compare two dates (A = 01/01/1994 and B = 01/01/1944), we can encode A and B using PHC and compute the difference of these dates in a secure manner, as shown in Figure 2.1.

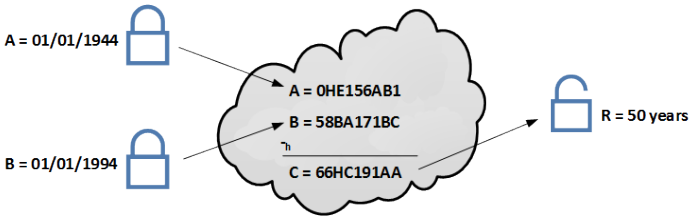


Figure 2.1: To compare the dates, first A an B are encrypted using a common key. The encrypted values are subtracted, generating a third value (C). The result of these operations (50 years) is only known by using a decrypt key.

In summary, Hashing and Encryption Techniques enable exact comparison over textual data at the quality and computational cost due to the encryption and hashing tasks. In turn, Homomorphic Encryption is able to perform arithmetic operations over encrypt numerical data, and therefore perform approximated similarity computation on such data. It is worthwhile to mention that both techniques are susceptible to privacy attacks when employed in a large dataset [177] (presented in Chapter 3). Therefore, its usage in the PPRL context requires further investigation.

2.1.3 Phonetic encoding

A phonetic encoding tenchnique groups values (words) together that have a similar pronunciation. It was originally designed to tolerate typographical variations during a string comparison [176; 28; 30]. The Phonetic encoding algorithm converts the words into a code according to predefined linguistic rules. For instance, using a phonetic encoding algorithm,

both "Ana" and "Anna" return the same code "A163" while "Ane" yields "A150". It happens because the phonetic representation of the words is the same. In the PPRL context, phonetic encoding has been used to provide additional privacy to the comparison step [176; 143].

In the indexing step of PPRL, phonetic encoding could reduce the number of record pairs that need to be compared by grouping entities with similar phonetic representations. The phonetic encoding usage in the indexing step could increase the scalability while also supporting approximate matching of those values with the same phonetic encoding. However, phonetic encoding can reduce linkage quality because false matches are generated when records with attribute values that are different but generate the same phonetic encoding are classified as matches [177] — for instance, the words 'Ana' and 'Anna'.

Regarding the usage of phonetic encoding technique in the PPRL context, several works [171; 178; 190; 179; 157; 176; 30] point out that significant privacy leakage may occur with the phonetic encoding approach due to the frequency distribution of encrypted phonetic encodings.

2.1.4 Bloom Filters

A BF consists of a vector of l-bits (filter length), with all bits set to '*0*', initially. The BF can be formalized as $[b_0, \cdots, b_l]$, where b_m represents the bit of position m. To insert a set of elements ($S = \{s_1, \cdots, s_n\}$) in a BF, k independent *hash functions*[1], $H(x) = \{h_1(x), \cdots, h_k(x)\}$, are employed to map the elements $s_i \in S$ to the l-bits vector. Furthermore, the output of $H(x)$ indicates the bits (b) that need to be set to '*1*' in the BF.

The quality of the anonymization depends on the BF parametrization, number of hash functions (k) and filter length (l) [170]. For a given number of elements (n) to be inserted into the BF, the probability of a specific bit still '*0*' is $p = e^{-\frac{k \times n}{l}}$. We can choose a k to minimize the probability of two different elements being mapped to the same bit position (f) by setting $p = 0.5$ [17]. In other words, the probability of a bit in the BF still 0 (or flipped to 1) should be 0.5 to reduce the false-positive rate. For PPRL, this is relevant because the bit patterns and their frequencies in a set of BF can be exploited by frequency [116] and cryptonalysis [181]

[1] A hash function is an algorithm that takes messages and maps them to a value of a certain length, called a hash value or hash.

attacks. Such attacks exploit the fact that BFs that are almost empty can provide information about rare elements. As a result, the re-identification of the entities is facilitated [172].

Figure 2.2 illustrates the insertion of the names **ANA** and **ANE** into the BFs a and b, respectively. First, each name is transformed into bigrams; then, each bigram is mapped by a hash function into a BF position. Finally, the positions are changed to '*1*' in the BF.

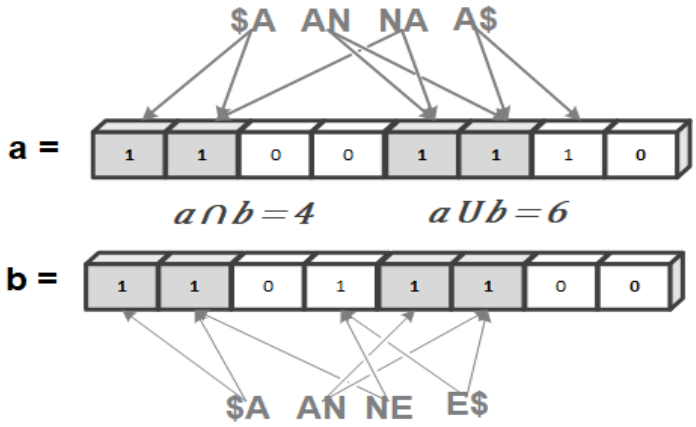


Figure 2.2: Inserting the names ANA and ANE into 8-bits Bloom filters (l=8 and k=2).

The BF technique also enables the similarity calculation of two filters (a and b) through token distance functions, such as $Jaccard = \frac{|a \cap b|}{|a \cup b|}$, where $|a \cap b|$ is the number of positions with the value *1* that coincide in both filters; and $|a \cup b|$ represents the number of positions with the value *1* in the union of the filters a and b. Regarding the example illustrated in Figure 2.2, the $Jaccard$ similarity between filter a (representing the name ANA) and filter **b** (ANE) is equal to $\frac{4}{6} = 0.66$.

In summary, the BF is relatively easy to understand and implement compared to other more complex techniques. Also, the computation over the BF bit-vectors is efficient and provides a good enough accuracy of the calculated similarity. Therefore, given these advantages, the BF is considered the anonymization method of choice for PPRL.

2.1.5 Differential Privacy

Differential Privacy (DP) is a privacy preservation technique that addresses the paradox of learning nothing about an individual while learning meaningful information about a specific population [45]. For instance, the DP can be used to reveal relevant medical information

regarding the treatment outcome (e.g., COVID-19 vaccination) over a specific population (e.g., diabetic patients) without revealing the identity of the individuals.

Unlike the privacy-preserving methods presented earlier, DP's basic idea is to introduce noise (by modifying) to the original dataset or queries' on a dataset. The concept of differential privacy technique based on the probability model was first introduced by C. Dwork [44]. Moreover, DP does not require prior knowledge of adversaries. Differential privacy aims to ensure that any query or database modification output result should not reveal enough information about any individual that leads to its identification. In the following, we present the formal definition of Differential Privacy.

Definition 2 (Differential Privacy)**.** *Let M be a randomized computation mechanism. M provides ϵ-differential privacy if for any datasets D and D' with symmetric difference $D\delta D' = 1$, in a set of possible outcomes $S \subseteq range(M)$, such as:*

$$\Pr[\mathcal{M}(D) \in S] \leq \Pr[\mathcal{M}(D' \in S)] \times e^{\epsilon} \tag{2.2}$$

In other words, M is ϵ-differentially private iff. D and D' differ in one element for any set S of all possible outputs of M. The parameter ϵ (also named as privacy budget) controls M's privacy level. For instance, lower values of ϵ mean more robust privacy guarantees, indicating that any individual in the database will introduce a small change on M. Section 2.7.2 shows the usage of DP mechanisms in the Deep Learning training context to protect individuals' privacy.

2.2 Difference between Privacy and Security Protocols

As presented in the previous sections, privacy is not just a technical task of controlling some assets; privacy is a fundamental human right. Therefore, data privacy issues are discussed in many different research communities, often in different terms or with the same terms denoting different concepts [4; 21; 159]. Thus, this section clarifies the difference between Data Privacy-Preserving and Security Protocols.

In summary, security protocols enable secure communication over insecure channels by employing cryptographic primitives. On the other hand, privacy protocols intend to preserve

the privacy of individuals. Moreover, privacy protocols can use secure channels, depending on the task [21]. In this context, security protocols can be broken without breaking the underlying cryptography, and some privacy protocols can be broken without breaking the underlying security.

In the PPRL context, the boundaries between privacy and security are not clear [21]. Moreover, the data anonymization step can be perceived as a security measure to preserve the privacy of the records. However, there are PPRL anonymization techniques that do not rely only on cryptographic primitives. For example, several techniques, such as Bloom and Flip [153], employ Differential Privacy techniques (a data privacy tool) to preserve the privacy of individuals. Making it hard to establish whether the PPRL data anonymization step is a security or privacy tool.

Knowing that the boundaries between privacy and security in the PPRL are blurred, we use the term security to refer to establishing a secure communication channel or environment. In contrast, the term privacy refers to any task, technique, and tool employed to ensure the privacy of the data. In addition, other researchers suggest that the boundaries between privacy and security protocols and privacy attacks present a new challenge for security and privacy protocol analysis [21].

2.3 Privacy Preserving Record Linkage

In this section, we present details about the PPRL process. In the following, we present concepts of adversary models, roles, and evaluations employed in the PPRL context.

2.3.1 PPRL Adversary Models

In the context of cryptography and information security, an adversary (or opponent) is a malicious participant who tries to make the use of cryptographic systems unfeasible by comprising the privacy and integrity of data protected by security and encryption systems [54]. The literature also classify the adversaries in different models from a malicious to an honest adversary model [6; 77; 120]. In the PPRL context, we highlight three adversary models presented as follows [177]:

1. **Honest-but-curious** (HBC): assumes that the participants will follow the protocol correctly but will try to obtain additional information from the data received during the execution of PPRL. This model does not prevent participants from conspiring with each other (collusion) in order to discover confidential information from other participants [179];

2. **Malicious**: participants behave arbitrarily in relation to the protocol, and may not follow it, send random values, or even cancel the protocol at any time. Few studies have been carried out in PPRL using this adversary model due to the difficulty of predicting how a malicious adversary can bypass the protocol [179];

3. **Covert and accountable computing**: tries to ensure, with high probability, that the participants who follow the protocol will be able to detect the malicious actions of the participants who do not follow the protocol [77; 120; 179]. Created to overcome the limitations of HBC, which can only be used in scenarios where all participants trust each other.

According to recent literature review works [30; 177; 178], the majority of approaches considers the HBC model, a few approaches consider malicious adversary models, and there is no approach the consider the Covert adversary model.

2.3.2 PPRL Roles and Number of Parties

The PPRL process can be classified according to the number and the roles of participants. The literature lists two distinct roles within a PPRL process: i) the PPRL parties; and the Linkage Unit (LU). The PPRL parties (database owner or custodian) are the providers of the databases to be linked [177]. Depending on the PPRL approach, the database custodian may or may not participate in the comparison step. A LU is a special party that participates in the PPRL process such that it may or may not be external to the dataset custodian [30]. In general, a LU is a party that other PPRL participants trust and acts as a trusted third party to conduct the Comparison and Classification steps [30; 177].

Considering that we have two dataset custodians that wants to execute a linkage over their private data, we could employ two protocols. These protocols are classified according

to the existence of a LU. Those that use a LU named as *three-party protocol* and those that do not use a LU *two-party protocol* [135].

The *three-party protocol* generally employs the LU as a trusted third party to carry out the steps of the PPRL. Figure 2.3 illustrates the *three-party protocol*. The PPRL starts after the initial parameter exchange (step 1). Next, data is pre-processed and anonymized by the participants in order to be sent to the LU (step 2). Finally, the LU performs the remaining steps of PPRL (Comparison and Classification). In the end, the LU sends to the participants only the identifiers (*id*) of the entities that are present in all databases [177].

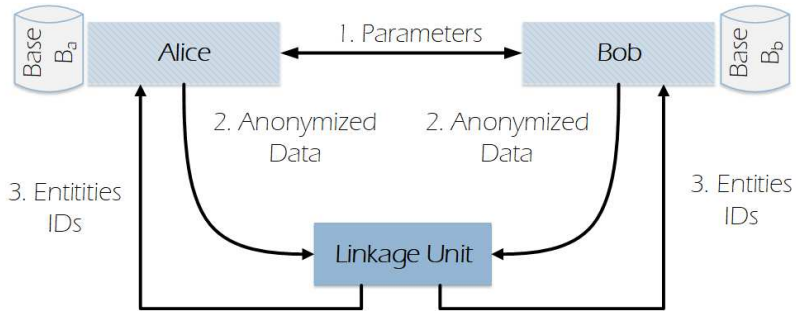


Figure 2.3: PPRL three-party protocol.

Although these protocols are more efficient due to centralized processing of records, the LU may collude with other PPRL parties to reidentify the entities of another custodian's dataset. Also, sending all encoded datasets to the LU can potentially increase the risk of privacy attacks because the LU can use this information to reidentify a sensitive plain-text value of a database. We explain privacy attacks in Section 2.4.

In turn, protocols that do not use a LU (*two-party protocols*) carry out all the stages of PPRL on the dataset custodian themselves. Figure 2.4 illustrates this protocol. As in the previous protocol, the initial parameters are exchanged; then, data is anonymized and sent to the other participants (steps 1 and 2). After the distribution of anonymized data among the PPRL parties, the remaining steps of PPRL are performed by each dataset custodian. In step 3, each participant informs the others which of their entities were classified as similar [177].

By sharing anonymized data with the other participant, this protocol needs to use more sophisticated and, therefore, more complex anonymization and Comparison techniques in order to ensure that data privacy is preserved during the execution of PPRL [80].

The two-party or three-party protocols are applied when PPRL is conducted with two

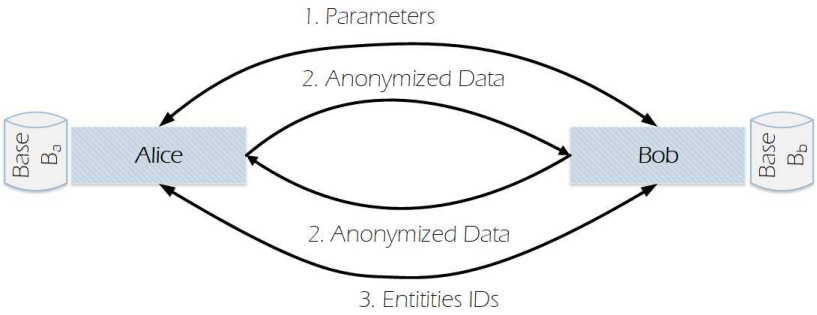


Figure 2.4: PPRL tow-party protocol.

datasets. However, sometimes multiple, more than two, datasets are provided as input to the PPRL process. In this scenario, we employ the multi-party protocol and the multi-party protocols with an LU protocol. The multi-party protocol is analogous to the two-party protocol,i.e., the PPRL is conducted without a LU. In turn, the multi-party protocols with an LU protocol use the LU to perform some steps of PPRL, similar to the *three-party protocol* [177; 30; 80].

2.3.3 PPRL Evaluation Measures

The final step of PPRL is to measure the performance of the linkage. In PPRL, performance is measured by evaluating the efficiency, effectiveness (quality), and privacy of the techniques used. In the following, we present a brief explanation of the aforementioned evaluation measures.

Effectiveness (Quality) Evaluation

Three well-known quality measures (Precision, Recall, and F-measure) from Information Retrieval are employed to evaluate a PPRL process's effectiveness. These measures are defined as $Precision = \frac{|\Gamma_e \cap \Gamma_r|}{|\Gamma_r|}$, $Recall = \frac{|\Gamma_e \cap \Gamma_r|}{|\Gamma_e|}$, and $F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall}$, where Γ_e represents the set of all existing duplicated entities between the data sources (true matches), and Γ_r is the set of entities identified as duplicate by an approach. Intuitively, low precision indicates that false correspondences were made, while low recall implies that the approach missed true matches. In turn, F-measure ($F1$) is the harmonic mean between recall and precision.

Efficiency (Scalability) Evaluation

Efficiency measures the scalability of a PPRL process. Efficiency can be measured using metrics based on the computational environment or the number of candidate record pairs generated by a PPRL approach. Next, we present some of the efficiency metrics:

1. **Runtime and Memory Usage**: this measure is based on the execution time and the amount of memory employed to conduct the PPRL process. It can include the time and memory required for the different steps of the PPRL process, e.g., pre-processing, Indexing, Comparison, and Classification;

2. **Communication Usage**: usually, PPRL employs protocols, and every protocol has a communication cost to exchange the messages. Therefore, it is important to measure how much information is exchanged between the parties. The communication usage is measured as the number of messages and their sizes (in bytes). Also, the total number of messages transmitted among the parties or the average number of messages sent;

3. **Reduction Ratio (RR)**: a scalability measure commonly used in PPRL which provides information about the size of the comparison space generated by a blocking technique. RR requires two metrics: the number of candidate record pairs generated by blocking; and the number of all possible record pairs that can be generated across the databases.

Privacy Evaluation

Privacy is a fundamental aspect of any PPRL solution. Several standard information theory measures such as entropy, Information Gain (IG), and Relative Information Gain (RIG) have been used in PPRL to assess the privacy of reidentifing a record in a database [42; 81]. However, Vatsalan et al. [173], demonstrated that the aforementioned methods fail to quantify the privacy of the PPRL process.

In this sense, an adequate evaluation method to a PPRL process is the simulation paradigm. The paradigm assesses the approach through the messages (information) exchanged by the parties and LU during a simulated PPRL execution. The simulation must be executed according to the guidelines proposed by Lindell et al. [103]: the messages granted in the simulation must be the same as an adversary would have access to use in a

real attack. The simulation fails if an adversary learns anything different from the expected output.

2.4 PPRL Privacy Attacks

As previously stated, PPRL parties must share their entities among each other or with a Linkage Unit. In order to preserve the privacy of the shared entities, the parties employ anonymization techniques to prevent the reidentification of the shared entities. However, the usage of anonymization techniques has limitations, and an opponent can use the anonymized entities (i.e., data) to break the confidentiality and privacy of the shared entities. Therefore, approaches, techniques, protocols, and data used in PPRL are vulnerable to the five types of attacks described as follows [179; 182]: collusion, frequency, dictionary, composition, and cryptanalysis.

The **collusion** attack occurs when two or more participants team up to discover information about other participants' data. For example, suppose the PPRL (using a *three-party protocol*) is performed in a scenario where the participants are malicious. In that case, a malicious participant could collude with the Linkage Unit in order to violate the confidentiality of the data of the other participants [177].

The **frequency attack** consists of observing the frequency at which a given value occurs in an anonymized dataset and comparing it with the frequency at which this value occurs in a known dataset [177]. For example, in a database of patients diagnosed with breast cancer, the name Maria is expected to appear more frequently than the others, since Maria is the most common name for women in Brazil [69]. Thus, by observing the frequency of anonymized data in a database, an opponent can break the confidentiality and privacy (reidentify) of the original data.

In a **dictionary attack**, an opponent anonymizes a list (dictionary) of words (values) using various anonymization techniques. Then, the dictionary values are compared against the target anonymized data in order to reidentify the entities [179]. For example, the dictionary attack on a database of AIDS patients can be performed as follows: an opponent knowing the technique used to anonymize the data, and knowing that an attribute represents the estate of residence of the patients, can easily build a dictionary, with only 27 values (Brazil), and

identify the state (of the federation) of AIDS patients.

In a **composition attack**, auxiliary information (external information) is used to break the confidentiality and privacy of anonymized data [60]. Part of the rationale for the composition attacks is described by Sweeney [161] who, using only three attributes (postal code - 5 digits, gender, and date of birth), was able to reidentify almost 90% of the US population (216 out of 248 million individuals).

Finally, **cryptanalysis** attacks use the combination of multiple attacks to obtain information about the key or algorithm used to anonymize the data. For instance, BF can be vulnerable to this type of attack depending on the anonymization parameters and the volume of data used. As a result, the participants exchange information/parameters about the length (number of bits) and algorithms used in the Bloom Filters [99; 125]. In Chapter 3, we provide a summary of the current attacks and limitations of the BF.

2.5 Blockchain

Blockchain, also called distributed ledger, is essentially an append-only DataBase Management System (DBMS) maintained by a set of nodes that do not fully trust each other [50; 122]. In other words, Blockchain is a technology that maintains the states and the historical transactions, using a peer-to-peer network, without any central node to enforce compliance of the rules. Blockchain provides immutable storage (temper evidence guarantee) of transactions in a chain of blocks, by storing data (records) in blocks that are linked using cryptography tools, i.e., the previous block hash, the transaction owner signature, and the identifier (id) of the machine (miner) that executed the transaction. Figure 2.5 provides a high-level illustration of the Blockchain internal elements.

For each block of the Blockchain, Figure 2.5 presents a brief illustration of the most relevant data structures of the technology. Notice that each block contains the indication (hash) of the previous block to maintain the order of the data and transactions. A critical element is the nonce, a considerable large random number (32 to 4,096 bits). The nonce represents a cryptographic puzzle that needs to be solved by a Blockchain node (a.k.a., miner). Then, with this puzzle resolved, the miner can append the block to the chain in a process named as

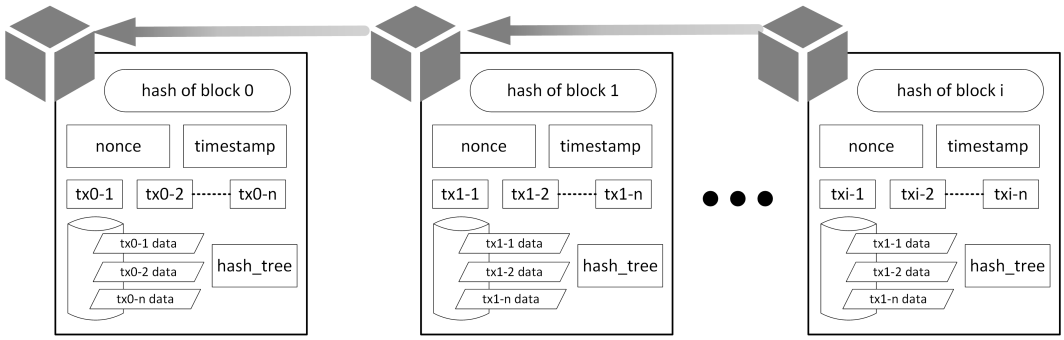


Figure 2.5: Blockchain data structures.

mining. For instance, Bitcoin miners need to guess a valid nonce as they perform multiple attempts to calculate a block hash that meets certain requirements (i.e., a prime number with more than 64 bits that starts with a certain number of zeros).

Another important element is the timestamp; this element is used to provide proof of computing (or existence) to the block data. For instance, this element can be used to track public records, such as Educational Degrees[2].

The data is stored in a public database (a.k.a, distributed ledger) by the transactions (tx). Notice that one block can execute multiple transactions. It is worthwhile to mention that data and metadata (e.g., the miner how executed the transaction) are stored in a database - a.k.a, distributed ledger.

For a record (block) to be effectively stored in the distributed ledger, the record needs to be replicated in every node of the Blockchain. In order to guarantee the consistency of the ledger, a party sends the record within a transaction with a cryptographic puzzle to be mined by the Blockchain nodes. After the first miner solves the cryptographic puzzle (embedded with the transaction), the other nodes will verify the transaction execution and store it in its local ledger [185]. In other words, every node in the network verifies all transactions, replicating the data and the computation in all nodes of the network. Therefore, all inputs and outputs of the transaction are publicly available from every Blockchain network member.

Regarding the verification of the block, in Figure 2.5, we have an element that represents one of the most important verification tool of the technology, the Hash Tree. The Hash Tree is a modified Merkle Tree [71] that uses specials hash functions to sign the data stored on

[2]http://www.ufpb.br/primeiros-diplomas-digitais-da-ufpb/

the chain. Figure 2.6 presents a simplified version of the Hash Tree mechanism.

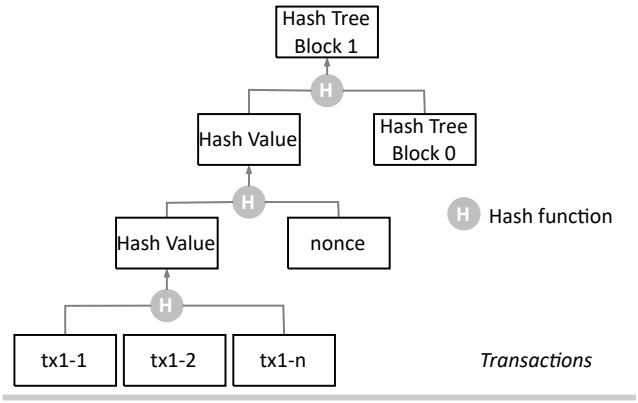


Figure 2.6: Hash Tree example.

The Hash Tree hash uses encryption and hash functions to digital sign every peace of data stored on the chain. Notice that the block 1 Hash Tree uses multiple hashes for each data and nonce of the block data. Moreover, it uses the Hash Tree of the previous block to compute the final hash of the block. Consequently, the uses of Hash Tree and digital signatures make it hard for an attacker to modify the data stored on the Blockchain. It is worthwhile to mention that the chance for an attacker to successfully execute a malicious modification of the data decreases as the block number increases (longer chains). The multiply signatures and hashes justify such difficulty.

Considering the access (publicity) to the transaction data in the Blockchain, we can classify the Blockchain into two environments: i) public (or permissionless) Blockchains, where anyone can join the network (e.g., Ethereum and Bitcoin); and ii) private (or permissioned) Blockchains, where there is an access control to manage who can join the network (e.g., Hyperledger, Ethereum, and Microsoft Blockchain Confidential Consortium Framework) [168].

In both scenarios, public and private Blockchains, it is expected that some of the nodes present a malicious (Byzantine) behavior. In other words, some nodes will collude to change the transactions stored in the distributed ledger or try to prevent an honest party from storing its transaction in the distributed ledger. For instance, if a set of malicious nodes claims that a legitimate transaction performed wrong computations, this transaction will not be stored in the distributed ledger [168]. To prevent this, the Blockchain employs the consensus mechanism.

The most common consensus mechanism employed in public Blockchains is the Proof-

of-Work (PoW). In this mechanism, only a miner which has successfully solved a computationally laborious puzzle can append data to the Blockchain. The PoW is very effective in preventing the misbehave of a malicious party. However, it has a high computational cost - imposing a delay in the transaction execution [5]. In private Blockchains, we remark the usage of Proof-of-Authority (PoA) in order to add a block to the chain. This mechanism invokes the nodes to vote if the block can be appended to the Blockchain. If the nodes reach a consensus about the block, it will be added permanently to the Blockchain; otherwise, the block will be rejected. Thus, to employ the PoA, every node needs to be identified and authenticated.

The PoA is faster than PoW since reaching a consensus is simpler and faster than solving a cryptographic puzzle. However, PoA can be employed in a scenario where the parties have some level of trust in others, in contrast to PoW, where the parties do not need to trust others [92].

One interesting aspect of the Blockchain is the fact that we can upload applications written in a Turing complete programming language to be executed by the nodes. This application, also known as Smart contracts, allows trustworthy code execution in an untrusted environment. In other words, Smart contracts allow the nodes of a Blockchain to perform computation (or transactions) without a trusty-third party.

A smart contract can be executed when a transaction is performed in the Blockchain network. This can be compared to a stored procedure invoked by a transaction in a DBMS. It is worthwhile to mention that the smart contract execution inherits all characteristics provided by the Blockchain distributed ledger: immutable (temper evident), public available (auditable) and verified by all nodes (trustable computation) [168].

2.6 Transfer Learning

Data mining and Machine Learning (ML) techniques have been widely and successfully used in many real-world applications where information (training data) can be extracted in order to predict upcoming results [110]. Traditional ML is characterized by training and testing data having the same input feature space and the same data distribution. In specific scenarios, obtaining training data that match the feature space and predicted data distribution character-

istics of the test data can be difficult and expensive [134]. Furthermore, in privacy-preserving applications, such as PPRL, obtaining any labeled data can be considered an impossible task.

To tackle the aforementioned problem, Transfer Learning (TL) is used to create or improve a learner (i.e., a classifier) from one domain by transferring information to a related domain. To provide a real-world example that employs TL concepts, consider two people who want to learn how to play the Cello. One person has extensive music knowledge and plays the violin for more than two decades, and the other person has no experience and never had played any instrument. The person who knows how to play the violin will probably learn how to play the Cello faster than the other person. This is expected because the violin player transfers the previous knowledge to the task of playing the Cello [186].

A transfer learning process is illustrated in Figure 2.7. The process on the left corresponds to a traditional machine learning process. The transfer learning process is represented on the bottom and right of the figure. As we can see, transfer learning makes use of the knowledge produced by another learning system (named as source). This figure shows a key concept of transfer learning: it mitigates the lack of training data problem by a learning system with more knowledge gained from a source learning system.

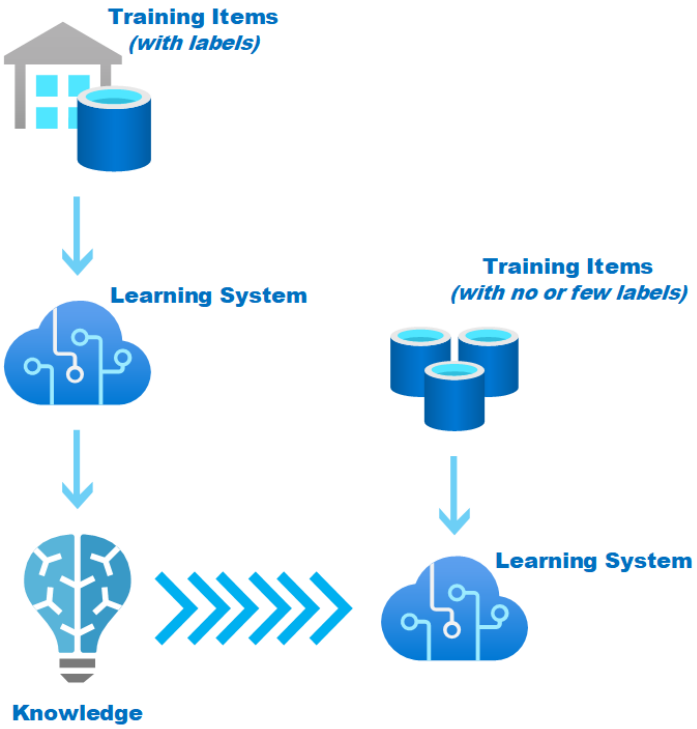


Figure 2.7: Transfer Learning workflow.

In summary, TL deals with how ML systems can quickly be adapted to new situations and tasks [192]. It gives ML systems the ability to leverage auxiliary data and/or models to help solve target problems even when no labeled data is available in the target domain. In the following, we formalize the concept of Transfer Learning.

2.6.1 Definition of Transfer Learning

In order to present the definition of the TL, first, we have to present the concept of "domain" and "task". A domain D consists of two components: a feature space X and a marginal probability distribution $P(X)$, where each input instance $x \in X$. In general, if two domains are different, then they may have different feature spaces or different marginal probability distributions.

Given a specific domain, $D = \{X, P(X)\}$, a task T consists of two components: a label space Y and a function $\Xi(\cdot)$, formalized as $T = \{Y, \Xi(\cdot)\}$. The function $\Xi()$ is a predictive function that can be used to make predictions on unseen instances $x \in X$. From a probabilistic viewpoint, $\Xi(x)$ can be written as $P(Y|X)$. In the following, we present the definition of Transfer Learning.

Definition 3 (Transfer Learning)**.** *Given a source domain D_s and a learning task T_s, a target domain D_t and a target learning task T_t, transfer learning aims to help improve the learning of the target predictive function $\Xi_t(\cdot)$ for the target domain using the knowledge in D_s and T_s, where $D_s \neq D_t$ or $T_s \neq T_t$.*

2.6.2 A Categorization of Transfer Learning Techniques

Considering the TL definition, we could classify it according to several aspects. In cases where $X_s \neq X_t$, it is defined as **heterogeneous transfer learning**; in other words, the feature spaces are different in both domains. In turn, when the source and target domains share the same feature space, $X_s = X_t$, it is named as **homogeneous transfer learning**.

In Table 2.1, we compile the relationship between traditional machine learning and several TL settings. Based on different scenarios between the source and target domains and tasks, it is possible to categorize TL under three classes: i) inductive transfer learning; ii) transductive transfer learning; and iii) unsupervised transfer learning.

Table 2.1: Summary of TL techniques, from [132]

Learning Settings		Source and Target Domains	Source and Target Tasks
Traditional Machine Learning		the same	the same
Transfer Learning	Inductive Transfer Learning	the same	different but related
	Unsupervised Transfer Learning	different but related	different but related
	Transductive Transfer Learning	different but related	the same

Notice that the traditional machine learning approaches consider $D_s = D_t$ and $T_s = T_t$, while TL considers several other configurations. In the inductive transfer learning setting, the source and target tasks are different, even when the source and target domains are the same. In this case, some labeled data in the target domain is required to induce an objective predictive model $f_t(\cdot)$ for use in the target domain.

In the Transductive transfer learning setting, the source and target domains are different while the tasks are the same. In this situation, no labeled data is available in the target domain, while many labeled data in the source domain are available. Finally, unsupervised transfer learning is similar to inductive transfer learning, with a focus on solving unsupervised learning tasks, such as clustering, classification, dimensionality reduction, and density estimation.

Another important aspect of TL is which part of knowledge can be transferred across domains or tasks. The form of transfer the knowledge is categorized into four general transfer categories [131]. Figure 2.8 illustrates that transfer learning makes use of not only the data in the target task domain as input to the learning algorithm but also any of the learning processes in the source domain, including the training data, models, and features.

The first transfer category is named as instance-based transfer learning, also called Domain Adaptation, which assumes that certain parts of the source domain's data can be reused for learning in the target domain. In this case, instances from the source domain need to be reweighted to correct the marginal distribution differences. These reweighted instances are then directly used in the target domain for training (examples in Huang [196] and Yang [192]). These reweighting algorithms work best when the conditional distribution is the same in both domains [192].

The second category is the feature-representation transfer learning [132]. This category

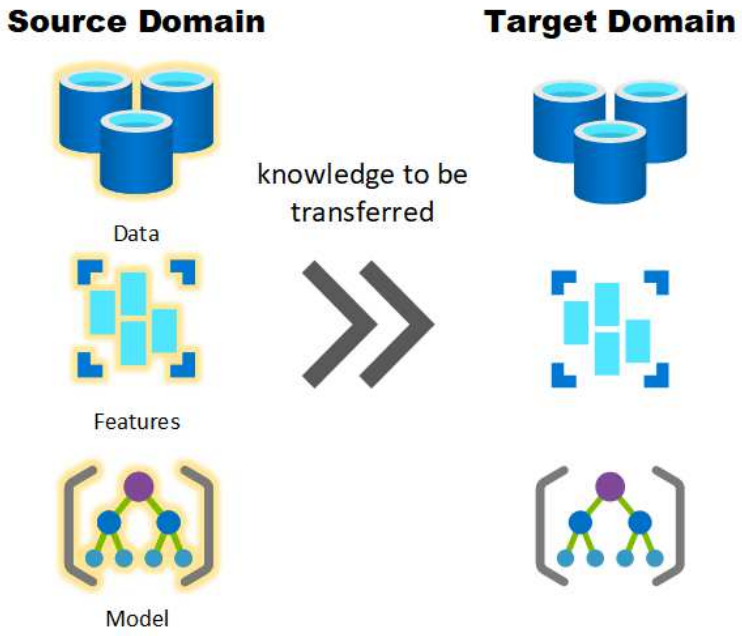


Figure 2.8: Knowledge to be transferred.

intends to learn a ”good” feature representation for the target domain. The knowledge utilized to transfer across domains is encoded into the learned feature representation. With the new feature representation, the performance of the target task is expected to improve significantly. This TL approach is used in a widespread scenario. However, heterogeneous transfer learning benefits in most of this form of TL due to the difference between the domains [132].

The third category intends to transfer knowledge through shared parameters, distributions of the hyper-parameters of the models, or by creating multiple source learner models and optimally combining (ensemble) the models. This form is known as parameter-based transfer learning [192].

Finally, the last category is the relational transfer [196], which deals with learning for relational domains. The assumption of this form is that some relationships among source and target domains are similar by employing statistical relational learning techniques [132; 106; 186].

2.6.3 Data Relatedness

In instance-based Transfer Learning, the relatedness of the source to the target domain data is crucial to the learning task, in our case, to the classification of the entity pairs. For instance,

using the example of the musicians, the sheet music (score) for a guitar is much more related to the electric guitar than an oboe score. In the music context, there are methods to measure how similar a score is. This section presents how to measure the relatedness of two datasets (such as the music scores).

In some contexts, such as the PPRL one, it is not usual to have abundant labeled data (training data) from a source domain. However, to train a classifier that performs well on a target domain with a different distribution and no labeled training data is not straightforward. It depends on a series of aspects, e.g., if the training instances from the source are representative to the target domain task. To determine if a classifier trained from the source domain can perform well on the target domain, Ben-David et al. [14; 13] bound the classifier's target error in terms of its source error and the divergence between the two domains. Moreover, they employ a classifier-induced divergence measure that can be estimated from finite, unlabeled samples from the domains.

Before presenting the work of Ben-David et al. [14; 13], it is necessary to formalize some concepts and assumptions employed by the authors. Consider a domain, $D = (X, P(X))$, where $x \in X$ has a specific label $y \in Y$ defined by a perfect label function f. Thus, using these concepts, it is possible to map the error, a.k.a. risk, (ϵ_s) of a source domain comparing the hypotheses prediction (h) against the actual label (f), $\epsilon_s(h, f) = \exists_{x\ X_s}[|h(x) - f(x)|]$.

The solution proposes a symmetric hypotheses space $H\Delta H$ of finite VC dimensions where the set of hypotheses is defined by $g \in H\Delta H \iff h(x) \oplus h'(x)$. In other words, every hypothesis $g \in H\Delta H$ is the set of disagreements between two hypothesis.

Consider this symmetric hypothesis space. It is possible to use $H\Delta H$-divergence to bound the error of two hypotheses and, consequently, measure the divergence between a source and target domain. The divergence is measured by considering the error of the domains, and it is defined by Equation 2.3:

$$d_{H\Delta H}(D_s, D_t) = \frac{1}{2}\hat{d}_{H\Delta H}(U_s, U_t) + \sqrt{\frac{2dlog(2m') + log(\frac{2}{\sigma})}{m'}} + \lambda \tag{2.3}$$

where U_s, U_t are unlabeled samples of size m' each, from D_s and D_t, respectively. For any $\sigma \in (0, 1)$, with probability at least $1 - \sigma$ (over the choice of the samples). The term $\hat{d}_{H\Delta H}$ is the empirical $H\Delta H$-divergence.

The $\hat{d}_{H\Delta H}$ can be computed by finding a classifier that attempts to separate one domain from the other. The basic intuition is to label the instances from U_s (i.e., with 0) and U_t (i.e., with 1), indicating the origin of each instance. Afterwards, a classifier is trained to discriminate between source and target instances. The $\hat{d}_{H\Delta H}$ is computed from Equation 2.4.

$$\hat{d}_{H\Delta H} = 2\left(1 - \min_{h \in H\Delta H}\left[\frac{1}{m}\sum_{x:h(x)=0} I[x \in U_s] + \frac{1}{m}\sum_{x:h(x)=1} I[x \in U_t]\right]\right) \tag{2.4}$$

where $I[x \in U]$ is the binary indicator variable which is 1 when $x \in U$.

In summary, domains with smaller divergence values have a higher chance of training a classifier in a source domain, and this classifier performs well on a target domain. In this thesis, we present an approach that directly exploits this idea in Chapter 5.

2.7 Deep Learning

Neural networks (NN) have been employed with outstanding results in several contexts, including as classifiers for RL tasks [46; 121; 184]. NN are composed of a series of parameterized functions, also named layers. In these layers, a set of transformations and non-linear functions are employed to perform classification/prediction (output) tasks over an input. Sigmoid and rectified linear units (ReLUs) are examples of functions employed to perform non-linear operations over the inputs [64]. In summary, NN are composed of a set of layers (with multiple functions) in which we can use a known input (labeled data) to train the layers of the network, set the parameters of the functions, and perform classification or prediction of a finite set o input/output example [72].

Neural Networks employ a loss function (L) to define the NN parameters. Moreover, L represents the penalty for wrongly fitting the training data over the task performed by the NN. In other words, L is employed to define the increment/decrement of the NN layer's parameters.

The NN's training process consists of finding a loss (θ) that results in a as small as possible loss value. The average of the loss ($L(\theta)$) over the training instance $[x_1, \cdots, x_n]$ is defined by $L(\theta) = \frac{1}{n}\sum L(\theta, x_j)$. It is worth mentioning that several loss functions exist

- e.g., Mean Squared Error (MSE) and Cross-Entropy (CE) - and, therefore, the loss function must be carefully chosen considering the input/output data, NN architecture, and task performed by the NN [72].

In general, the loss function L is non-convex, and challenging to find its minimum value (minimize). Therefore, optimizers such as Adaptive moment estimation (Adam) or Stochastic Gradient Descent (SGD) algorithms are required. Both of the aforementioned algorithms compute a mini-batch and use the loss function to update the layer's parameters at each step [72; 86]. For instance, SGD forms a batch B considering the training instances and calculates $g_B = \frac{1}{|B|} \sum_{b_i \in B} \nabla_\theta L(\theta, b_i)$ as the gradient of the loss. Then, θ is updated towards the local minimum (gradient direction).

This section presents the basic concepts of NN and the methods and tools employed to train NN. In the following sections, we detail the architecture of a widely used NN for pattern matching, and a privacy-preserving training method for NN.

2.7.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) have been successfully employed in a wide range of applications, especially in applications that aim to recognize features or patterns in signals. For instance, medical applications (i.e., detection of heart diseases over electrocardiograms [87]), and computer vision application (e.g., face recognition [64]).

The CNNs' basic idea is to extract features from the input signal (i.e., image) in convolutional layers. Unlike traditional approaches, in Deep Convolutional Networks (DCNN), features are automatically discovered, starting with low-level features up to high-level ones. In traditional machine perception, the features are manually configured. To illustrate how CNN works, consider the face recognition example depicted in Figure 2.9.

The layers of a CNN can be arranged so that low-level features are extracted in the first layers of a network, and later on the network, these low-level features are combined to form higher-level features. For instance, in Figure 2.9, the first layers will extract the face's edges, colors, and shapes (low-level features) of the input (raw data) [73]. Afterward, the low-level features will be combined into mid-level features, such as eyes and noses. Finally, the mid-level features will be combined to distinguish the faces of different persons.

In order to extract features and perform classification task, the CNNs' comprises two

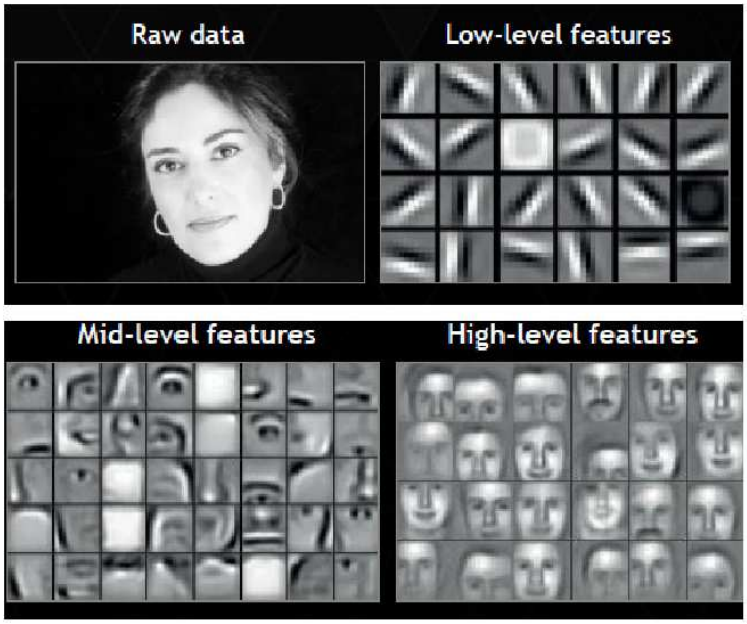


Figure 2.9: Example of CNN for face recognition - *source: Tomodan et. al. [166].*

types of layers: convolution and pooling layers. Convolution layers highlight patterns in the CNN input (i.e., an image), whereas pooling layers downsample the output of convolutional layers to select more representative elements of the operations. It is important to mention that at the end of CNN, there is a fully connected neuro network (Dense Network) to perform the classification or recognition task [73; 64].

Convolution layers are composed of multiple convolution filters, where each filter determines whether a particular local feature/partner is present in the input [73]. The convolution filter adds each element of the layer input (a matrix) to its local neighbors, weighted by a kernel in a series of matrix multiplication operations. The convolution filter is defined by Equation 2.5.

$$conv(X_{m\times n}, Y_{m\times n}) = \sum_{i=0}^{m-1}\sum_{i=0}^{m-1} x_{(m-i,n-j)}y_{(i+1,1+j)} \tag{2.5}$$

where X and Y are matrices of $m \times n$, such that $x_{(m,n)} \in X$ and $y_{(m,n)} \in Y$. Notice that the convolution operation can be performed over matrices of different dimensions.

For instance, assume that the input of a convolutional layer is a 4×3 matrix, such that $I = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \\ a_{41} & a_{42} & a_{43} \end{vmatrix}$ and the convolutional will be performed considering a 2×2 kernel, such that $K = \begin{vmatrix} \alpha & \beta \\ \gamma & \delta \end{vmatrix}$. The resulting convolution will be:

$$conv(I,K) = \begin{vmatrix} a_{11}\alpha + a_{12}\beta + a_{21} + a_{22}\delta a_{12}\alpha + a_{13}\beta + a_{22} + a_{23}\delta \\ a_{21}\alpha + a_{22}\beta + a_{31} + a_{32}\delta a_{22}\alpha + a_{23}\beta + a_{32} + a_{33}\delta \\ a_{31}\alpha + a_{32}\beta + a_{41} + a_{42}\delta a_{32}\alpha + a_{33}\beta + a_{42} + a_{43}\delta \end{vmatrix}_{1\times 3}$$

Convolution layers use several filters to detect a variety of different-patterns (e.g., oriented edges and shapes) in the input. Moreover, filters are employed for a specific classification task. Finally, it is worth mentioning that CNNs typically apply the ReLU activation function to the convolved input [72].

The second particular layer, the pooling layer, provides a way to summarize a large input into a smaller summary input. There are several ways of performing pooling, e.g., average pooling and max pooling. The pooling operation summarizes each non-overlapping input value in blocks of the dimensions of the kernel. Moreover, the max polling employs the maximum value in the block, while the average pooling uses the average value in the block. For instance, the function $MaxPool(I, K_{2\times 2})$ represents the max polling operation, and K represents a kernel with 2×2 dimension. The output of $MaxPool$ is illustrated below:

$$MaxPool\Big(\Big|\begin{smallmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 2 & 5 \\ 1 & 3 & 7 & 5 \\ 5 & 2 & 3 & 4 \end{smallmatrix}\Big|, K_{2\times 2}\Big) \rightarrow \begin{vmatrix} 4 & 5 \\ 5 & 7 \end{vmatrix}$$

2.7.2 Differential Privacy Optimizer

As previously mentioned, NN and other ML techniques usually employ representative (large) datasets that could contain sensitive/private information. In a privacy-preserving context, such as the PPRL one, the models should not expose private information in the training datasets. Moreover, a malicious adversary could use the internal representations of deep neural networks to reveal details of some instances of the training data [1; 102]. For example, Fredrikson et al. proposed a model-inversion attack that retrieves images from a facial recognition system [59]. In order to preserver the privacy of the training data, several techniques for private learning were developed [1; 156; 102].

Abadi et al. [1] propose an approach which controls the influence of the training data during the training process, particularly Abadi et al. modify the SGD to consider privacy during the computations. Moreover, Abadi et al. [1] incorporated Differential Privacy in

the SGD. Algorithm 1 outlines the modified SGD for training a model with parameters θ by minimizing the empirical loss function $L(\theta)$.

Algorithm 1: Differentially private SGD - *adpated form : Abadi et. al. [1].*

input : training data $[x_1, \cdots, x_n]$, loss function $\Big(L(\theta) = \frac{1}{n}\sum L(\theta, x_j)\Big)$, learning rate ($\eta_t$), noise scale ($\alpha$), group size ($GS$), gradient norm clip ($C$)

output: θ_T and compute the overall privacy cost (ϵ, δ) using a privacy accounting method.

1 $\theta_0 \leftarrow$ `initialize(`*random*`)`

2 **for** $t \in [T]$ **do**

3 // random sample with sampling probability $\frac{GS}{N}$

4 $GS_t \leftarrow$ `sample(`*GS*`)`

5 // compute the gradient

6 **for** $x_i \in GS_t$ **do**

7 $g_t(x_i) \leftarrow \nabla_\theta L(\theta, x_i)$

8 **end**

9 $\bar{g}_t(x_i) \leftarrow$ `clipGradient(`$g_t(x_i)$,C`)`

10 $\tilde{g}_t(x_i) \leftarrow$ `addNoise(`$\bar{g}_t(x_i)$,GS,α,C`)`

11 // descent

12 $\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{g}_t$

13 **end**

At each step of the algorithm, the gradient ($\nabla_\theta L(\theta, x_i)$) for a random subset of examples (line 4). The norm of each gradient is clipped (line 9), and noise is added (line 10) to preserver the privacy, and move towards the opposite direction of the noisy gradient - $\bar{g}_t(x_i)$, line 12. Notice that there are two new methods in the SGD, norm clipping (line 9) and noise addition (line 10). Figure 2.10, illustrates the impact of the methods over the gradient.

In the first stage (a), the gradient is calculated as usual in the SGD - line 7 of the Algorithm 1. In stage b, a threshold is employed to perform the norm clipping. The clipping operation ensures that if $||g||_2 \leq C$, then g is preserved, whereas if $||g||_2 > C$, it gets scaled to the norm C. In stage c, a noise (α) is added to the clipped gradient ($\bar{g}_t(x_i)$), resulting a a final gradient ($\tilde{g}_t(x_i)$) close to the original.

As mentioned, the method described in this section ensures Differential Privacy guaran-

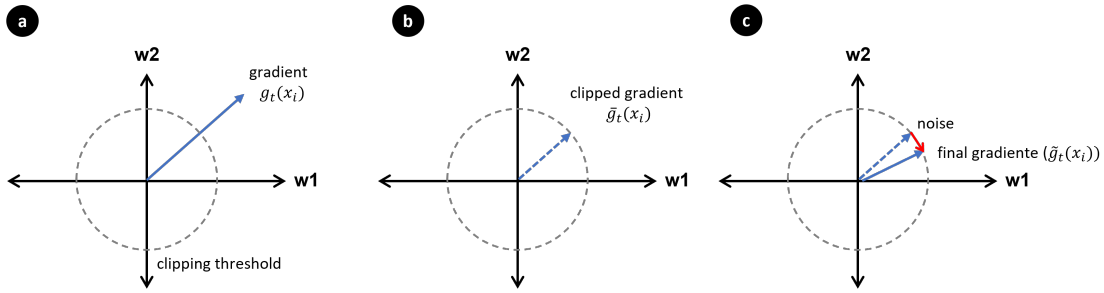


Figure 2.10: Graphical illustration of Algorithm 1.

tees for the training data. The privacy budget ϵ is defined by the following theorem.

Theorem 1. *There exist constants c_1 and c_2 so that given the sampling probability ($q\frac{GS}{N}$) and the number of steps T, for any $\alpha \geq c_1 q^2 T$, Algorithm 1 is (ϵ, δ)-differentially private for any $\delta > 0$ iff. α is chosen by the following equation:*

$$\alpha \geq c_2 \frac{q\sqrt{Tlog(1/\delta)}}{\epsilon}$$

Moreover, considering a Gaussian noise, the equation above can be simplified to $\alpha = \frac{\sqrt{2log\frac{1.25}{\delta}}}{\epsilon}$. For further details, consult the Abadi et al. work [1]. Finally, to illustrate the aforementioned method's privacy guarantee assumes a training dataset with N instances, witch $GS = 0.01N$, $\alpha = 4$, $\delta = 10^{-5}$, and T= 10,000 will result in a $\epsilon \approx 1.26$.

2.8 Recurrence in dynamical system

Dynamical systems theory is used to describe the behavior of complex dynamical systems, such as complex meteorological cycles (e.g., El/Niño), the seasonal effects of changes in the earth's movements on its climate over time (named Milankovic cycles), electronic circuit behavior as well as systems that arise in medical and economic context [112]. These systems derive mostly from non-linear systems polluted by noise, defying scientists to provide meaningful analyzes of these systems. An important task in understanding dynamical systems is the discovery of the recurrences of system states, e.g., the periodicity of meteorological events. Recurrence Plot (RP) is a tool that can be used to investigate recurrences in a dynamical system [47].

To illustrate how the Recurrence Plot is employed, we provide a didactic example depicted in Figure 2.11. In Figure 2.11a, we have two dynamical systems, each one represented by the trajectories of two drivers in the same 2-dimensional space. It is worth mentioning that Recurrence Plot is usually used to map particles (instead of drivers) and considers a d-dimensional space, such as $d \gg 2$.

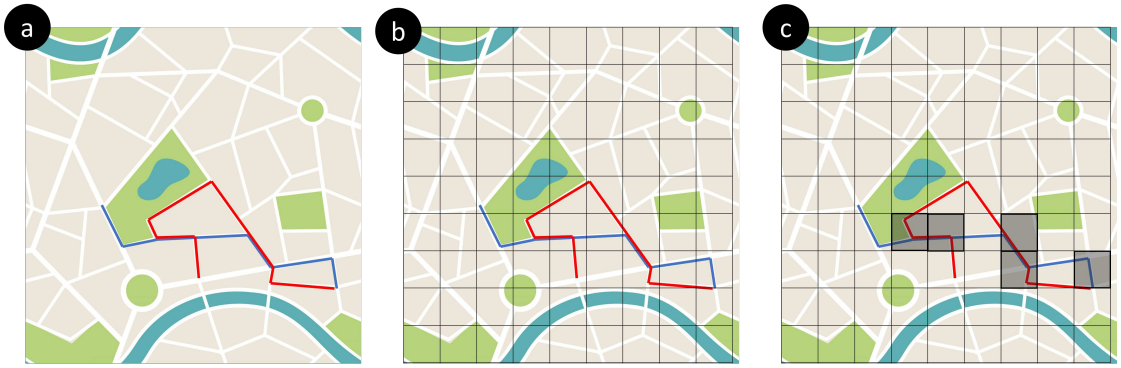


Figure 2.11: Graphical example of RP.

Figure 2.11b depicts the trajectories over the regions of the city, the map grid. To plot Figure 2.11c, we consider the drivers' trajectory, time, and speed to plot RP. The RP (especially the CRP detailed in the following) looks for those times when the state of the first system recurs to one of the other systems. In other words, Figure 2.11c depicts the regions the drivers were in simultaneously.

To provide an example of the actual usage of RP, in Figure 2.12a, we depicted the CRP of trajectories of two coupled Rössler systems (particles), represented by the black and grey lines corresponding to the first and second particle.

Notice that the corresponding CRP (Figure 2.12b) presents lines that are diagonally oriented. These lines represent segments on both trajectories, which run parallel for some time. The frequency and length of these lines are related to a certain similarity between the dynamics of both systems. Moreover, in Chapter 6 we will represent Bloom Filters (the encoded data of PPRL) as dynamical systems and use the similarity of the lines (patterns) to distinguish matching for non-matching Bloom Filters pairs. In the following, we presented the fundamentals of RP.

The concept behind the RP is to represent the time when states x_i recur in the system. Notice that the original purpose of the RP was to provide a tool that can provide insights into high-dimensional dynamical systems within the Physics context [47]. In summary, RP

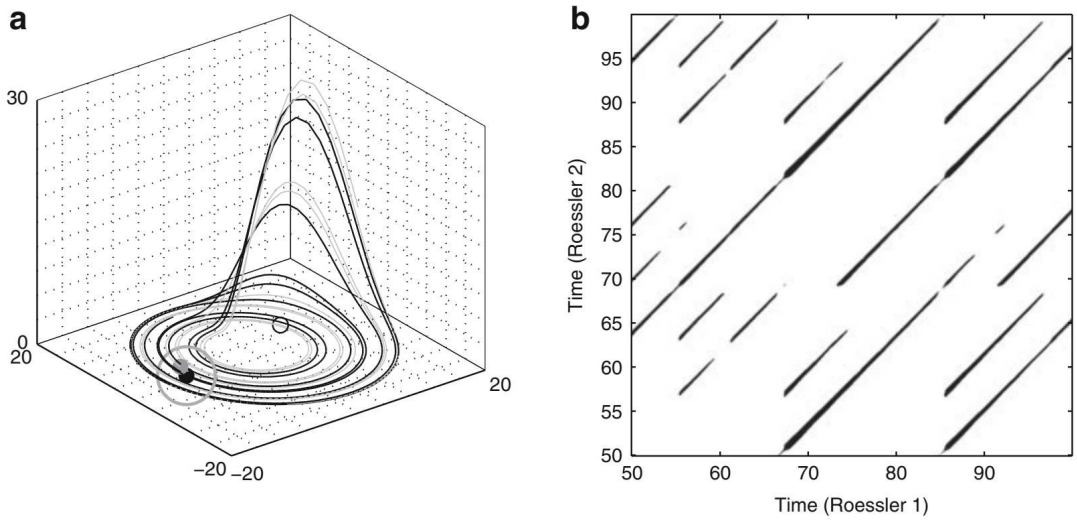


Figure 2.12: CRP of phase space trajectory of the Lorenz system - *source: Marwan et. al. [112]*

enables the analysis of a m-dimensional space through a two-dimensional representation of its recurrences [112]. Such recurrence of a state at time i at a different time j is pictured within a two-dimensional squared matrix R. Equation 2.6 formalizes the RP.

$$R_{i,j}^{m,\varepsilon} = \Theta(\varepsilon - ||x_i - x_j||), i, j = 1, \cdots, N \tag{2.6}$$

where ε is threshold distance, $\Theta(\cdot)$ the Heaviside function[3], $||\cdot||$ norm (L_1,L_2 or L_∞) and m represents the m-dimensional neighbourhood of size.

It is important to remark that RPs can operate on noisy data [164; 163; 150] and still reveal system characteristics. For instance, by analyzing the patterns of an RP, it is possible to determine whether a system is stationary or not, which parts of the system are deterministic, identify the periodic event patterns, and so on [112].

In order to study the correlations (also named synchronization) between two dynamical systems, several bivariate recurrence techniques were proposed. The Cross Recurrence Plot (CRP) is a bivariate recurrence technique that was proposed to investigate (simultaneously) two different dynamical system evolution, allowing the study of dependencies between two different systems [195; 111]. In other words, the CRP observes the states of both systems,

[3]The Heaviside function is commonly used in control theory and signal processing to represent a signal that switches on at a specified time [112]

and if the state recurs in both, the CRP will capture it. Suppose we have two dynamical systems, each one represented by x_i and y_i in the same d-dimensional space. We find the corresponding cross recurrence matrix by computing the pairwise mutual distances between the vectors of the two systems. Equation 2.7 represent the CRP calculation:

$$CR_{i,j}^{x,y}(\varepsilon) = \Theta(\varepsilon - ||x_i - y_j||), i, j = 1, \cdots, N \tag{2.7}$$

In Section 6.1.1, we use RP as a novel feature representation of PPRL encoded data to highlight patterns that could indicate whether or not an encoded record pair is a match or not.

2.9 Summary

In this chapter, we have provided the background on the concepts, principles, and techniques which lay the foundation for the contributions in this thesis. We covered aspects of anonymization techniques, PPRL, TL, and Deep Learning. We will use these concepts in the later chapters of this thesis.

Chapter 3

Related Work

A study of related work in the Privacy-Preserving Record Linkage aspects, tackled by each of our contributions, is reviewed in this chapter. As previously presented, we have three contributions to the PPRL process, one contribution in anonymization, comparison, and classification steps. Thus, due to the broad spectrum of our contributions, this chapter is divided into three sections.

We proposed an extension of the BF (called SBF) that intends to increase the privacy of the similarity computations of PPRL, detailed in Chapter 4. In order to provide a context to SBF, Section 3.1 presents improvements and limitations of the most used anonymization technique, the BF.

In order to provide an overview of the current progress in the privacy and adversary models in PPRL, in Section 3.2, we briefly summarize the recent literature related to Privacy-preserving protocols and auditable computation in the PPRL context. Furthermore, this section intends to elucidate the context of our second contribution (presented in Chapter 4) that intends to allow the usage of a novel adversary model (the covert adversary) by enabling the auditability of computation performed during PPRL execution.

Finally, to provide the context for the novel automatic classification step (detailed in Chapter 4) and the deep learning-based classifier (detailed in Chapters 5 and 6), we summarize in Section 3.3 techniques and strategies for automatic classification in the PPRL context.

3.1 Privacy aspects of Data Anonymization in PPRL

In this section, we briefly summarize recent literature related to BF that can improve the privacy guarantees or works that demonstrate how to reidentify anonymized data.

3.1.1 Bloom Filter Hardening Techniques

The Bloom Filter technique has been used in several real-world PPRL applications [79; 83; 89; 143; 144; 145; 171; 174; 178]. However, privacy concerns remain due to the susceptibility of BF to cryptanalysis attacks. In order to make BF more resilient to these attacks, recent researches propose hardening techniques to address privacy concerns.

The intuition behind most hardening techniques is to modify BF bit patterns to reduce the information required by attacks while still allowing accurate approximate similarity calculations. It is worthwhile to mention that the hardening technique applied on BF may lead to a reduction in linkage quality [30; 182]. In the following, we describe a set of recently proposed hardening techniques to reduce BF vulnerabilities.

The first group of hardening techniques modifies the BF bit pattern (i.e., the 0 and 1 inside the filter) irrespective of its actual content, such as the Balanced Bloom Filter (BBF) [152] and Xor-folding Bloom Filter (XBF) [154] techniques. The BBF relies on the choice of the parameters in order to achieve a uniform distribution of elements in the filters and, consequently, maximizes privacy [152]. To prevent bit pattern attacks, XBF folds (in half) the bit array and employs the XOR logical operator in each half of the filter in order to assemble a new filter [154].The second group of hardening techniques intends to systematically modify a BF according to the distributions of its 0 and 1 bits, such as random noise, Bloom-and-FlIp (BLIP), and Markov chaining techniques [153].

The random noises technique [151] is the most straightforward way to add noise by randomly flip bit positions (from 0 to 1 or from 1 to 0) in each BF. In turn, BLIP [153] applies the concept of randomized responses in order to flip the values of bits positions and, consequently, reduce the probability of entity re-identification during the PPRL process. Markov chaining technique [155] intends to prevent frequency-based attacks on BF by adding extra q-grams into a BF, based on their co-occurrences with the QID values.

3.1.2 Bloom Filter Attacks

The majority of the attack methods on the BF technique in the context of PPRL are based on the fact that plain-text values that frequently occurs in the dataset will generate a BF with the same frequency as the plain-text values. This attack is named as frequent bit pattern attack [30; 99; 98; 32; 182]. In this context, the work of Christen et al. [32; 182] shows that, to reidentify the entities, an adversary needs to generate the bit pattern using the possible values of the attributes. For instance, to reidentify the encoded name of a patient thread for a specific pathology, an attacker could generate the bit pattern attack using a list containing the most common names in a social network.

Recent works [143; 181] propose a novel cryptoanalysis attack, a pattern mining-based cryptoanalysis attack. The major advantage of this attack method over the frequent bit patterns attacks is that it neither requires frequent BF nor frequent plaintext values. For instance, the work of Vidanage et al. [181] applies maximal frequent itemset mining [37] (using a language model) on a BF database to identify sets of frequently co-occurring bit positions that correspond to encoded frequent sub-string values. Frequent pattern mining is a data mining technique commonly used for market basket analysis to, for example, identify which shopping items are commonly purchased together by customers [28]. In other words, the work of Vidanage et al. [181] uses the frequencies of individual q-grams as they occur across many values in the QIDs encoded into BF.

The last attack is the Graph-based cryptoanalysis attack. This attack uses the q-grams encoded into one single BF or the similarities calculated between BF pairs [30] to build a graph and employs the generated graph to reidentify the anonymized data.

3.1.3 Comparison Among BF Hardening Techniques

In the previous sections, we presented work attacks and improvements to BF. Table 3.1 displays a comparison between our contribution, the Splitting Bloom Filter (SBF), to the aforementioned BF hardening techniques.

Table 3.1: Comparison between BF hardening techniques and SBF.

	Compatible with other BF techniques	Iterative comparisons
Balanced Bloom Filter (BBF) [152]	no	no
Xor-Folding Bloom Filter (XBF) [154]	no	no
Bloom-and-FlIp (Blip) [153]	no	no
Markov chaining technique [155]	no	no
Splitting Bloom Filter (Chapter 4)	yes	yes

In order to compare SBF against related work, we highlight two aspects: i) whether the technique can be combined with other hardening techniques, or not; and ii) whether the entity pair similarity can be iteratively computed, or not. The last aspect, entity pair similarity iterative computation, can be used to perform the similarity calculation iteratively using small chunks (splits) of the original BF. In other words, SBF can be employed in a protocol that filters records pairs that probably will not be classified as a match. By filtering record pairs that will not be matched, SBF can reduce the amount of information, and minimize the chance of success in privacy attacks.

Notice that SBF can reduce the risk of reidentification for most of the attacks, be integrated with several BF hardening techniques, and enable the iterative computation of the similarity. It is worthwhile to remark that the related works does not reduce the reidentification risk toward all of families of attacks. Moreover, the compared work can not be combined and iteratively compute the similarity of entity' pairs.

3.2 Privacy-preserving protocols and auditable computation in PPRL

Within the Security and Cryptographic community, several Security Multiparty Computations (SMC) protocols have been proposed considering different adversary models. The SMC is a subfield of cryptography that aims to create methods for parties to jointly compute a function over their inputs while keeping those inputs private [104]. An example of

such protocol is the Garbled Circuits [12], a method to encrypt a computation that reveals only the computation output considering a malicious adversary model. The term ”circuit” is because garbled circuits work by modeling the encrypted computation, i.e., the similarity between two BFs, as a circuit and then performing cryptographic operations in each gate (AND, OR, NOT) in the circuit.

Considering the limitations of the HBC and Malicious security models, Aumann and Lindell [7] introduces the notion of covert adversaries. The author presents a cryptographic protocol that enables honest parties to detect the misbehavior of an adversary with high probability - this protocol applies Homomorphic Encryption (HE) to detect an adversary misbehave. However, it is important to remark that the Garbled Circuits [12] and the work Aumann and Lindell [7] were designed to perform one-time private computation. Therefore, these protocols are not adequate to the PPRL context that usually performs millions or billions of private computations during a typical linkage process.

In the PPRL context, Vatsalan and Christen's [170] presented a two-party protocol that eliminates the need for a third party by iteratively revealing selected bits in the BFs between two parties. However, this work is unable to audit the comparison performed during PPRL. Thus, we acknowledge as a research opportunity the proposition of a HBC security model in the context of PPRL.

In a subsequent work, Vatsalan and Christen [172] proposed a protocol to identify matching sets of entities (records) held by multiple parties that have a similarity above a certain threshold, under the semi-honest adversary model. The protocol divides and distributes the BF chunks amongst the parties. To calculate the entities' overall similarity, the parties employ the BF chunks in a SMC. However, their work considers an HBC model and does not explore neither the relationship between the individual similarity of the chunks nor the entities' overall similarity, resulting in low efficiency and poor linkage quality.

Hybrid PPRL protocols that combine differential privacy techniques with SMC techniques have been proposed to reduce the computational cost of PPRL [66; 70]. However, such protocols need to disclose all entities stored in their data sources amongst the parties compromising the privacy capabilities of the anonymization. To mitigate this issue, Rao et al. [146] propose a framework under a HBC security model that employs a trust-third party to coordinate the record matching between the parties. The parties send to the

trust-third party synopses (with Differential Privacy guarantees) of the data. Based on the received synopses, the trusty-third party matches the entities with a distance beyond a threshold specified by the parties. Notice that the works [66; 70; 146] consider an HBC and trusty-third party, which is a hindrance to the wide usage of PPRL in real-world applications [30; 177].

Finally, we present a technology that could eliminates the need for a trusty-third party, the Blockchain. The Blockchain technology has gained much attention in the data management community due to the trust achieved in not fully trusted computation environments. Tuan et al. [168] survey the state-of-the-art of Blockchain technologies. They propose a benchmarking framework for understanding the performance of private Blockchains considering data processing workloads. Nathan et al. [122] design and implement a decentralized replicated relational database with Blockchain properties, named as Blockchain Relational Database. The previously mentioned works confirm the need for privacy-preserving data processing in a Blockchain context.

Table 3.2 presents a comparison between our work and the related works mentioned above. The works are classified into four aspects: i) if the work can be employed in the PPRL context; ii) the adversary model; iii) if the work reduces the level of trust needed to execute the task; and iv) if the computations can be audited.

Table 3.2: Comparison between Privacy-preserving techniques/protocols and the ABEL.

	Is a PPRL solution?	Adversary model	Reduced level of trust	Auditability
Garbled Circuits [12]	no	Malicious	no	no
Aumann and Lindell [7]	no	Covert	no	yes
Nathan et al. [122]	no	Malicious	yes	yes
Vatsalan and Christen's [170; 172]	yes	HBC	no	no
Rao et al. [146]	yes	HBC	no	no
ABEL (Chapter 4)	yes	Covert	yes	yes

Notice that several works consider covert [7] and malicious [12] adversary models, and one of the presented works can be used to audit the computations [122]). However, the related works, compatible with the PPRL, solely consider the HBC adversary model. Furthermore, the works (compatible with the PPRL) are unable to audit or reduce the level of trust needed to execute the process. It is worthwhile to mention that, to the best of our knowl-

edge, besides the ABEL (presented in Chapter 4), there is no other technique or protocol that is able to audit the computation and consider a covert adversary model in the PPRL context.

3.3 Machine Learning usage in PPRL

Record Linkage (RL) researchers started to explore the usage of techniques originated in machine learning, data mining, artificial intelligence, information retrieval, and database research to improve the classification of linkage process [33; 41]. Many of these approaches are based on supervised learning techniques [2; 8; 26; 65; 67; 105] and assume that training data is available - i.e., record pairs with labels indicating whether they are a match or not. However, such datasets with labels are often not available in real-world applications or have to be prepared manually (considering a traditional RL context). Furthermore, in the PPRL context, it is not possible to manually label these datasets due to the privacy constraints imposed by the PPRL.

In the RL context, Deep Learning has been explored in several stages of RL, from the Blocking step [75] to the Classification step of the process [46; 91; 105]. Regarding the Deep Learning-based Classifier, it has been surveyed [121; 41], and categorized into three groups according to the nature of its input: i) structured data instances, ii) textual data instances, and iii) dirty data instances. According to Mudgal et al. [121], Deep Learning-based Classifiers do not outperform current classifiers (e.g., SVM and Logistic Classifiers) on RL tasks over structured data. However, Deep Learning classifiers can significantly outperform the existing classification techniques on RL tasks over textual and dirty data. [41; 121].

It is important to mention that, to our best knowledge, Deep Learning techniques have not been employed in the PPRL context. In Chapter 6, we present a Deep Learning-based Classifier for PPRL that, instead of employing standard similarity measures (e.g., Jaccard and Dice), leverages the bit patterns encoded to identify matching entities.

In Chapter 2, we presented the usage of ML techniques in the RL context. However, few of the works and techniques discussed are suitable to be used in the PPRL context due to the absence of labeled instances to train a classifier. Still, the work of Christen [25] proposes an unsupervised two-step approach to train a classifier in a traditional RL context, without labeled instances. In the first step, training examples are selected from the compared record

pairs. Records with the highest and lowest similarity values are labeled as duplicated and not duplicated, respectively. In the second step, the training examples are used to train a SVM classifier. In a subsequent work, Christen [27] uses a Nearest-Neighbour algorithm to iteratively add unclassified record pairs to the training sets.

The use of ML in the PPRL context is closely related to Privacy-Preserving Machine Learning (PPML). PPML techniques intend to protect the privacy of the data (training and testing data), model, and prediction [3]. Thus, our work and PPML share one goal in common, the protection of data privacy. The works [22; 16; 162] use encryption techniques, i.e., homomorphic encryption, to protect the privacy of data. The works of Rajkumar et al. [142] and Mivule et al. [118] uses the concept of differential privacy to provide privacy-preserving capabilities to their approaches. The work of Miyajima et al. [119] uses security protocols, a security multi-party computation (SMC) protocol, to train a classifier in a federated learning context. It is worthwhile to mention that all the presented PPML works [22; 16; 162; 142; 119] still need labeled data, which is unavailable within the PPRL context.

In a scenario where labeled data includes sensitive information, such as medical and biometric labels (e.g., labels holding information about the diseases), Kim et al. [85] propose a privacy-preserving domain adaptation technique to protect the privacy of the labeled instances from the source domain. The work leverages a pre-trained model from the source domain and updates the target model in a self-learning manner. It is worthwhile to remark that the work of Kim et al. has the primary goal of protecting the privacy of the labels in the source dataset and has unrestricted access to the information in the target domain. Therefore, the work [85] is not usable to the PPRL context.

Transfer Learning (TL) is another technique that has been explored in recent years in the RL context. The work of Thirumuruganathan et al. [165] considers a traditional RL scenario to propose the usage of TL. The authors propose TL usage along with Distributed Representation for Words (called word embeddings). Distributed Representation for Words [117; 139], recently introduced to deep learning, are learned from the data such that semantically related words have embeddings that are often close to each other. Typically, these approaches map each word in a dictionary into a high dimensional vector (e.g., 300 dimensions [165]) where the geometric relation between the vectors of two words – such as vector difference or cosine similarity – encodes a semantic relationship between them.

Several Transfer Learning techniques [133; 158] try to identify a common feature space that minimizes the distribution difference between the source and target domains. For instance, Sun et al. [158] minimizes domain shift by aligning the second-order statistics of source and target distributions without requiring any target labels. Notice that the work [158] does not need the target dataset to be labeled. However, the works [133; 158] are highly dependent on normally distributed data [88], which is not the case for most RL applications.

Kirielle et al. [88] propose a TL method for RL over structured data. The work assumes homogeneous domains with the same feature space (same attribute types and similarity functions). In other words, it employs the similarities of the entities' attributes (e.g., names and addresses) from one domain to train a classifier to be employed in another domain that shares the exact attributes. It is worthwhile to mention that the comparison in PPRL is usually performed over the complete record, harming the linkage quality in a PPRL context.

Negahban et al. [124] use TL in order to scale up RL to multiple data sources. The approach's idea is to adaptively share the structure learned about one scoring problem with all other scoring problems sharing a common data source. It is worthwhile to mention that both works (Thirumuruganathan et al. [165] and Negahban et al. [124]) are related to our contribution presented in Chapter 5. However, to the best of our knowledge, in Chapter 5, we present the first approach that proposes TL usage in a PPRL context.

Several linkage processes employ automatic (ML-based) techniques to identify matching entities. However, few of them are compatible with PPRL. Table 3.3 presents a comparison between ML-based classifiers and our approaches, AT-UC and DLC.

Table 3.3: Machine Learning usage in PPRL.

	Privacy-Guarantees	Classifier	Labeled Instances
Christen [25; 27]	no	unsupervised	no
PPML [22; 16; 162; 142; 119]	yes	supervised/semisupervised	yes
Thirumuruganathan et al. [165]	no	unsupervised	no
Sun et al. [158]	no	semisupervised	no
Kirielle et al. [88]	no	semisupervised	no
AT-UC (Chapter 5)	yes	unsupervised	no
DLC (Chapter 6)	yes	semisupervised	no

Notice that the majority of works do not provide privacy-preserving guarantees. It is worthwhile to mention that PPML, which provides privacy guarantees, requires labeled data. As mentioned early in this section, labeled data is usually unavailable in the PPRL context. In turn, the AT-UC and DLC can provide ML-based classifiers without labeled data to the PPRL process. Furthermore, DLC refers to a novel method to compare and classify entity pairs that do not rely on standard similarity measures, addressing the bias introduced by the standard similarity measures [95; 105].

Chapter 4

PPRL Comparison Step Auditability

According to recent PPRL surveys [176; 34; 136; 171; 30], most of the state-of-the-art PPRL techniques assume an Honest But Curious (HBC) security model during the matching process. This model considers that the parties will follow a pre-agreed protocol, but will attempt to learn all possible information from legitimately received messages [171]. In contrast to the HBC model, we have the malicious security model, which assumes that the parties will not follow the pre-agreed protocol by refusing to participate, performing arbitrary computation in the input data, or aborting the protocol at any time [34].

The HBC model requires the parties to fully trust each other, which is not realistic for real-world applications [171; 176], and the malicious model is computationally expensive, due to the anonymization techniques (i.e., homomorphic encryption) and communication cost [176]. The need for new security models that enable an auditability of the similarity computations (i.e., models that lie in between the HBC and the malicious models) is reported as an open problem [34; 136; 171; 176; 181].

In this context, we introduce a novel security protocol that enables the auditability of the computations, also named as covert adversaries model [7], performed during a PPRL process. In other words, the proposed protocol allows the parties to detect, with high probability, the misbehave of a malicious party during the entities' similarity computation. To implement the protocol, we employ a decentralized environment where untrusted (or semi-trusted) parties perform the computations required by the PPRL.

As the decentralized computing environment, we use (public and private) Blockchain networks in order to provide auditability to PPRL. The Blockchain technology aims to provide

shared data access and computation for parties that do not trust each other [50]. The computations and the data sharing are possible due to an immutable cryptographically append-only log, which is replicated and managed in a decentralized environment composed by untrusted parties [122].

Blockchain platforms are being used by applications of different domains [39; 40; 50; 122], including government, healthcare, and IoT. In such applications, the Blockchain is treated as a shared database or data processing platform which is employed when the participants do not trust each other. For this reason, we use the Blockchain (Smart Contract) as a Semi-Trusted Third Party (STTP) in PPRL.

However, the Blockchain does not provide a mechanism to preserve the privacy of the entities during the PPRL process. In fact, Blockchain reduces the privacy of the PPRL by replicating the entire data amongst the untrusted parties. To overcome this limitation, we also propose an improvement for the most prevalent anonymization technique used in PPRL applications: Bloom Filter [152]. Such improvement, named Splitting Bloom Filter (SBF), offers a lower risk of privacy leakage and, additionally, reduces the possibility of collusion between malicious parties. Thus in the following sections, we present;

- The Splitting Bloom Filter (SBF) that increases the privacy guarantees of the entity comparisons performed in PPRL, by reducing the amount of information shared amongst the PPRL parties;
- A novel privacy-preserving protocol that considers a covert security model, which lies between the HBC and Malicious models;
- A Proof-of-Concept implementation of our privacy-preserving protocol using the Blockchain technology;
- Theoretical and empirical evaluations using real-world data to assess the efficacy, efficiency, and privacy capabilities of our contributions.

4.1 Problem Formalization

In order to improve the privacy capabilities of the PPRL problem, we assume that multiple parties and an STTP engage in a PPRL process that considers an accountable security

model (covert adversaries).

Problem Statement 1 (Decentralized execution of the decision model)**.** *Given multiple participants ($|P| \geq 2$) with their respective anonymized data sources ($\hat{D}$), and a Semi-Trusted Third-Party (STTP) implemented as a Blockchain Smart Contract (Transaction), how to distribute the decision model execution among the parties and the STTP? Eq. (4.1) formalizes the decentralized execution model ($\hat{\Xi}$).*

$$\hat{\Xi} = (\Xi_1(\hat{e}_1, \cdots, \hat{e}_p) \cup \cdots \cup \Xi_p(\hat{e}_1, \cdots, \hat{e}_p)) \cup \Xi_{sttp}(\hat{e}_1, \cdots, \hat{e}_p) \tag{4.1}$$

where Ξ_j and Ξ_{sttp} represent the output of the decision model generated by a party p and STTP, respectively.

Since we consider an accountable security model, our approach needs to audit all computations performed by the decision model (Ξ), regardless of who executed them (the parties or STTP).

Problem Statement 2 (Auditability of the decision model)**.** *Given the decision model execution Ξ, with p participants, the audit function (audit($\hat{\Xi}$)) should detect when the parties or the STTP tries to deviate from the protocol by sending (or computing) wrong similarity values of the entities. This misbehavior is detected by comparing the decision model output of each participant. Eq. 4.2 formalizes the audit function.*

$$audit(\hat{\Xi}) = \begin{cases} FP \iff \exists(\Xi_p, \Xi_{p+1}, \Xi_{sttp}) \in \hat{\Xi}\,,\ \Xi_p \approx \Xi_{p+1} \approx \Xi_{sttp} \\ MB \iff \exists(\Xi_p, \Xi_{p+1}, \Xi_{sttp}) \in \hat{\Xi}\,,\ \Xi_p \not\approx \Xi_{p+1} \not\approx \Xi_{sttp} \end{cases} \tag{4.2}$$

where MB means that a party has misbehaved (i.e., did not follow the protocol) and FP means that all parties followed the protocol.

Knowing that a Blockchain stores all information in an unalterable and public readable database (Ψ), we also have to address the following problem.

Problem Statement 3 (Privacy-preserving guarantees of the decision model)**.** *Given an anonymized entity ($\hat{e} \in \langle \hat{e}_1, \cdots, \hat{e}_p \rangle$), how to maximize the privacy capability (i.e., avoid the reidentification of entities) during the excution of the decision model in a Blockchain environment?*

Given the aforementioned problems, the goals of this chapter can be summarized as:

$$\begin{aligned} \textbf{Given:} \quad & \{D_1, \cdots, D_{|P|}\}, \Xi, \Psi \\ \textbf{Find:} \quad & M \in \Xi(D_1, \cdots, D_{|P|}) \\ \textbf{Uncover:} \quad & The\ Misbehave \\ \textbf{Preserving the privacy of:} \quad & \hat{e} \in D_i \forall i \in \{1 \cdots |P|\} \end{aligned}$$

A summary of the notation employed in this work is presented in Appendix A.1.

4.2 Splitting Bloom Filter

As presented in Chapters 2 and 3, Bloom Filters may fail to preserve the privacy of the data in two cases. The first one occurs when an adversary has a list of possible values assumed by the entities [32; 116; 125; 181]. The second case happens when an adversary is able to access several anonymized entities and execute a pattern mining attack [32; 181]. Both vulnerabilities may endanger the privacy of the entities employed in PPRL. Considering that the entities involved in PPRL belong to a unique domain (e.g., patients with the same pathology), the participants may have a list of possible values assumed by the entities. This facilitates cryptoanalysis attacks as reported in [125; 151]. Also, in order to perform the comparison step of PPRL, the parties need to share their entire anonymized entities, which favors a sophisticated cryptoanalysis attack (e.g., pattern mining attack) [32; 181].

In order to mitigate the majority of the attacks to the BF and enable auditable privacy-preserving protocols to PPRL, we explore a novel aspect of the BF, named as Splitting Bloom Filter (SBF). SBF's basic idea is to perform iterative similarity computation using only a small piece (split) of the original BF (instead of the entire BF) to reduce the amount of information shared during the comparison step of PPRL. In other words, the SBF intends to split the original BF in s splits $[\phi^0, \cdots, \phi^{s-1}]$, where each split will have a fraction of the original BF length. The SBF is formalized as follows.

Definition 4 (Splitting Bloom Filter). *The SBF approach divides the l-bits of a BF into s splits ϕ. Considering that the $BF(e) = \hat{e}$, we can express the anonymized entity as*

$\hat{e} = [b_0, \cdots, b_l]$. *The SBF can be formalized as:*

$SBF(\hat{e}, s) = [\phi^0, \cdots, \phi^{s-1}]$, *such that* $\phi^i = [b_j, \cdots, b_{j+(\frac{l}{s}-1)}]$ *and* $j = i \times \frac{l}{s}, \forall(i)|0 \leq i \leq s-1$.

Based on the SBF, the similarity computation between two different entities, e.g. $Jaccard(\hat{e}_1, \hat{e}_2)$, can be executed independently by the parties. The similarity computation[1] using the Jaccard function is detailed in the Eq. (4.3):

$$Jaccard\_SBF(\hat{e}_1, \hat{e}_2, s) = \frac{1}{s} \sum_{i=0}^{s} \frac{|\phi_1^i \bigcap \phi_2^i|}{|\phi_1^i \bigcup \phi_2^i|} \tag{4.3}$$

, where s is the number of splits. Also, ϕ_1^i *and* ϕ_2^i *are the i-th splits from* $SBF(\hat{e}_1, s)$ *and* $SBF(\hat{e}_2, s)$*, respectively.*

The distributed similarity computation characteristic of the BF has been exploited in previous works [172; 175] to enable secure multi-party computation (SMC) in the PPRL context. However, in this work, we assume that the SBF splits (ϕ) can be employed to verify the computation performed by a PPRL participant, reducing the information available to execute attacks by malicious parties. Moreover, in Section 4.4, we demonstrate how SBF allows the identification of malicious participants that eventually did not follow the agreed protocol and consequently enables an auditable security model in the PPRL context. Finally, the splits also reduce the amount of shared information during the linkage process, maximizing privacy capabilities of PPRL.

In order to estimate the mean similarity between the BF splits, we use the following assumptions: i) the SBF does not change the probability of the elements being flipped to '1'; ii) the SBF does not alter the false positive rate; and iii) an ideal BF ($p = 0.5$) has a near-uniform distribution of 1s along the filter [17]. Considering the previous assumptions, we propose the following statement to provide auditability of the SBF comparison.

Acknowledging that the splits of SBF are different, we also assume that the similarity of the splits is slightly different from the complete BF similarity, i.e., $Jaccard(e_a^\tau, e_b^\tau) \neq$

[1] Other similarity functions available for regular BF can also be used in the SBF approach, e.g. token distance.

$Jaccard(\phi_a^i, \phi_b^i)$, such that $\phi_a^i \in SBF(e_a^\tau, s)$ and $\phi_b^i \in SBF(e_b^\tau, s)$. The rationale of this assumption is justified in Appendix A.3. Therefore, the similarity comparison between any pair of splits is related to the similarity between the BFs plus a certain $error$ (ϵ) and can be represented by Eq. (4.4).

$$Jaccard(e_a^\tau, e_b^\tau) = Jaccard(\phi_a^i, \phi_b^i) + \epsilon \tag{4.4}$$

, such that $0 \leq \epsilon \leq 1$.

In order to illustrate the similarity difference between the split and complete BF, Figure 4.1 depicts this difference using the words ANA and ANE. In Figure 4.1(a), the similarity of the words produced by a regular BF is 0.66. In Figure 4.1(b), we apply SBF to divide the original BF into two splits. Notice that, for each split, we introduce an error (ϵ) of 0.14 and 0.16 during the comparison (Eq. 4.4), respectively.

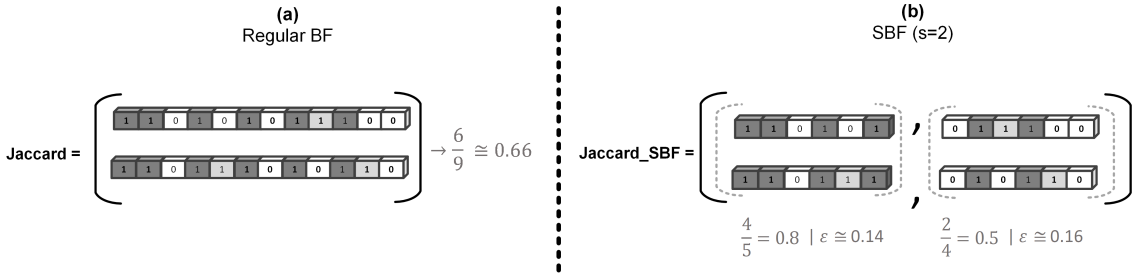


Figure 4.1: An example of SBF using two splits, (a) illustrates the similarity calculation over anonymized values of the words ANA and ANE (Fig. 2.2). In (b), the BFs are split in two. The similarities between the splits are presented as well as the resulting $error$.

It is worthwhile to mention that the relation between the number of splits and the error is examined through experiments in Section 4.3. The experiments demonstrate that the number of splits is important to predict the $error$. Moreover, we prove that the error is crucial to determine the privacy and the quality of the PPRL linkage.

4.3 SBF Evaluation

To assess the accuracy of the similarity computation and investigate the privacy capabilities provided by SBF, we need to answer the following Research Questions (RQs):

RQ. 1 Does the similarity between two entities can be accurately calculated by SBF, Eq. (4.3)?

RQ. 2 What is the influence of split length (number of bits) over the similarity calculated using the SBF (Eq. 4.4)?

RQ. 3 Does SBF improve the privacy capabilities of PPRL processes?

To answer these questions, we employed seven pairs of real-world data sources from the following domains: bikes, beers, books, electronic goods, movies, music, and restaurants. Each pair of data sources presents different characteristics, such as the number, values, length, and the level of dirtiness of attributes. A summary of the data sources characteristics is shown in Table 4.1, and details about the data sources are presented in the Appendix A.5.

Table 4.1: SBF Data sources and anonymization parameters details.

Domain	—A—	—B—	attributes	l	k
bike	2,689	3,721	3	920	7
beer	4,330	2,999	3	920	7
books	1190	1,106	4	2,456	8
eletronics	2,769	4,007	4	9,816	10
movies	7,029	6,063	4	7,360	9
music	20,062	3,805	7	9,816	10
restaurants	4,655	8,400	5	1,224	8

Table 4.1 also presents the anonymization parameters employed in the BFs: l (number of bits of the filter) and k (number of hash functions). These parameter values were chosen to achieve an optimal privacy ($p = 0.5$) of the filters [153].

4.3.1 Accuracy of the Similarity Computation

To evaluate the accuracy of the similarity calculation produced by SBF, we employ three different BF strategies [152; 153; 154]: i) Balanced Bloom Filter (BBF); ii) Xor-Folding Bloom Filter (XBF); and iii) Bloom-and-FlIp (BLIP). We vary the BF implementation to ensure that SBF provides accurate results using a regular implementation (BBF) or privacy-enhancing implementations such as BLIP and XBF.

BBF relies on the choice of the parameters in order to achieve a uniform distribution of elements in the filters and, consequently, maximizes privacy [152]. To prevent bit pattern

attacks, XBF folds (in half) the bit array, and employs the XOR logical operator in each half of the filter in order to assemble a new filter [154]. In turn, BLIP [153] applies the concept of randomized responses in order to flip the values of random bit positions and, consequently, reduce the probability of entity re-identification during the PPRL.

The $error$ produced by SBF (tackled by RQ 1 and RQ 2) is illustrated in Figure 4.2. This $error$ was calculated as the difference between the modulus of the similarity produced by the regular BF (Jaccard) and SBF (Eqs. 4.3 and 4.4). The horizontal and vertical axis show the number of splits and the error of the similarity computation, respectively. The dashed lines indicate the results achieved by SBF using only one split (Eq. 4.4), while the continuous line shows the use of all SBF splits in the similarity computation (Eq. 4.3).

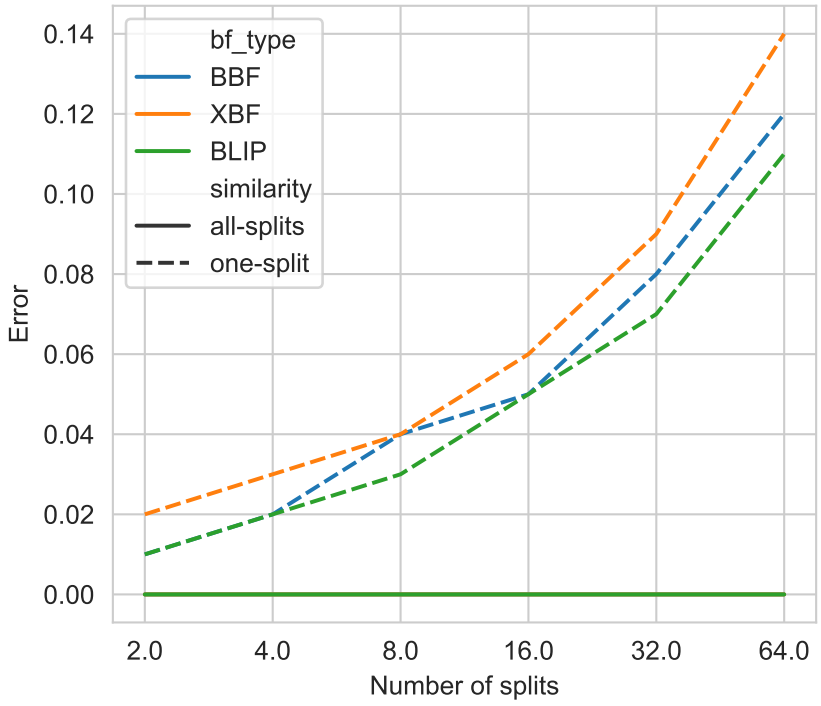


Figure 4.2: Error obtained using different BF hardening strategies.

As illustrated in Figure 4.2, regardless of the BF anonymization strategy, the similarity calculation produced by Eq. 4.3 presents the same similarity values as the traditional BF. In other words, SBF is able to calculate the similarity of two entities accurately when all splits are employed, answering RQ. 1.

Figure 4.2 also exposes that the error (dashed lines) of utilizing a single split for calculating the similarity of two entities increases as the number of splits grows. In

other words, the error grows as we increase the number of splits, regardless of the BF hardening strategies employed. The SBF error disagreements of the BF hardening strategies are not statistically different from the mean error of the strategies examined using the Wilcox test with 95% confidence.

According to Figure 4.2, it is possible to observe by the dotted lines a relation between the $error$ (presented in Eq. 4.4) and the number of splits. In order to confirm this relation, we applied a Statistical correlation test (Pearson R test) over the calculated similarities. We refuted the null hypothesis with a correlation factor of 0.82 and *p-value*< 0.005. In other words, we confirm that there is a correlation between the number of splits and the *error* presented in Eq. 4.4.

For a better understanding of the relation between the number of splits and the $error$, we plotted Figure 4.3 considering the BBF strategy. The horizontal axis illustrates the length of the split divided by the original filter length (l/s), — e.g., if a BF originally has 1,024 bits and we split this BF into four parts, each split will represent 25% of the original split. The vertical axis describes the error of comparing two entities using only one split. The colored lines exhibit the mean error produced by the use of the individual splits for each data source.

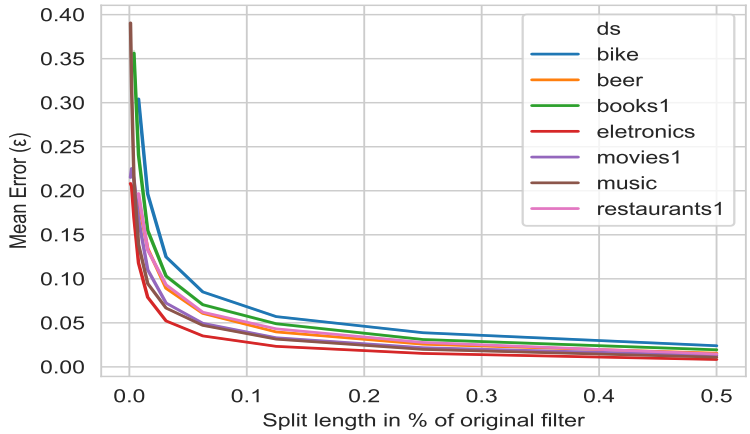


Figure 4.3: Error associated to split length (percentage) by data source, employing the BBF.

Figure 4.3 shows an increase of the error while we decrease the split length. The raise of the error can be explained by the fact that, in smaller splits, every bit that diverges in Eq. 4.4 has a higher impact over the calculated similarity.

As shown in Figure 4.3, a logarithmic function describes the error. To confirm the previous statement, we build a logarithmic regression model ($f(x) = a * ln(b * x) + c$) and use

the result of the experiments to train and test (with cross-validation) this model. Figure 4.4 illustrates the fitting of the model for each data source.

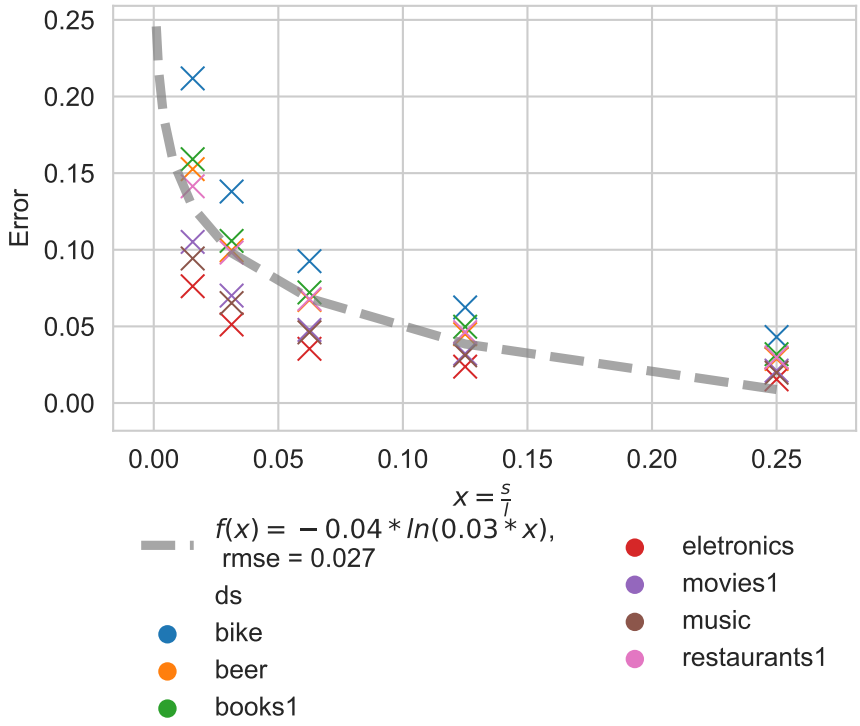


Figure 4.4: Accuracy of the estimated error in SBF.

Figure 4.4 presents the same axis of Figure 4.3. The line represents the model output, and red x mark represents the model error. As illustrated in Figure 4.4 and confirmed by the Root Mean Square Error (RMSE) result (rmse=0.027), we demonstrate that a logarithmic function fits well with the error growth in our experiments. This is relevant because i) we provide an estimation of the impact of the split length in Eq. 4.4 error (RQ. 2), and ii) the fact that a logarithmic function can describe the error is relevant to some privacy aspects, detailed in Section 4.3.2.

4.3.2 Privacy

In order to investigate the privacy capabilities provided by SBF (RQ. 3), we assume that an attacker needs to have access to a significant amount of information to execute a successful privacy attack [32; 181]. Such attacks are more likely to succeed if the n-grams (n) are hashed into the shared splits in SBF. It is possible to execute frequency-based and crypto-

analysis attack successfully in one split. However, if we consider the usage of a small split in SBF (e.g., split with 16 or 32 bits), such attacks need to possess more information than the regular BF to re-identify a single split's information based on few bits of information. In other words, we believe that the re-identification attack is more prone to succeed when the k parts of the n-gram are stored in the shared split. Thus, we estimate the SBF's privacy by the probability of n-grams being mapped in the SBF split.

We can estimate the probability of an individual position of the filter (b_j) be used to map a particular n-gram by $P(b_j) = 1 - e^{\left(-\frac{k \times n}{l}\right)}$ [17]. Consider that SBF divides the BF in s splits, where each split has $\frac{l}{s}$ bits, and each n-gram of the plain data is mapped to k positions by different hash functions. Equation 4.5 estimates the likelihood of an n-gram (ng) being mapped into a split $P(ng \in \phi)$ by calculating the probability of k bits of the splits be used to represent the n-gram.

$$P(ng \in \phi) = P(b_1 \in \phi) \cap \cdots \cap P(b_k \in \phi) \tag{4.5}$$

Knowing that the BF hash function provides a uniform random distribution of the n-grams inside the filter [152; 17], we can assume that the $P(b_k \in \phi) = P(b_j)$. Thus, it is possible to modify Eq. 4.5 as follows.

$$P(ng \in \phi) = P(b_1 \in \phi) \cap \cdots \cap P(b_k \in \phi)$$

considering that, $P(b_k \in \phi) = P(b_j),\ and\ P(b_j) = P(b_{j+1})$

$$P(ng \in \phi) = P(b_1) \cap \cdots \cap P(b_k)$$

$$\Rightarrow P(ng \in \phi) = P(b_1) \times \cdots \times P(b_k)$$

$$\Rightarrow P(ng \in \phi) = (P(b_k))^k$$

$$\Rightarrow P(ng \in \phi) = \left(1 - e^{-\frac{k \times n}{l}}\right)^k$$

considering that we are calculating the probability over the split ϕ*, l represents the split length, such that,* $l = \dfrac{l}{s}$

$$\Rightarrow P(ng \in \phi) = \left(1 - e^{-\frac{s \times k \times n}{l}}\right)^k$$

It is worthwhile to mention that the hash functions used (k), number of n-grams (n), and the length of the regular Bloom Filter (l) provide a trade-off between the linkage quality and

privacy [17]. This trade-off is given by the false-positive rate ($fpr = 1 - e^{-\frac{k \times n}{l}}$) of the BF. Therefore, the higher the value for fpr, the higher the privacy and the lower the quality of linkage, because the number of q-grams mapped to a single bit (and therefore the number of resulting collisions) rises, which leads to lower linkage quality but makes it more challenging for an adversary to learn the q-gram combinations in BF [30]. Thus, a value used to provide a suitable comprising between quality and privacy is $fpr = .5$ [30; 152; 176].

In order to investigate the influence of the SBF parameters over the privacy guarantees, we plotted the graph depicted in Figure 4.5 varying the false-positive rate and the split length from 50% to less than 0.01% of the original BF ($0 < fpr < 1$). The original BF's false-positive rate is depicted by the lines, while the BF length (in percentual of the original length) is illustrated in the horizontal axis. The $P(ng \in \phi)$ is depicted in the vertical axis.

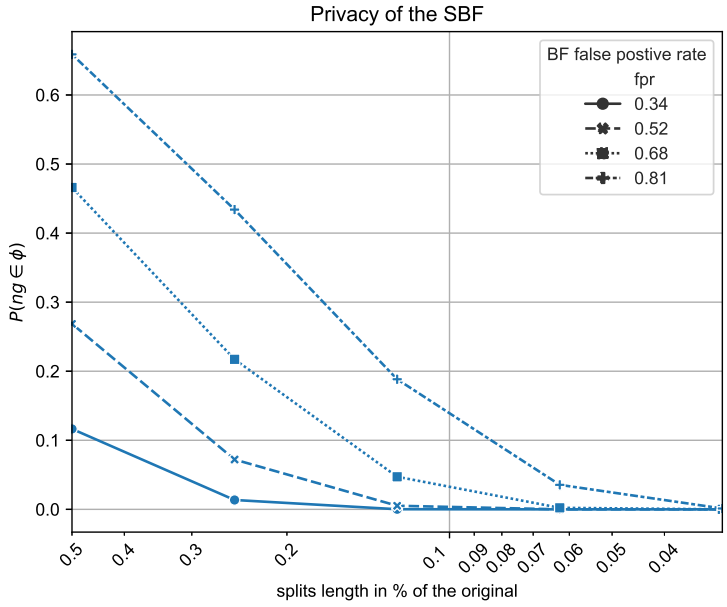


Figure 4.5: Probability of an n-gram being stored in a split considering different BF parameters.

Figure 4.5 shows the influence of the BF false-positive rate and the split configuration over the SBF privacy. Before explaining the result, we have to remember that the false-positive rate is proportional to the number of '1' in the BF. Lower f implies in a low populated filter while higher fpr results in a filter with more '1', due to the relation between the number of k and the l.

Another remark about the splits is that each split can store $\frac{l}{sk}$ n-grams. Thus, we can

intuitively state that smaller splits increase SBF's privacy guarantees, illustrated in Figure 4.5 and can be calculated by Equation 4.2.

To investigate if these properties are enough to keep the entities' privacy during a real-world PPRL process, we employ a state-of-the-art Bloom Filter cryptoanalysis attack [181], presented in the related work section, against the NCVR dataset [2] with 224 thousand and 6 millions records. In this experiment, we employ the most used configuration of the BF [152], when $fpr = .5$, over five attributes (firstname, lastname, age, gender, and city) of the NCVR dataset. The attacks were executed over the SBF split, varying l/s from 0.01 to 0.25.

It is worth mentioning that the attacks were unable to re-identify any entities in the experimented data source. This result can be explained by the fact that most BF attacks need to have access to a considerable amount of information to re-identify the anonymized entities [32; 34; 173; 176; 181]. Therefore, since we share only a fraction of the original data, making it hard for state-of-the-art cryptoanalysis attacks to re-identify the entities using their strategies (e.g., pattern mining and dictionary attacks).

4.4 Auditable Blockchain-based PPRL

In this section, we present the components of the ABEL approach. First, we present a generic protocol (3PAC protocol) that generalizes and formalizes the concepts employed to implement the ABEL approach. Finally, we present an extensive evaluation and a summary of the ABEL capabilities.

4.4.1 An Auditable Protocol for PPRL

In this section, we present a novel privacy-preserving linkage protocol, which considers covert adversaries to audit the comparison and classification steps of PPRL. This protocol uses SBF to reduce the amount of information shared during the PPRL execution. Moreover, this protocol uses the decentralized SBF characteristic (presented in Eq. 4.3) to provide audibility in the similarity computations performed by the PPRL parties. The protocol also considers a Semi-Trusted Third Party (STTP), where the parties may audit all their actions.

[2] Available at the authors website: https://dmm.anu.edu.au/pprlattack/

Definition 5. *(**Semi-trusted third party**) The STTP is a participant that is partially trusted to perform a specific computation. Although STTP may misbehave, it will not conspire with none of the parties. In other words, it can deviate from the protocol by:*

1. *Leaking the information sent by the participants;*
2. *Executing wrong computations over the information shared by the participants.*

The intuition behind the protocol is to perform the comparison step in two stages in order to improve the privacy-preserving capabilities of the protocol. In the first stage, STTP performs the similarity computation according to Eq. (4.4) using only one split of the original BF. In this stage, the single split's similarity is used to eliminate entity pairs that present similarity values that differ from the designated threshold, β-threshold. In summary, the first stage selects entities that present a high probability of being similar using a small fraction of the parties' anonymized data source. As a result, a list of entities is disclosed to the other participants.

In the second stage of the protocol, the parties perform similarity computations (interactively) in the remaining splits of the entities disclosed by STTP. To audit the protocol execution, the parties may compare the similarity calculated in the second stage with the similarity exposed by STTP. If the difference between these similarities is higher than $error$, it means that one party (or the STTP) did not follow the protocol. The protocol is detailed in Protocol 1.

Protocol 1. 3-Party Comparison and Classification Auditable Protocol (3PAC)

Setup

1. The parties agree on the BF and SBF parameters; the number of splits (s), the splits that will be employed in the first stage (ϕ_i), the $error$ and the thresholds α and β;
2. The parties anonymize the entities and randomly generate a unique ID for each entity;
3. The parties send the SBF splits to STTP $send(\phi_i, STTP) \mid \forall SBF(e^{\tau}, s) \in D_p^{\tau}$.

Stage 01

4. STTP computes the similarities between the splits;
5. STTP reveals (to all parties) a list (ζ) comprising the IDs and the similarity values between the entities.

Stage 02

6. For each split (s) of the entities stored in ζ, the parties alternately exchange among themselves the splits, one at time. At the end, each participant receives φ splits, where $|\varphi| = \frac{s-1}{|P|}$;
   (a) The splits sent by the parties are employed as input to calculate the similarity of the entities using Eq. (4.3);
      i. If the difference between this similarity and the value calculated by STTP is greater than the $error$, the participant detects the misbehave and aborts the protocol execution.
   (b) The parties exchange the splits similarity calculated in the previous step in order to update the entities' overall similarity;
      i. The parties check if the difference between exchanged similarity and the value stored in ζ. If this difference if higher than $error$, the parties detect a misbehave and abort the protocol execution;
      ii. At the end of this step, the parties update the entities' similarity value in ζ.
7. Finally, the parties select entities that have a similarity value higher than α-threshold.

In the Setup, the parties agree among themselves on the anonymization parameters (step 1), execute the data anonymization (step 2) and send the splits to STTP (step 3). Notice that STTP is not aware of the data anonymization parameters or which split was chosen to be used. In other words, STTP receives only a small part of the original BF, which makes it difficult to execute cryptoanalysis attacks [181; 32]. Two important parameters of our protocol are chosen during Setup, β, and α thresholds. These thresholds are employed in Stage 1 and 2, where β is employed in both stages, and α is employed in the final step of Stage 2. These thresholds must consider the SBF error and should be calculated as $\beta = \alpha - error$.

In Stage 1, STTP computes the similarity of all pairs of entities received during the Setup, using only one split to represent each entity. At the end of the stage, STTP publishes a list (ζ) of all entities pairs with a similarity value greater than β. The β threshold must be chosen carefully to reduce the amount of information shared in the subsequent stages. In other words, by choosing a lower value of β, the number of entities forwarded to Stage 2 is increased, which enhances the probability of successful cryptoanalysis attacks and the computation cost of the protocol. The reduction of shared information and computational costs are detailed in Section 4.5.2.

In Stage 2, the parties distribute, among each other, the remaining splits to perform the similarity computation of the entities disclosed in ζ. Each party is responsible for performing the similarity computation in φ splits, where $|\varphi| = \frac{s-1}{|P|}$, using Eq. (4.3) for each pair of entities (step 6a). In other words, the splits of the entities (such that $\hat{e} \in \zeta$) are distributed among p participants.

Afterwards, in step 6a, the similarity computation is performed over the splits (φ) of the entities. In step 6ai, each party compares the similarity value (calculated in step 6a) to the value stored in ζ (step 4). Then, if the similarity value calculated in step 6a is different from the similarity present in ζ plus $error$, the party assumes that a malicious party did not follow the agreed protocol and aborts the protocol execution.

The entities' overall similarity (Eq. 4.3) is interactively calculated by the parties in step 6b. The parties check the difference between the exchanged similarity value against the calculated similarity, and the value stored in ζ. If the difference is higher than $error$, a party aborts the protocol execution (step 6bi) because a malicious party seems to have sent bogus similarity values. Finally, the parties learn which of their entities exist in all datasets (step 7).

Notice that our protocol is intended to execute the Comparison and Classification steps of PPRL in a covert adversary model. In this sense, the SBF splits provide auditability in the comparison step of PPRL. It is worth mentioning that: i) no participant has access to the complete anonymized representation of an entity; and ii) the similarity computation is performed using entities (splits) that have a high probability of being similar. In other words, the protocol also reduces the amount of information shared by participants, decreasing the probability of a successful cryptoanalysis attack.

4.4.2 Auditable Blockchain-based Privacy-preserving Record Linkage

In this section, we describe the **A**uditable **B**lockchain-based Privacy-preserving R**e**cord **L**inkage (ABEL), a Proof-of-Concept implementation of the 3PAC protocol presented in Section 4.4.1.

Different from the traditional HBC protocols, which consider an external (neutral) party in which all participants trust [44; 128; 171], ABEL considers a semi-trusted third party (STTP), which parties do not fully trust. Because they do not rely on STTP, the parties need to audit STTP computations. In order to enable the auditability of the computations performed by the STTP, we implement it as a Smart Contract hosted in a Blockchain. In other words, the use of a Smart Contract transforms the STTP into a piece of code that executes in a Blockchain environment.

By considering the STTP as a Smart Contract, we offer three important characteristics inherited from the Blockchain. The first is the tamper-evident characteristic of the STTP. In other words, once that the STTP Smart Contract is deployed to the Blockchain, it can not be modified [189], in other terms, a malicious party unlikely will change the Smart Contract code.

The second characteristic is the decentralized execution model. This characteristic enables the STTP to be executed in different machines eliminating the need for a centralized computing environment. In other words, the PPRL parties could create a private network with its own machines, and the computations will be executed in all machines of the network, without a central server.

The final characteristic is the auditability (transparency) of the Blockchain. Once that STTP is stored in the Blockchain, the STTP code, inputs, and outputs can be read (audited)

by the Blockchain members. This advantage provides the ABEL auditability capability while posing as a privacy disadvantage by making available private data to every Blockchain member. As presented in Chapter 2, every Smart Contract invocation (transaction) is stored in the Blockchain with all inputs and outputs. In other words, by default, all transaction attributes (input and outputs) are readable by the Blockchain members. It is worth mentioning the existence of Blockchain technologies that keep the transaction attributes private, such as the Microsoft Confidential Consortium Framework[3].

The aforementioned disadvantage can be minimized by building a private Blockchain network amongst the PPRL parties and using an adequated SBF configuration during the ABEL execution. The use of a private Blockchain network limits the access to the shared data only to the PPRL parties and hide the data from the external malicious parties. In order to preserve the entities' privacy from an adversary among the PPRL parties, we employ the SBF and send a split configuration (number and length of the splits) that diminish the chance of success of attacks performed by a malicious party. Furthermore, the computations performed at 3PAC Stage 2 are executed by the parties outside of the Blockchain, hiding the data from the PPRL parties. In Stage 2, the Blockchain is employed by the parties only to verify the computations and update the similarity value stored in ζ.

Another drawback of the Blockchain usage happens if one adversary controls at least 51% of the nodes of a Blockchain network. This adversary can take control over the Blockchain and change the ledger's states [5; 92; 189], i.e., switch the output of a transaction and modify the values stored in ζ. However, we will not address these attacks in this thesis. For further details, we refer the reader to the related work [44]. Our research assumes that any Blockchain owner has control of 51% of the Blockchain nodes. For instance, if two parties want to execute ABEL, they will set up a private blockchain network where each one will have one node. Thus, each party will control 50% of the blockchain network.

Figure 4.6 illustrates the ABEL execution of two participants (Alice and Bob). In this example, we assume that the parties have already deployed the STTP (Smart Contract) in the Blockchain, agreed on optimal anonymization parameters, executed all entities' anonymization in their datasets, and agreed on the number of splits ($s = 3$) and the thresholds ($\alpha = 0.7$ and $\beta = 0.4$). In summary, Alice and Bob have executed steps 1 to 3 of the 3PAC protocol.

[3]https://microsoft.github.io/CCF/

The example starts in step 4 of 3PAC. Steps 1a and 1b of Figure 4.6 illustrates the parties sending the splits to STTP.

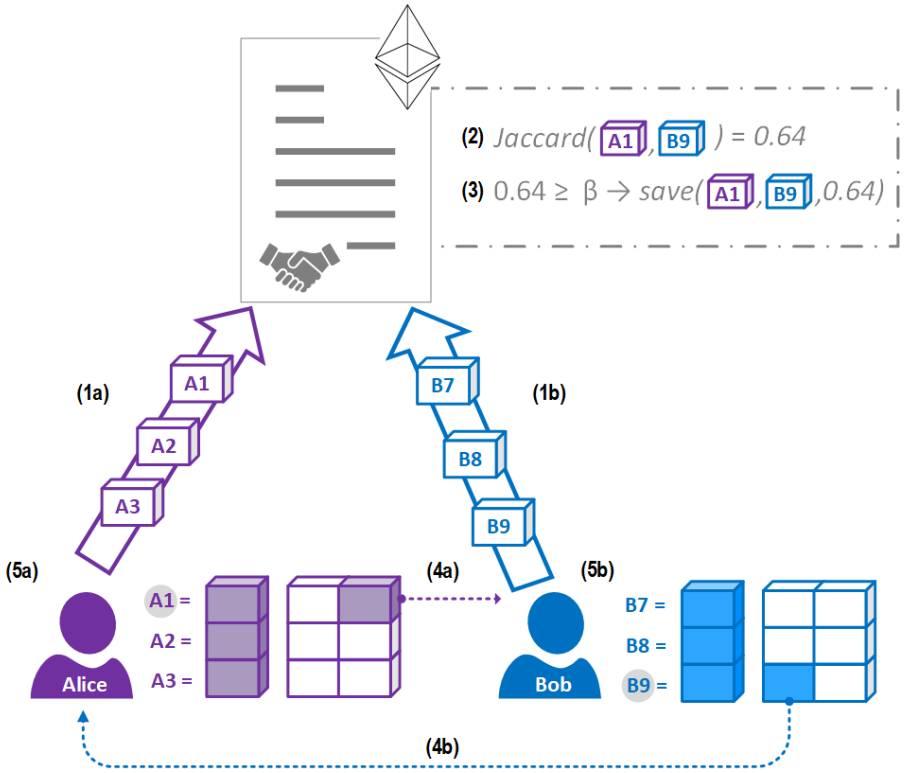


Figure 4.6: Outline of ABEL execution.

The similarity computation is performed in all splits (Cartesian product) received by SCTP. If the calculated similarity value is higher than threshold β, STTP saves the IDs of the splits as well as their similarity values in the Blockchain (step 6 of 3PAC). In our example, the restriction of Stage 1 is satisfied only by the splits A1 and B9 ($similarity(A1, B9) > \beta$); this fact is illustrated in steps 2 and 3 of Figure 4.6.

It is to mention that Stage 1 acts as a filtering process, removing from the subsequent steps entities pairs that present values smaller than the β threshold. In other words, Stage 1 reduces the number of entities (information) shared during the ABEL execution, and this reduction has an impact on the comparison reduction rate (detailed in Section 4.3) and to the privacy of our approach. For instance, notice that the ζ list was assembled using a minimal amount of information (i.e., only one split of each entity), reducing the success rate of cryptoanalysis attacks; the privacy impact is presented in Section 4.5.4.

Steps 4a and 4b of Figures 4.6 illustrates the parties exchanging among each other, the remaining splits of entities stored in the ζ list. Notice that, in our example, entities A1 and

B9 presented a similarity value greater than β. Therefore, only A1 and B9 splits will be exchanged by the parties. Notice that, by alternating the distribution of the split, Alice and Bob only have access to two splits as well as their similarity values, making it challenging to successfully execute a cryptoanalysis attack.

For each split, ABEL alternately distributed among the parties the splits of entities stored in ζ. In our example, Alice sent the third split of A1 to Bob, and Bob sent the second split of B9 to Alice. With the splits exchanged, the parties calculate the splits similarity, exchange the calculated similarity among themselves, audit the computation performed by other parties (steps 6.a and 6.b.i of 3PAC), and update the similarity value in ζ (step 6.b.ii of 3PAC).

The final stage is illustrated in Figures 4.6, 5a, and 5b, where the duplicate entities are marked (with a grey circle). Thus, at the end of the process, a party only learns which of its entities are present in the other parties' datasets. A more in-depth discussion of the privacy attacks performed during the ABEL execution is presented in the following sections (4.4.3 and 4.5.4).

4.4.3 Privacy Sketch

This section evaluates the privacy capabilities of our approach in an auditable security model using the simulation paradigm. The paradigm assesses the approach through the messages (information) exchanged by the parties and STTP during a simulated PPRL execution. The simulation was executed according to the guideline proposed by Lindell et al. [103]: the messages granted in the simulation must be the same as an adversary would have access to use in a real attack. The simulation fails if an adversary learns anything different from the expected output.

Considering that STTP is a Smart Contract, we can assume that the STTP code (protocol) can be read and audited by the parties. Moreover, after STTP is deployed in a Blockchain i) the code cannot be changed (tamper evident); ii) STTP cannot deviate from the protocol (collude); and iii) all computations (Transactions) performed by STTP will be stored in the Blockchain. We also consider that each party performs the anonymization of its respective data source. Thus, in order to provide a better explanation, the simulation paradigm was executed over Stages 1 and 2 of the 3PAC Protocol.

Before we detail the action performed in Stage 1, we present the inputs used in this

stage: one split (ϕ_1) of each entity stored in the parties' data source ($\hat{D}_p$) and the β threshold. To the best of our knowledge, the β threshold is not susceptible to any vulnerability or privacy attack.

[3PAC] Stage 1 simulation

Inputs. $\phi_i \forall i \in \hat{D}_p$, β

1. *Input Validation*. STTP validates the parties' inputs.
2. *Single Split Comparison & Classification*. STTP performs the similarity computations and publishes (steps 4 and 5) the entities pairs that have a high probability of being similar in ζ.

Output. ζ

STTP should implement a method to validate the splits (Input validation). For instance, the validation method verifies if all parties sent their splits, or if all splits have different values. Thus, if one (or more) party tries to deviate from the protocol by sending wrong inputs, STTP detects the misbehave and may abort the protocol execution. After the Input validation, STTP engages in the classification of the entities (Single Split Similarity Computation) and stores the ζ list (the output) in the Blockchain.

The ζ list contains pairs of entity ids with their similarity values (Eq. 4.4), such that these values are higher than the β threshold. Thus, since the single splits (ϕ_1) were the inputs of a Blockchain transaction (STTP execution), all the ϕ_1 are available in the Blockchain. Notice that, until this point, an adversary has access to one split of each entity, with its estimated similarity.

Regarding the privacy issues related to the usage of single splits (ϕ_1) as input to a privacy attack by a malicious party, as demonstrated in Section 4.3.2, it is a hard task to re-identify an entity using information from a set of single splits.

The information reduction (reduction rate, detailed in Section 4.3) provided by Stage 1 also helps to improve the privacy capabilities of the SBF. Due to the reduction of the number of entities employed in Stage 2, where only the entities stored in ζ will be used as Stage 2 inputs, the chance of successfully executing a frequency or cryptoanalysis attacks decreases.

[3PAC] Stage 2 simulation
Inputs *(for each party)*. $\phi_1, \varphi_p, \beta, \alpha, \zeta$
For each $\varrho \in \zeta \cap D_p$ (step 6 of 3PAC):
3. Exchange the φ_p splits, such that $\varphi_p \in \varrho$.
4. Compute the similarity of φ_p (step 6a).
5. Exchange and audit the calculated similarity in the previous step (step 6b of 3PAC)).
6. Conjointly classify the entities listed in ζ (step 7 of 3PAC).
Output. Each party learns which of its entities is also present in other parties' data sources.

The input of Stage 2 (i.e., α, β, and ϕ_1) is the same for every party. However, each party receives a different split set φ, at each iteration. Figure 4.7 illustrates the split exchange at each iteration. For instance, after the parties executed Stage 1, Charlie shared ϕ_4, with splits of entities C7 and C8 in the first iterations. Due to the filtering processing (i.e., similarity smaller than β threshold), Charlie excludes entity C8 in iteration 2. In the split exchange process, the parties alternate the splits in order to reduce the amount of shared information and distribute the split computations.

After exchanging the first split, the parties compute the similarity of the received splits' against their owns splits (step 6a of 3PAC). Then, the parties exchange among themselves the calculated split similarity values and compute the entities' overall similarity (step 6b). Notice that, at this point, an adversary may try to force the honest parties to send more information (splits), or hide information, by executing the following attacks:

- Send bogus split (e.g., with random data);
- Send wrong similarity values.

The honest parties could detect both attacks by auditing the other parties' inputs. If some adversary attempts to deviate from the protocol, it will be detected with a high probability — the misbehave identification is detailed as follow.

In order to detect the bogus split sent by an adversary, the honest parties will compare the similarity computed by themselves with the value stored in ζ. If the computed value diverges from ζ (considering the $error$), an honest party will detect the misbehave and abort the protocol execution. In other words, if an adversary sends a bogus split, e.g, a split with zero in every position, in Stage 2 step 6b, the parties will calculate the similarity and compare the similarity value against the value stored in ζ. Thus, if the calculated similarity presents

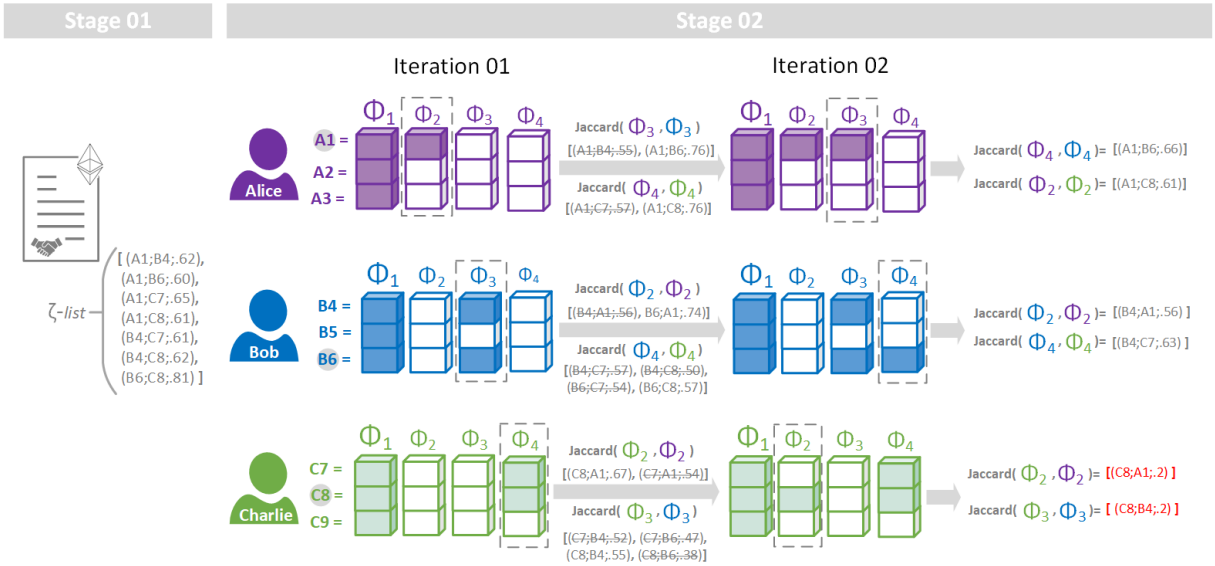


Figure 4.7: Simulation of the ABEL execution. The parties' exchanged messages (i.e., splits and similarity values of entity pairs) are designated at each iteration. The misbehavior (attack) of Charlie is illustrated in the final iteration and highlighted in red.

a value different than the expected SBF error plus the value stored in ζ, the party will detect the adversary misbehave.

Regarding the second attack, where an adversary sends wrong similarity values, the honest parties will confront the wrong similarity values with their calculated split similarity and the value stored in ζ. And abort if the exchanged similarity diverges from the calculated values or the value stored in ζ.

Figure 4.7 illustrates an attack in iteration 2 when Charlie tries to hide entity C8 by sending a lower similarity value. The similarity value sent by Charlie (0.02) is not tolerated because the overall similarity is 0.65 and the similarity of ϕ_3 is 0.61. Thus, the other parties identify the attack and abort the protocol execution.

Regarding the usage of the exchanged splits in the re-identification attack, we need to remark that our approach performs an iterative filtering process where, at each iteration, fewer splits are shared amongst the parties. In other words, only entities that will be marked as a match are shared in the last iterations, making it hard to successfully employ re-identification attacks.

In the attack illustrated in Figure 4.7, Charlie deviates from the protocol in the final iteration. Otherwise stated, Charlie could have access to the maximum number of splits

as possible, meaning that the attack was executed in the worst scenario. However, even if the parties collude, Charlie has access only to the colored split designated in iteration 2, reducing the chance of success during a re-identification attack.

Parties could collude and share the split among each other to gather more information (splits) for a re-identification attack. In other words, the collusion among multiple parties could reveal more information to be employed in an attack, increasing the chance of success in a re-identification attack. However, the consequences of collusion among the parties could be minimized if the SBF parameters are properly chosen. Furthermore, we demonstrate in Section 4.5.4 that our approach could preserve the privacy of entities against the BF state-of-the-art privacy attacks until a certain limit. We also remark that collusion is considered in a malicious adversary model, and our approach considers a covert adversary model.

4.5 ABEL Evaluation

In this section, we evaluate the ABEL approach and present a discussion about the experimental results. The main goal of the evaluation is to assess the effectiveness, efficiency, and privacy capabilities of ABEL. To this end, we formulate the following research questions:

RQ. 1 Does the ABEL approach provide an effective linkage of entities?

RQ. 2 What is the computational cost of ABEL?

RQ. 3 Is the ABEL approach capable of preserving the privacy of entities considering a covert-security model and state-of-the-art attacks?

To answer these questions, we use real-world personal data sources detailed in Table 4.2, which are widely used by the community. For further details on the data sources, see Appendix A.5.

Table 4.2: Data source characteristics.

data source	—A— x —B—	attributes	l	k
ncvr	2,890,000	8	1,144	7
mvr	2,250,000	9	1,008	5
yv-er	90,231,001	6	976	7

Considering the relevance of personal data sources, we selected three data sources from distinct countries with different linguistic and structural characteristics to provide an extensive evaluation of our approach. The YV-ER data source contains information about Italian Holocaust victims, while NCVR and MVR comprise information about voter registration from North Carolina and Michigan, respectively. We used the methodology presented in [181] to build the NCVR and MVR data sources. The former contains data collected in October 2016 and June 2020 while the latter includes data obtained in September 2014 and March 2017.

In our experiments, we compare ABEL against the regular BF representation of the entities. We also test our approach using Standard blocking [115] as a blocking stage of the PPRL. Thus, the comparison stage calculates the similarity for all entity pairs marked by the blocking stage. We also consider that the regular BF will be handled by a trusty third party.

4.5.1 Linkage Quality

To evaluate the effectiveness of our approach, we compare the results of the SBF execution, varying the number of parties as well as the α and β thresholds against the competitor (regular BF). Briefly, we compare our approach (which considers a covert adversary) against an approach that considers an HBC security model. In this experiment, we vary the split length as well as the α and β thresholds. The experimental design is detailed in Table 4.3.

Table 4.3: Experimental design.

splits	α threshold	error
0.125%	[0.1, $\cdots$, 0.9]	[0.01, $\cdots$, 0.49]
0.060%		
0.030%		
0.010%		
0.004%		

Figure 4.8 illustrates the comparison between the results achieved by the competitor and ABEL. The vertical axis represents the quality measure results, while the horizontal axis refers to the α threshold. Different colors represent each data source, while the colored dotted line delineates the competitor results, and the colored bars represent ABEL's results.

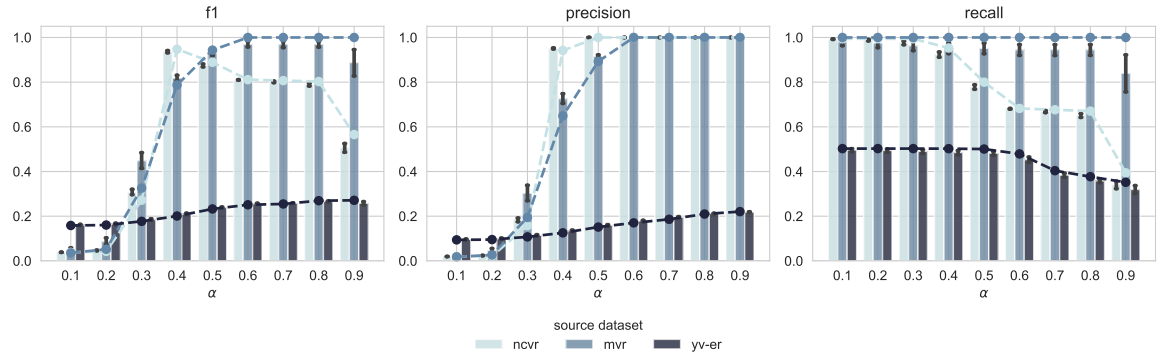


Figure 4.8: Effectiveness results of ABEL for different α values.

The results produced by ABEL are similar to the ones of the BF. However, the F1 results also show that BF overcomes SBF for some threshold values, for instance, $\alpha = 0.9$ for the ncvr and mvr data sources. Two factors can explain this fact: i) the precision and recall results; and ii) the fact that we consider a combination of different α (from 0.1 to 0.9 threshold) and β (from 0.01 to 0.4) thresholds to build the figure; in other words, we use non-optimum parameter combinations.

By observing the precision and recall results, notice that the precision value is more closer to the competitor than the recall value. These results are expected and can be explained by the fact that the ABEL approach only performs the full comparison of entities if the first split has a similarity value higher than β threshold (comparison filtering), reducing the recall. Otherwise, ABEL would miss true match entities pairs by eliminating comparisons.

In order to investigate the impact of the ABEL parameters (split length, α, and β), we chose the α threshold that presented the best result (in terms of F-1). With a fixed α, we varied each parameter combination exhibited in the experimental design. The result is depicted in Figure 4.9. The vertical and horizontal axes represent the quality measure and β threshold ($\beta = \alpha - error$), respectively. The continuous lines represent the split length (in percentage) while the red dotted line expresses the competitor result.

Due to the error introduced by SBF, smaller splits need to consider a bigger $error$ value to achieve better quality results. This error can be estimated by providing the split length as input to an exponential function, as illustrated in Figure 4.4. The vertical colored areas representing F1 in Figure 4.9 highlight the calculated $error$ for each split length. Notice that the maximum F1 value starts near the designated colored area - calculated using the optimum-β threshold for the split length.

It is important to remark that the ABEL results depicted in Figure 4.9 overcome or deliver the same result of the competitor when we employ an SBF error calculated by the exponential function

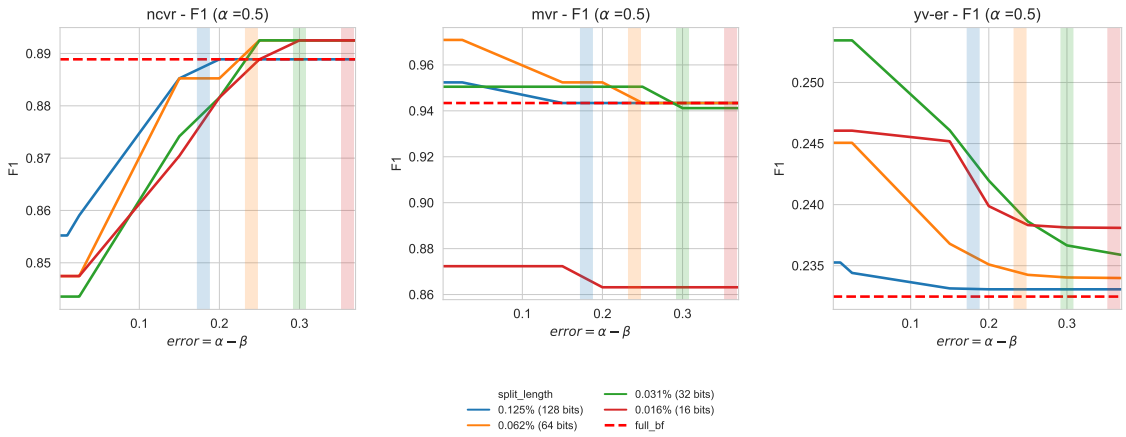


Figure 4.9: Quality results achieved by ABEL for different β-thresholds.

presented in Section 8.1. The comparison filtering performed by ABEL can explain this result. For instance, consider that $\alpha = 0.65$ and the entities pairs ('Ana','Ane') and ('Ana','Lana') have similarity values equal to 0.66 and 0.60, respectively. If we define that $\beta = 0.55$ ($error = .1$), both pairs will be tagged as similar, reducing the precision. However, if $\beta = 0.61$ ($error = 0.05$), only the first pair will be tagged as similar, improving the precision.

Figure 4.9 also demonstrates that ABEL was unable to overcome the competitor in only one scenario when we employ a small split length (16-bits split) for the MVR data source. We believe that this result is explained by the fact that a split containing 16 bits is two small, considering that the MVR was encoded with less than 1000 bits (Table 4.2), requiring a more significant SBF error than the one we considered in our experiments. This result also demonstrates the tradeoff between privacy and quality; a more in-depth discussion about this tradeoff is presented in Section 4.5.4.

The experiments presented in this section demonstrate that ABEL reaches a similar result when compared to the competitor. Furthermore, the quality metrics achieve their maximum values when we employ the optimum-β threshold. Thus, the answer to the first research question (RQ 1) of this section is: the ABEL approach provides a linkage result as effective as the approaches that utilize BF and consider an HBC security model.

4.5.2 ABEL Efficiency

In this section, we evaluate the efficiency of ABEL by analyzing its computational cost. As detailed in Section 4.4.1, ABEL computes the similarity between entities at two distinct moments. First, STTP performs the computations using only one split. Then, the parties use the output of STTP to perform the remaining similarity computations. Thus, the total computational cost is expressed as,

$Cost(ABEL) = Cost(STTP) + Cost(Parties)$.

STTP performs its computation over the splits of all entities (m) provided by the parties (P). In our experiments, m represents all entities that belong to a participant. However, if a blocking or filtering technique is employed, the available entities m are determined by these techniques. Thus, $Cost(STTP)$ is related to m and the number of participants (P), such that, $Cost(STTP) = \mathcal{O}(m_{max} \times |P|)^2$, where $m_{max} = max([m_1, \cdots, m_{|P|}])$. It is worth noticing that STTP performs the comparison only in one split(ϕ) of each entity.

It is worth mentioning that the β threshold, which is defined by the parties and employed in the First Stage of the 3PAC protocol, presents an influence on the number of pairs in ζ-list. In other words, the β threshold has an impact on the number of computations performed by ABEL in the Comparison step. The impact is illustrated in Figure 4.10, where the vertical axis represents the comparison reduction and the horizontal axis refers to the error value considered to set up β.

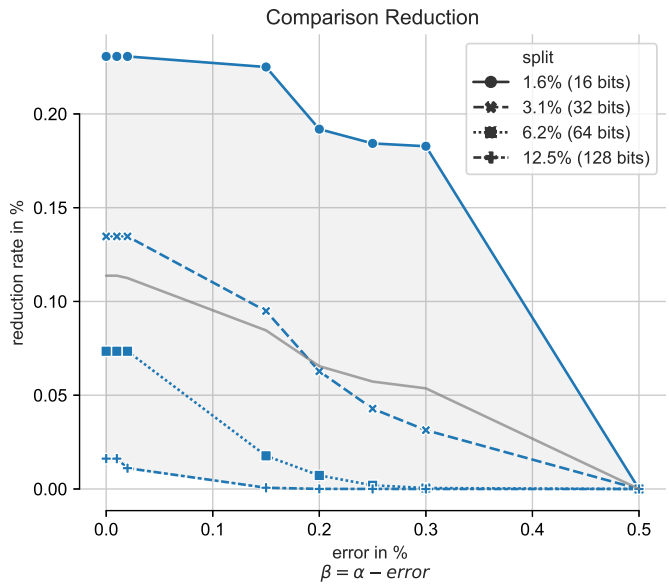


Figure 4.10: Comparison reduction provided by ABEL.

Notice that the reduction rate (rr) depicted in Figure 4.10 was achieved over a blocked data source. Figure 4.10 reveals that ABEL provides an extra filtering process to the comparisons. Furthermore, the experimental results indicate that the usage of smaller split lengths in SBF, with less than 5% of the original BF length, granted the highest rr. For instance, we were able to reduce the number of comparisons by 25% when we executed ABEL with 16 bits splits.

Regarding the split length influence over the rr, it is possible to observe that the use of smaller splits resulted in higher rr. Figure 4.10 also evidences that, when β considers a small $error$ in SBF,

the number of comparisons reaches the maximum reduction rate, regardless of the split size. These results occur due to the rigorous filtering process performed by STTP, which selects only the most similar pairs of entities to be compared by the parties.

The grey line in Figure 4.10 represents (with 95% of confidence) an exponential function that describes the reduction rate in relation to the split length and error considered in β, observed in our experiments. Thus, we can estimate the ζ-list's length by $|\zeta| = (1 - rr) \times (m \times p)$, where $0 \leq rr \leq 1$. Notice that we employ $1 - rr$ to estimate the impact of the filtering process over the number of comparisons. For instance, if we have a $rr = .25$ our approach will consider the 75% $(1 - .25)$ of the original comparisons.

The number of iterations is calculated as a ratio between the number of remains parties $(|P| - 1)$ and the number of exchangeable splits (φ). As φ is proportional to the split length (in percentage), we can define the number of iterations by Eq. 4.6.

$$
\begin{aligned}
&iterations = \lceil (|P| - 1) \times |\varphi| \rceil \\
&\textit{replacing } \varphi\textit{, where } \varphi = \frac{(s-1)}{|P|} \\
&iterations = \left\lceil \frac{(|P| - 1)(s-1)}{|P|} \right\rceil
\end{aligned}
\tag{4.6}
$$

Figure 4.11 depicts the relationship between the number of iterations, split length, and number of parties. The vertical axis refers to the number of iterations performed by ABEL whilst the horizontal axis represents the split length as a percentage of the original length. The colored lines (generated using Eq. 4.6) delineate the number of iterations for different parties (from two to five parties) for each split length.

According to Figure 4.11, if the parties employ smaller splits to improve the privacy guarantees of SBF, then the number of iterations increases, raising the computational cost of ABEL. It is worthwhile to mention that, as more parties engage in ABEL, the number of iterations slightly increases, especially for smaller split lengths. For instance, for two parties considering splits with 25% and 1% of the original size ABEL will iterate 2 and 50 times, respectively. When we employ the same slpit length (25% and 1%) for 5 five parties, ABEL will iterate 4 and 80 times.

Finally, it is possible to estimate the parties' computational cost. As shown in Eq. 4.7, the cost is related to the ζ list and the number of ABEL iterations.

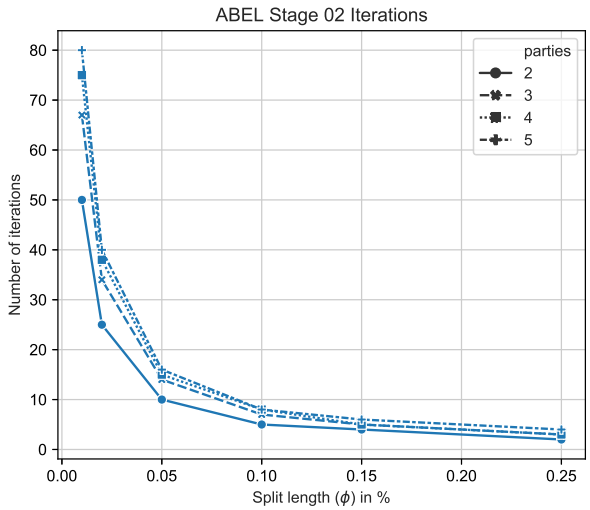


Figure 4.11: Comparison reduction provided by ABEL.

$$Cost(parties) = iterations \times \mathcal{O}(\zeta)^2 \tag{4.7}$$

The overall cost of ABEL is defined by Eq. 4.8.

$$\begin{aligned} Cost(ABEL) &= Cost(STTP) + Cost(parties) \\ &= \mathcal{O}(m_{max} \times |P|)^2 + iterations \times \mathcal{O}(\zeta)^2 \end{aligned} \tag{4.8}$$

4.5.3 Blockchain Efficiency

Since ABEL considers STTP as a Blockchain Smart Contract, we provide a Proof-of-Concept implementation using the Ethereum DL technology. In order to investigate the impact of BC, we use three different BC setups to measure the comparison time (Λ_{sttp}) between STTP and the standard BF.

We execute the experiments using private (consortium) and public Ethereum networks. For the private network (Private-PoW), we build a three-nodes network using the Prof-of-Work (PoW) consensus mechanism — this network was accessible only by the parties. Concerning the public networks, we use the Ethereum Roposten network (Ropsten-PoW) that employs the PoW consensus mechanism, and the Rinkbey network (Rinkbey-PoA) that provides the Proof-of-Authority consensus mechanism — these networks are publicly accessible.

Figure 4.12 illustrates the execution time of the individual comparisons. The vertical axis represents the execution time in seconds. The red dotted line outlines the result of the regular BF.

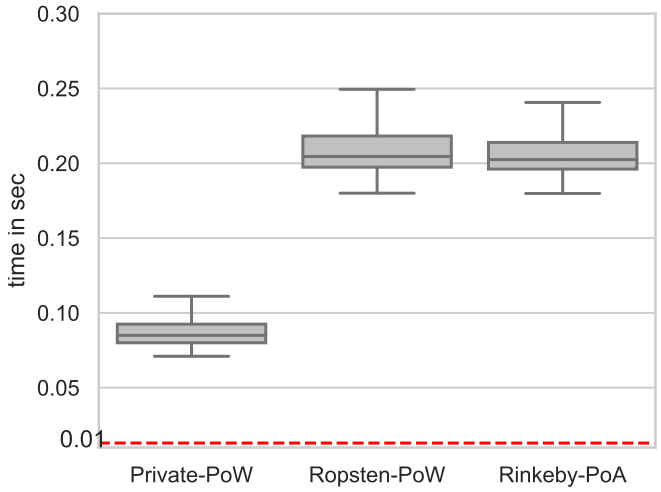


Figure 4.12: Comparison time of a pair of entities over different Blockchain solutions.

The results shown in Figure 4.12 highlight that the mean execution time of the public networks were equivalent (with 95% confidence), regardless of the consensus mechanism. Another highlight is the fact that the private network outperformed the public network, by reaching half of the execution time of the public network. However, when we compare the STTP comparison time against the regular BF, the private network was ten times slower.

These outcomes were expected and can be explained by the mining schedule of BC, the consensus mechanism, and network delays. We detail these reasons as follows. As presented in Chapter 2, each transaction needs to pay a fee to the miner, setting this fee as a critical feature in the transaction scheduling [189], with an important impact on the transaction mining scheduler. For instance, if a transaction A sets its fee to 0.1 ether and transaction B sets its fee to 0.2, probably transaction B will be executed prior to A.

In order to store and execute a transaction in BC, one miner needs to break a cryptographic enigma (in the PoW) or a miner that has the execution token needs to schedule the transaction (in the PoA). In bigger networks, such as Roposten and Rinkbey, the transaction needs to be delivered to the miner node, and the network delay has an important influence on the BL transaction scheduler. Otherwise stated, delays in the transaction propagation among nodes of the networks hold back the transaction execution.

4.5.4 Privacy Evaluation

In Section 4.4.3 we provided a privacy sketch evaluation; in this section, we evaluate if our approach is able to preserve the entity's privacy against attacks. To this end, we employ the state-of-the-art BF cryptoanalysis attacks [32; 181] using all splits available in Stages 1 and 2 for the NCVR data source and considering two and three parties. We varied the split length for each different number of parties and executed the re-identification attack over the entity splits available at the end of ABLE. In other words, we tested our approach considering that we have from 10% to 50% of the original amount of information of the entities.

As a result, we were unable to re-identify none of the entities from the data source. We attribute the privacy-preserving capability of ABEL to the: i) reduction rate (detailed in Section 4.5.2), that prevents the parties to share entities (splits) that are not possible match, and ii) the use of the SBF in ABEL, which add an error (exposed in Section 4.5.1) to the entities that are possible matches. However, we believe that our approach is unable to preserve the privacy of the entities when the parties choose inappropriate parameters. For instance, if the parties choose a low α threshold (e.g., 0.1), STTP will return almost every entity of the original data source, increasing the success rate of any privacy attack.

In this Section, we tested our approach against privacy attacks, and as a result, we demonstrate that ABEL is capable of preserving the privacy of the entities during the PPRL execution. This results answers RQ.6, formulated at the beginning of Section 4.5.

4.6 Summary

In this Chapter, we presented the SBF, 3PAC protocol, and the ABEL approach. We detailed the results of the approaches mentioned above in terms of effectiveness, efficiency, and privacy. We demonstrated that SBF improves the privacy-preserving capabilities of the original BF while maintaining the comparations' quality. We tested the ABEL implementations with several data sources and Splitting Bloom Filter parameters (number of splits and thresholds) over different Blockchain configurations (PoW, PoA, public and privates networks). All experiments conducted in public Blockchain are available at our wallet public address[4] whilst the source code is available at the main author website[5].

Regarding the results, our experiments demonstrate a relation between privacy capabilities, computational cost, and linkage quality. This relation is influenced by the BF parameters (k,l,n), SBF split

[4] `https://ropsten.etherscan.io/address/0x99429f64cf4d5837620dcc293c1a537d58729b68`

[5] `https://github.com/thiagonobrega/auditable_pprl`

length (s), ABEL thresholds error (α and β), and the number of parties (P).

The values of the aforementioned parameters should be carefully chosen. If one of the aspects is prioritized (quality, efficiency, or privacy), the remaining ones will be negatively impacted. Roughly speaking, if we use small splits we increase the privacy at the cost of minimizing quality and efficiency. Similarly, if bigger splits are preferred, we increase efficiency at the cost of lower privacy capabilities. The selection of ABEL thresholds (α and β) follows the same idea: if we consider a high error, we prioritize the quality over privacy and efficiency. The tradeoff between effectiveness (quality), efficiency, and privacy is summarized in Table 4.4.

Table 4.4: Effectiveness, efficiency, and privacy tradeoff

Parameters		Influence		
splits	**error**	**Privacy**	**Efficiency**	**Effectiveness**
bigger	bigger	↓↓ Deacrease	↑↑ Increase	↑ Increase
bigger	smaller	↓ Deacrease	↑ Increase	↓ Deacrease
smaller	bigger	↑ Increase	↓ Deacrease	↑ Increase
smaller	smaller	↑↑ Increase	↓↓ Deacrease	↓ Deacrease

As shown in Table 4.4, if we opt to maximize privacy by employing small splits and considering a small error in the thresholds, we prejudice the effectiveness and efficiency of the approach. In turn, if we employ bigger splits and a bigger error in the thresholds, we prioritize the efficiency and effectiveness, minimizing privacy. Thus, thresholds error and split length should be carefully selected to avoid impacting the effectiveness, efficiency, or privacy of the linkage process negatively.In the following chapter we present our second contribution, the Unsupervised Classification step for PPRL.

4.7 Disclaimer

This chapter [130] was published in a peer revised Journal, Information Systems, which is specialized in designing and implementing languages, data models, process models, algorithms, software, and hardware for information systems. After the publication, Peter Christen designed a novel privacy attack that could re-identify records in certain conditions. Thus, committed to research, we opted to publish a second article demonstrating the limitation of SBF in the same journal (Information Systems). The content of the article is shown in Appendix A.2.

Note

We opted to preserve the content of the original chapter/article and presented the limitation in the Appendix as a matter of transparency and to provide more information to the readers.

Chapter 5

Unsupervised Classification step for PPRL

Due to the widespread usage of private data - i.e., medical [52; 55; 76; 167], shopping preferences [33], and financial records [30] - as well as the recent data privacy laws (i.e., GDPR and LGPD) and regulations (HIPPA), PPRL has gained attention within the research community. Several works tackle privacy, efficiency, and quality aspects in different steps of the PPRL process. However, according to recent surveys [30; 33; 176], PPRL lacks contributions in the classification step.

In the PPRL context, the classification step employs the comparison step output to classify entity pairs. In other words, the classification step utilizes a vector containing the entity pairs similarity values as input to a decision model Ξ, which will classify them into matching or non-matching [128; 176].

In the RL context, i.e., a scenario with no concern regarding data privacy, several classification techniques are available. The most widely used in the RL context are the simplest ones: threshold-based and rule-based classifiers. However, more sophisticated techniques are available such as:

- The probabilistic classifier (Fellegi and Sunter method [53]), which consists in employing the frequency distributions of QIDs, similarity of the entity pairs, and error (previous estimated on the data) to classify the entities;
- Supervised machine learning classifiers, such as support vector machines and decision trees [68; 194; 51; 41; 67; 188]. These classifiers require training data with labels for matching and non-matching to train the decision model;
- Semi-supervised classifiers (active learning-based approaches) [19; 57; 96]. The goal of active

learning is to reduce the manual labeling effort as much as possible by asking the oracle (a specialist) only for the most important unlabeled training data to train and improve the decision model;

- Unsupervised machine learning classifiers: in some real-world applications, such as the PPRL scenario, there is no labeled dataset available, and it is not possible to manually label the dataset due to privacy or financial restrictions. In this context, unsupervised ML algorithms (e.g., clustering algorithms) are employed to identify matching entities [26; 48; 78; 88];

- Deep Learning (DL): in the past few years, DL has become a major direction in machine learning [91; 41; 46; 75; 105]. DL exploits some hidden structure of the data, using labeled examples, in order to automatically construct important features without the need for manual feature engineering. DL can also be used in the RL context as a classifier to decide whether an entity pair is a match or not. However, the results show that DL does not outperform current solutions on structured RL, although it can significantly outperform them on textual and dirty RL [121].

However, the panorama in the PPRL context is different. Due to the restrictions imposed by the privacy-preserving guarantees, the Classification step of PPRL has the following challenges. In general, there is no labeled data available to train a classifier, making it hard to employ supervised machine learning and deep learning classifiers. Furthermore, the classification step does not have access to the data (plain or anonymized data), but only to the entity pairs' similarity, making it hard to use a probabilistic or ruled-based classifier. Moreover, we can not ask a specialist to label a few instances to train a classifier since the specialist can not observe the data due to the privacy restrictions imposed by PPRL. Thereby hampering the usage of Semi-supervised classifiers.

Furthermore, most PPRL solutions developed so far use the threshold-based classifier [30], which is very simple. First, a specialist defines a threshold value, e.g., threshold value is defined as 90% similarity. Finally, the classifier will mark as a match all entity pairs that manifest a similarity value greater or equal to the defined threshold value. The definition of the threshold value is a complex task and requires a specialist to configure an adequate value for each PPRL scenario.

To better illustrate this problem, we plotted (Figure 5.1) the mean similarity value of the matching entities from four widely used datasets, detailed in Appendix (A.5), which belong to the following domains: i) products commercialized from best buy and abt (abt-buy), ii) books from amazon and Goodreads website (books3), iii) publications from DBLP and ACM, and iv) movies from IMDB and rotten tomatoes (movies2).

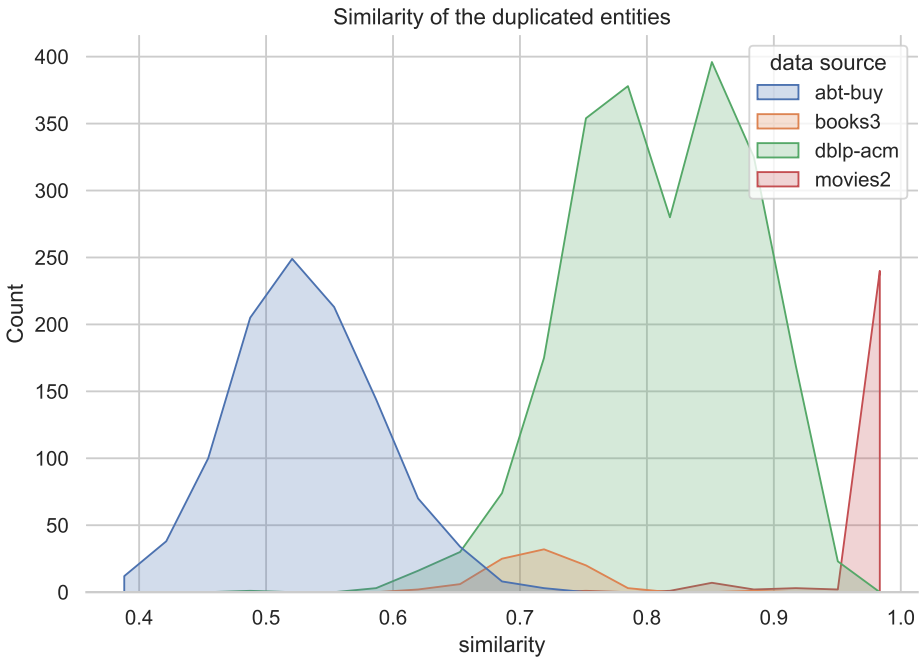


Figure 5.1: Mean entity pairs similarity by dataset.

In Figure 5.1, the vertical axis represents the number of matching entity pairs whilst the horizontal axis refers to the mean similarity of the entity pairs. Notice that more than 95% of the DBLP-ACM dataset (green area) matching instances are after the 70% threshold value. However, if we apply the same threshold value (70%) to the abt-buy dataset, the threshold-based classifier will probably mark non-matching entity pairs as duplicates. Therefore, resulting in a low linkage from the RL process. It is worthwhile to mention that the threshold value definition can be observed in all datasets depicted in Figure 5.1; there is no unique threshold value that fits all datasets. We also remark that we do not plot the non-matching mean similarity, and therefore we simplify the threshold definition problem.

In this context, we propose the Auto-Tuned Unsupervised Classification (AT-UC) step. This approach tackles the classification problem, specifically, the need for a specialist to guess the threshold classification value. Moreover, AT-UC is an unsupervised approach that does not require complex parameter tuning and/or adjustments for different datasets. To accomplish that goal, we employ Transfer Learning in a privacy-preserving context without labeled training data.

As previously explained, classification in PPRL urges for the use of ML-based classifiers. However, employing ML-based classifiers in PPRL is still an open problem [30; 165]. In order to accomplish our goal, we propose the use of public datasets to build ML-based classifiers. These classifiers will be readjusted using a technique named Transfer Learning [165], where a classifier trained in one dataset is modified and reused in a different dataset. To readjust the classifier, we intend to verify the public and private dataset's resemblance and use a resemblance metric to customize

the original model to be employed in the private dataset. It is important to notice that, due to the privacy-preserving guarantees of PPRL, the resemblance between the datasets needs to be calculated employing the anonymized representation of the private data.

The use of Transfer Learning in traditional Record Linkage has been proposed in [165; 124]. However, due to the privacy restriction, it is not possible to assume that such works can be used in the PPRL context. For instance, in PPRL, we cannot see the parties' data to adjust the transfer model. To the best of our knowledge, we are not aware of any ML-based approach that provides satisfactory results that could be applied in real-world PPRL applications. Thus, in the subsequent sections, we present the following contributions:

- A feature space that can be employed in PPRL scenarios;
- An implementation of three similarity metrics to anonymized data;
- A feature and dataset selection based on the datasets resemblance;
- A novel ML-based classifier to PPRL;
- A Proof-of-Concept implementation of the ML-based classifier;
- An empirical evaluation using real-world data to assess our contributions.

5.1 Problem Formalization

In this work, we consider a PPRL process under the HBC adversary model with p dataset custodians $[P_1, \cdots, P_p]$ with their individual datasets $[D_1, \cdots, D_p]$. Each dataset D contains anonymized records, such that $e_{[p,i]} \in D_p$. It is worthwhile to remark that we consider a dedicated Linkage Unit (LU) as a semi-trusted third party (STTP) [30]. In our approach, the linkage unit is an external party that conducts the linkage's classification step over the PPRL comparison step output [177]. We formally define the problem of the PRRL classification step on multiple datasets as follows.

Definition 6 (PPRL Classification step)**.** *The classification step aims to determine which of their records $e_{[1,i]} \in D_1, \cdots, e_{[p,k]} \in Dp$ matches (correspond to the same entity) according to a decision model $\Xi(\cdot)$. The decision model Ξ classifies records pairs stored in all datasets into one of the two classes: M for matches and U for non-matches.*

In a PPRL scenario, the use of a traditional ML-based decision model is challenging due to the absence of training data to build a classifier as a consequence of the privacy-preserving guarantees

needed by the PPRL. To overcome this problem and enable the usage of ML-based decision model in PPRL, we propose the adoption of a Transfer Learning technique to train an ML-based decision model over an auxiliary dataset, named as source dataset D_{source}, to be employed over a sensitive dataset, named as target dataset D_{target} [197; 132]. We now present a definition of inductive transfer learning in a PPRL context.

Definition 7. *(Inductive Transfer Learning) Given a source dataset D_{source} and a target dataset D_{target} with the same learning task ($T_{source} = T_{target}$) of identifying the duplicated entities, transfer learning aims to help improving the learning of the target classifier $\Xi(\cdot)$ in D_{target} using the knowledge in D_{source}, where $D_{source} \neq D_{target}$.*

Regarding the information presented in the source and target datasets, we assume that the D_{target}, contains the similarity values of the masked records of p datasets (comparison step output) [177], where each pair of records is represented by similarity metrics x_j. Thus, we have $D_{target} = \{(x_1, \cdots, x_j)_1, \ldots, (x_1, \cdots, x_j)_t\}$, where the t is length of the comparison step output. Regarding D_{source}, it contains an extra information in each row, the linkage label y, i.e., $D_{source} = \{(x_1, \cdots, x_j, y), \ldots, (x_1, \cdots, x_j, y)\}$. This scenario, where we do not have any training data available from D_{target} and an adequate amount of training data from D_{source}, is named as "unsupervised domain adaptation" in Transfer Learning [132; 192]. Thus, we state the Unsupervised Domain Adaptation for PPRL problem as follows.

Problem Statement 4 (Unsupervised Domain Adaptation for PPRL)**.** *Given D_{target}, with no available training data, how to train a good ML decision model (Ξ) by adapting the training data from D_{source} and preserving the privacy of the target dataset?*

Note that, different source datasets may produce different classifiers [106; 186; 197; 192], and therefore may influence the linkage quality (F1) of the Ξ over D_{target}. Thus, the usage of an adequate D_{source} as a training dataset is crucial to our approach, and we need to address source selection in a PPRL context.

Problem Statement 5 (Unsupervised Source Selection for PPRL)**.** *Let C be a set of candidates dataset, $C = [D_1, \cdots, D_p]$. How to select a suitable candidate dataset for the target dataset? In other words, how to select a D_{source} that may train a good decision model Ξ to D_{target} without disclosing the target (plain or anonymized) information?*

As previously stated, we use n similarity metrics in our work, and the selection of proper similarity metrics may increase the linkage quality [100]. However, the existing methods were proposed

to a RL scenario, a scenario where it is possible to observe the original data. To select the proper similarity metric, in the PPRL context, we have to address the following problem.

Problem Statement 6 (Unsupervised Similarity Metric Selection for PPRL)**.** *Given a D_{source}, how to select the most relevant similarity metrics to be employed as features to a ML-based decision model (Ξ) for D_{target}, considering the privacy-guarantees of PPRL?*

Given the aforementioned problems, the goals of this contribution can be summarized as:

Given:	$\{C, D_{target}\}$
Find:	D_{source} *and* Ξ
Maximizing:	$F1(\Xi, D_{target})$
Considering:	*PPRL with a HBC adversary model*

5.2 Auto-Tuned Unsupervised Classification step

In the PPRL context, the linkage quality is heavily impacted by privacy restrictions. For instance, the PPRL parties can not examine the other parties' data (entities/records), making it hard even to establish the threshold that will be employed to define whether a record pair is duplicated or not. In this context, we propose the use of an Unsupervised Domain Adaptation [132], a Transfer Learning approach to overcome the challenges of building a classifier using only the information available in a traditional PPRL process. This section presents an Auto-Tuned Unsupervised Classification step approach (AT-UC) for PPRL. The main goal of AT-UC is to provide an effective ML-based classifier for the PPRL process.

The Unsupervised Domain Adaptation employs an auxiliary dataset, named as the source dataset. The source dataset is altered (e.g., by excluding/modifying entities or attributes) to train a classifier to be executed over the private data, the target dataset [192]. Intuitively, the source and target datasets must have some characteristics in common. Thus, in order to properly employ the AT-UC, we have to address the two research questions presented in Section 5.1: i) How to select a suitable source dataset? and ii) How to alter the source dataset to train a good classifier for the target dataset?

In order to address the research questions and design the AT-UC, we have to make assumptions about the features space, source dataset, and adversary model. Regarding the features, we assume a

homogeneous feature space [134]. In other words, by considering the similarity values (calculated by the comparison step of PPRL) as features, we assume that the source and the target datasets share the same set of features.

Regarding the source dataset, the PPRL parties will elect the source candidates dataset from datasets that do not store sensitive data, e.g., public or synthetic datasets. Concerning the adversary model, we consider a three-party protocol with an HBC adversary model; alternatively stated, a semi-trusted third party (STTP) will perform the classification step obeying the pre-agreed protocol between the PPRL parties. Figure 5.2 illustrates the AT-UC execution outline by the semi-trusted third party.

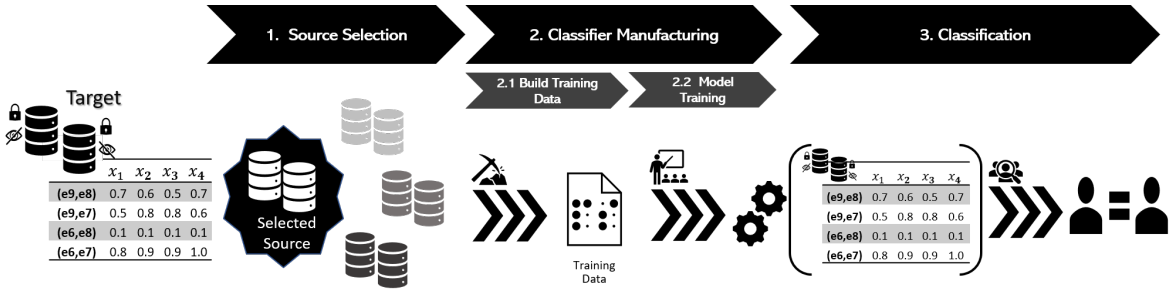


Figure 5.2: AT-UC execution outline.

Notice that AT-UC builds a classifier using a three-stage Transfer Learning approach. The first stage (Source Selection) consists of selecting a source dataset as well as the features that will be employed to train a classifier. The second stage aims to build relevant training data and a classifier for the target dataset. Finally, in the third stage (Classification), the classifier trained in stage 2 is executed over the target dataset. Each stage of AT-UC is detailed in the following sections.

For didactic reasons, Figure 5.2 does not show an important step of the AT-UC execution, the initialization stage. Before PPRL begins, during the setup stage of PPRL, each party needs to upload to the semi-trusted third party a pair of datasets with the linkage labels between them.

For example, suppose that two hospitals aspire to find patients treated by COVID-19 in both hospitals. Consider that the hospitals agree to represent the patient records with three attributes (first name, last name, and address). In that case, one hospital could upload voter registration records with three attributes (name, family name, and country), and another hospital could upload a publication dataset with two attributes (full name and university address). Notice that voter registration and publication datasets are public, with an accurate gold standard, and widely used by RL and PPRL communities.

Regarding the target dataset, input of AT-UC, as illustrated in Figure 5.2, each row contains the identifiers (id) of the entity pairs, and each column represents a different similarity metric. The source dataset has the same format as the target dataset with an extra column, the linkage label, indicating if the row represents a match or not.

5.2.1 Source Selection

The Source Selection stage aims to identify a source dataset and a feature space (similarity metrics) that can train an effective classifier to the target dataset. Moreover, we present in detail the idea of source and features (similarity metrics) selection of AT-UC.

Regarding the source selection, it is intuitive that various source datasets can produce different linkage results to the target dataset in a Transfer Learning scenario. Thus, finding the most related source dataset in terms of similarity metrics to train our classifier is crucial to our approach.

To tackle this stage, we have to consider the limitations imposed by PPRL: i) the privacy of the target dataset; ii) the absence of linkage label in the target dataset; and iii) the entities' data (neither raw nor anonymized data) can not be observed to avoid reidentification attacks by the STTP. In other words, identify a suitable source dataset is a hard task because we do not have a linkage label in the target dataset to test the classifier, and the classifier must be trained without the entities' data, using limited information.

To overcome these limitations, we use the similarity metrics as features to our classifier and, consequently, to identify a relevant source dataset to the target. In order to select a source dataset, we employ an indirect measure; specifically, we use the work of Ben-David et al. [14; 13] which suggests that a suitable source dataset is one from which a classifier cannot learn to identify if a row is originated from the target or source dataset. In other words, a suitable source dataset must have a small $d_{H\Delta H}$ distance [13]. It is worth mentioning that AT-UC differs from the work of Thirumuruganathan et al. [165] that recommends using a source dataset with Matthews Correlation Coefficient (MCC)[1] [9] smaller than .2 as negative training criteria, whereas our approach proposes source selection that considers minimum $d_{H\Delta H}$. Algorithm 2 ($selectSourceDataset$) presents this process in more detail.

[1]The MCC or phi coefficient is used in machine learning as a measure of the quality of binary

Algorithm 2: *selectSourceDataset*

input : $C = \{D_1, \cdots, D_N\}$,$\mathbf{D}_{target}$

output: $\mathbf{D}_{train}$, $d_{H\Delta H}$

```
1  distances ← ∅
2  datasets ← ∅
3  for D_candidate ∈ C do
4      D_base ← prepareSource(D_candidate, D_target)
5      D_sample ← sample(D_base)
6      model ← trainSeparationModel(D_sample)
7      D_predicted ← predict(model, D_base)
8      distance ← calculate_d_HΔH(D_predicted)
9      D_canditade ← filter(D_predicted, 'source')
10     distances.append(distance)
11     datasets[distance] ← D_canditade
12 end
13 d_HΔH ← min(distances)
14 D_train ← datasets[d_HΔH]
```

The input of the $selectSourceDataset$ is a set of source candidate datasets (C) and the target dataset. For each candidate, we execute a pre-processing step over the source and target (line 4) to build a training dataset for a classifier ($separationClassifier$) that will distinguish if a row came from the target or the source dataset. In this pre-processing step, we concatenate the source and target datasets, i.e., we label the rows from the source and target with 'S' and 'T', respectively.

Figure 5.3 represents the execution of one iteration of Algorithm 2. Notice that the target dataset contains only the similarity metrics (X_n), while the source candidate dataset contains a label (y) indicating whether the an entity pair represents a match (1) or not (0). The pre-processing step and origin labeling procedure (line 4) are depicted in Figure 5.3.a. Notice that each entity pair is identified according to its origin, i.e., 'S' from source and 'T' from the target dataset.

Explaining the Idea

The purpose of the $prepareSource$ method is to create a training dataset to build a classifier that intends to identify whether an instance is originated from the source or target datasets.

Moreover, we also handle aspects that could impact the classifier quality in the pre-processing

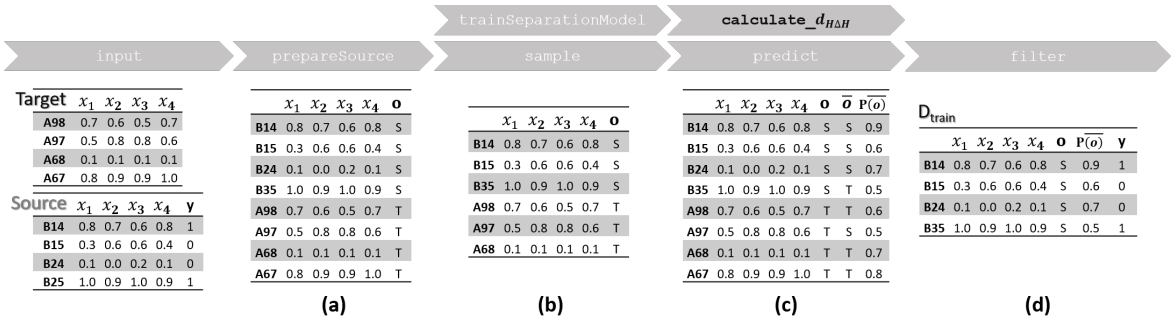


Figure 5.3: Source Selection example.

step of the $selectSourceDataset$ algorithm. In other words, traditional supervised and unsupervised learning algorithms work under the single-label scenario, i.e., each row (instance) in the training set is associated with a single label which characterizes its property. However, considering that we are employing entity pairs similarity as features, it is possible to exist two identical rows (with the same feature values), where one is labeled as 'source' and the other as 'target'. This raises uncertainty to the classifier. To avoid addressing the multi-label problem in this work, we disregard these conflicting labels in the pre-processing step.

After the data pre-processing, we extract a random sample (line 5) of the D_{base}. To this end, we employ D_{sample} to train the separation classifier sc - line 6 of Algorithm 2, illustrated in Stage b of Figure 5.3. In the following step (line 7), we employ the separation classifier over the D_{base} to predict if the rows are originated from the source or the target dataset. The prediction is stored in the $D_{predicted}$. It is worth to remark that the output of $D_{predicted}$ contains the same information of the D_{base} with two extras columns appended: the prediction of the classifier ('S' or 'T') and the confidence of prediction which varies from 0 to 1.

Figure 5.3.c depicts the $D_{predicted}$ (line 7 of Algorithm 2). Notice that the prediction is depicted by the column $\overline{o}$ and the confidence of prediction is represented by $P(\overline{o})$. The prediction will be utilized in the $d_{H\Delta H}$ calculation, whilst the prediction probability will be employed in the next step of AT-UC (Stage 02), detailed in the following section.

Explaining the Idea

The output of the $predict$ method contains two relevant pieces of information, the prediction label and the confidence of the prediction. The label is used to calculate the $d_{H\Delta H}$ distance. In turn, the confidence of the prediction will be employed during the Domain Adaptation, Stage 3 of Figure 5.2.

As demonstrated by Ben-David et al. [13; 14], the distance between the source and the target dataset distributions ($d_{H\Delta H}$) can be approximated by the output of the $separationClassifier$. Thus, in line 8, we use Equation 2.3 to calculate the $d_{H\Delta H}$. At line 9, we filter the rows that actually came from the source with their prediction probability as a candidate dataset ($D_{canditate}$), Figure 5.3.d. Moreover, for each $D_{canditate}$ in C, we save the distances (line 10) and the candidate's dataset in a map indexed by the distance (line 11). Finally, we return a training dataset that presents the smallest $d_{H\Delta H}$ (lines 13 and 14).

5.2.2 Feature Selection

The selection of the set of features to be employed in the classifier poses an important aspect to the linkage quality. The works [100; 192] demonstrate that an adequate similarity metric (features) could improve linkage quality in a traditional Record Linkage scenario. However, in our case, due to the privacy-preserving limitations - e.g., the STTP is unable to access the raw values of the entities - we cannot employ the techniques described in the previous works.

To tackle the privacy restrictions imposed by PPRL, we utilize the same intuition of Algorithm 2 to select the features. Moreover, we assume that the smallest $d_{H\Delta H}$ distance could select an adequate set of features and a source dataset. In summary, we vary the feature set of the candidate datasets, applying Algorithm 2 to select the feature combination that provides the smallest distance to the target dataset.

Algorithm 3: *featureSelection*

input : $C, D_{target}, features$

output: $\mathbf{D}_{train}$

1 $distances \leftarrow \emptyset$
2 $datasets \leftarrow \emptyset$
3 **for** $D_{candidate} \in C$ **do**
4 **for** $feature \in features$ **do**
5 $D_{target} \leftarrow$ `extractFeatures(`D_{target},$feature$`)`
6 $D_{candidate} \leftarrow$ `extractFeatures(`$D_{candidate}$,$feature$`)`
7 $\mathbf{D}, d \leftarrow$ `selectSourceDataset(`$D_{candidate}$,D_{target}`)`
8 `distances.append(`d`)`
9 $datasets[d] \leftarrow \mathbf{D}$
10 **end**
11 **end**
12 $\mathbf{D}_{train} \leftarrow datasets$[`min(`$distances$`)`]

The input of Algorithm 3 is a set of source candidate datasets (C), the target dataset (D_{target}), and a set of features to be tested. Notice that the input of Algorithm 3 is almost the same as Algorithm 2, with an extra parameter, a set of similarity metrics that will be employed as features to our classifier. The set of features could include the combination of all features ($\bigcup_{r=2}^{n} C_r^{|features|}$). It is worthwhile to mention that the PPRL parties could eliminate some features combination to reduce the computational cost of this step.

Algorithm 3 iterates over the set C (line 3) and, for each $D_{candidate}$, we extract one feature combination, provided by features parameter, for $D_{candidate}$ and D_{target} (lines 5 and 6). For instance, if we have three similarity metrics (i.e., Jaccard, Dice, and Hamming) in the original feature space, the method calls $extractFetature(D_{candidate}, [jaccard, dice])$ which returns $D_{candidate}$ without the hamming column.

For each combination of feature and source candidate, Algorithm 3 calculates the distance between the source candidate and the target (line 7) and stores the distance (line 8) as well as the $D_{candidate}$ in a map indexed by the distance (line 9). Notice that Algorithm 3 calls the $selectSourceDataset$ method (Algorithm 2) in order to calculate the distance of the source and target datasets. It is worthwhile to mention that, the $selectSourceDataset$ method (line 7) will return the distance between the $D_{candidate}$ (selected by $extractFeatures$ method in line 6) and D_{target}.

Considering that we calculated the distance of every combination of $D_{candidate}$ and features to the D_{target}, in the final step, the algorithm returns a training dataset with features that present the smallest distance to the target to be employed in the Domain Adaptation stage (line 12).

5.2.3 Classifier Manufacturing Stage (Domain Adaptation)

The Classifier Manufacturing stage aims to build a training dataset by adjusting the source dataset and training a ML-based classifier, illustrated at Stages 2.1 and 2.2 of Figure 5.2. In Stage 2.1, we perform the source dataset adjustment by selecting the entries (rows) that are more relevant to the target dataset, assuming that some rows of the source dataset are more appropriate to train the target classifier than others.

The need for stage 2.1 is justified by the limited information available from the entities that are used as training data. Thus, if we do not select the most suitable entities pairs to train a classifier, we could add a bias from the source dataset to the classifier, resulting in low linkage quality to the target dataset.

To reduce the source bias, we use only a percentual (μ) of the most relevant rows of the source (D_{source}) in the training dataset. To explain how we select the rows from the source, we need to

		x_1	x_2	x_3	x_4	L	$P(\bar{o})$
	~~(e1,e4)~~	~~0.8~~	~~0.7~~	~~0.6~~	~~0.8~~	~~M~~	~~.50~~
	~~(e1,e5)~~	~~0.3~~	~~0.6~~	~~0.6~~	~~0.4~~	~~U~~	~~.42~~
	~~(e2,e4)~~	~~0.1~~	~~0.0~~	~~0.2~~	~~0.1~~	~~U~~	~~.34~~
✓	(e2,e5)	1.0	0.9	1.0	0.9	U	.90
✓	(e9,e8)	0.7	0.6	0.5	0.7	M	.80
✓	(e9,e7)	0.5	0.8	0.8	0.6	U	.79
✗	(e1,e5)	0.1	0.1	0.1	0.1	U	.78
✗	(e6,e7)	0.8	0.9	0.9	1.0	U	.60

Table 5.1: Row selection, with $\mu = 40\%$. x_n represents the similarity values, L is the linkage label (where M indicates match and U no match), and $P(\bar{o})$ is the prediction probability.

remember the information available at Classifier Manufacturing input. Each row of the source dataset contains the similarity metrics values (features), the label marking if the entities are duplicated, and the probability of a row being originated from the target, added at line 7 of Algorithm 2. Thus, the Classifier Manufacturing stage employs the prediction probability to elect the most relevant rows. Table 5.1 illustrates the execution of our method.

In the example of Table 5.1, we set $\mu = 0.4$, i.e., we are interested in the 40% most relevant rows of the dataset. In other words, in our example, we will select three rows (40%) from all eight available rows. First, we remove the rows that Stage 1 did not mark as originated from the target dataset (rows crossed by the red line), i.e., rows which the prediction probability is smaller than 50%.

In the subsequent step, we order the remaining rows by the prediction probability and return the training dataset by selecting rows associated with the highest probabilities ($P(\bar{o})$) - rows marked with a green tick in Table 5.1. Only three (40%) of the eight remaining rows were used to build the training dataset.

It is worth mentioning that, differently from other works Kirielle et al. [88], we do not use a pseudo-label function to label the target data. Instead, we assume that the number of match instances in a typical RL process is significantly smaller than non-match instances. In other words, instead of considering the target dataset's label (or pseudo labels) to perform the domain adaptation, we employ the overall source/target datasets similarity to perform a domain adaptation. Furthermore, to ensure the existence of matching instances in our training data (source dataset), we check if the training data has a minimum percentage matching example - line 5 of Algorithm 2. For instance, in

our experiments, we require that the number of matches is at least 10% of the non-matches. If the selected instances do not fit the criteria, the dataset can not be employed as a source dataset.

The Classifier Manufacturing stage of the AT-UC differs from two other Transfer Learning techniques. These techniques are detailed as follows. The Naive Transfer Learning is a widely used TL technique [131; 192; 197; 196; 186]. This technique proposes that a classifier for a target dataset can be trained using the complete source dataset, without any modification. In other words, if we do no execute the Classifier Manufacturing stage, AT-UC selects a source for a Naive Transfer Learning technique.

The work of Thirumuruganathan et al. [165], which does not consider the privacy of the entities and has access to the raw entities data, considers that all predicted rows are originated from the target dataset. Thirumuruganathan et al. address the source bias by considering a high dimensional feature space with at least 300 features. However, the Naive Transfer Learning technique as well as a version of the work of Thirumuruganathan et al. were overcame by our approach in a PPRL scenario. The results are detailed in Section 5.3.

5.3 Evaluation

In this section, we evaluate the AT-UC effectiveness, efficiency and privacy in a PPRL scenario. To this end, based on the problem statements introduced in Section 5.1, we present a discussion regarding the experimental results to answer the following Research Questions (RQs):

RQ.1: Is $d_{h\Delta h}$ able to select a suitable source dataset compared to other methods (e.g. MCC), considering a PPRL scenario?

RQ.2: Is feature (similarity metrics) selection able to improve the linkage quality of our approach?

RQ.3: Is AT-UC able to improve the PPRL quality results when compared to the baseline (the threshold-based classifier) and the competitors (e.g., naive, or transER [88] classifiers)?

RQ.4: What are the most adequate classifiers to be used as separation and target classifiers?

In order to measure the linkage quality of the proposed approach, we employ the classical metrics: Precision, Recall, and F1 (explained in Section 2.3.3).

Before defining the experimental setup, we present considerations regarding the datasets, similarity metrics (features), ML-Classifiers, anonymization parameters, baselines, and competitors employed in our experiments.

5.3.1 Datasets and similarity metrics

To answer the aforementioned research questions, we employed 15 pairs of real-world and synthetic datasets from the following domains: personal, movies, books, restaurants, citation, and products. A summary of the data sources characteristics and their anonymization parameters are shown in Table 5.2.

context	DATASET DETAILS							BF	
	name	ds-type	input-type	entities pair	matching pairs	attribute number	missing values	n_hash	bits
personal	census	synthetic	dirty	685,584	345	6	291	4	472
	mvr	real	clean-clean	2,250,000	150	9	1,772	5	1,008
	nvr	real	clean-clean	2,890,000	170	8	352	7	1,144
	tse	real	clean-clean	4,000,000	154	3	0	7	1,072
	yv-er	real	dirty	90,231,001	5,109	6	5,551	7	976
books	amazon-barnesnobel1	real	clean-clean	126,360	232	2	3	7	1,984
	goodreads-barnesnobel	real	clean-clean	90,662	89	4	925	9	4,632
	amazon-barnesnobel2	real	clean-clean	83,448	64	3	331	8	2,760
movies	imdb-roten	real	clean-clean	310,248	190	3	0	9	5,456
	imdb-tmd	real	clean-clean	127,086	356	2	0	4	584
restaurants	fodors-zagats	real	clean-clean	176,423	112	4	0	7	1,288
	yelp-yellowpages	real	clean-clean	76,874	116	6	0	7	1,248
	yelp-zomato	real	clean-clean	98,496	119	3	0	7	1,112
citacion	dplp-acm	real	clean-clean	6,001,104	2224	4	14	8	6,400
products	abt-buy	real	dirty	1,157,776	1076	1	0	8	6,144

Table 5.2: Datasets details

Each dataset in Table 5.2 presents different characteristics, such as the dataset type, input type, number of entity pairs, duplicated entities, missing values, number of attributes, level of dirtiness of attributes (missing values), and the anonymization parameters.

The input type indicates if all parties executed a de-duplication process during the pre-processing step - i.e., Clean-Clean RL [33]. This is relevant to the classification step since, if a dirty input is employed to a PPRL context, we could have '1:m' matching results; in other words, we could have one entity marked as similar to m other entities. This makes it harder for the classifier to identify the matching entities.

Our experiment uses all personal datasets as target datasets because, typically, PPRL is employed to identify duplicated entities in data that contains private personal information. As source datasets, we use all datasets, including the personal datasets. It is worthwhile to mention that we do not employ

the same dataset as source and target. For instance, if CENSUS is the target dataset, all datasets are used as source candidates, except CENSUS.

Considering the relevance of personal datasets, we selected one synthetic (census) and four real-world datasets from three distinct countries with different linguistic and structural characteristics to provide an extensive evaluation of our approach. The YV-ER dataset contains information about Italian Holocaust victims. The NCVR and MVR comprise information about voter registration from North Carolina and Michigan, respectively. Finally, the TSE dataset contains information about Brazilian politicians. The methodology presented in [29; 82; 127] was employed to build the NCVR using data collected in October 2016 and June 2020, the MVR utilizing data obtained in September 2014 and March 2017, and the TSE dataset applying data gathered in 2014 and 2018 elections.

5.3.2 Similarity Metrics

An important remark about our experiments is the similarity metrics that we consider. Traditionally, the PPRL approach considers only one token similarity metric, Jaccard or Sørensen–Dice distance, to classify the BF anonymized representation of the entities [177; 157; 171; 135; 30].

Training a classifier using only two distances from the same family (token distance) can result in low linkage quality. The low linkage quality can be explained by the low variance[2] and high bias[3] of the classifier; in other words, a classifier trained with two token distances may be underfitted.

To overcome the underfitting problem, we assume that the similarity metrics (features) diversification could improve the classifier accuracy and implement five similarity metrics. We diversify our similarity metrics by selecting functions from different families (Token, Edit-based, and Compression-based distance) to capture various aspects of the data.

In order to better understand the similarity metrics, first, we need to present the notation used in these functions. Let A, B be two BF with the same number of bits (l), $|A|$ be the number of 1s stored in the filter. Knowing that the n-grams of the entities are mapped as 1 in BF, we use the word

[2] A model with high variance pays much attention to training data and does not generalize on the data which it has not seen before. As a result, a high variance may result from an algorithm modeling the random noise in the training data (overfitting). Such models perform very well on training data but produce high error rates on test data.

[3] Bias is the difference between the average prediction of a model and the correct value which we are trying to predict. The bias may produce a mistake from erroneous assumptions in the learning algorithm. Models with high bias pay little attention to the training data and oversimplify the model. Moreover, high bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting), leading to a high error on training and test data.

n-gram to reference the positions that were mapped as 1 in the filter. In the following, we detail three Token distances (Jaccard, DICE, and Overlap), one Edit-based distance (Hamming), and the Entropy as the Compression-based distance.

Jaccard similarity[10] is computed as that the number of n-grams shared (represented by 1), divided by the number of all unique n-grams present in both filters.

$$jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

The Dice's coefficient [129] is defined as two times the number of common n-grams in the filter and divided by the total number of n-grams present in both strings.

$$dice(A, B) = \frac{2 \times |A \cap B|}{|A| + |B|)}$$

The overlap coefficient measures the overlap between two BFs, considering two filters a full match if one is a subset of another. It is defined as the size of the intersection divided by the smaller of the size of the filters:

$$overlap(A, B) = \frac{|A \cap B|}{min(|A|, |B|)}$$

Regarding the Edit-based similarity metric, we implemented the Hamming distance [28]. This distance is computed by overlaying one filter over another and finding the positions where the filters vary. In other words, it measures the minimum number of substitutions required to transform one filter into the other.

$$hamming(A, B) = 1 - \frac{min(|A| - |A \cap B|, |B| - |A \cap B|)}{l}$$

Shannon entropy [36; 127] is a well-known concept in Information Theory. It is employed for quantifying the expected value of information in binary data, a BF in our context. Let X be a BF with alphabet χ (where $\chi = [0, 1]$) and the probability mass function $p(x) = Pr\{X = x\}, x \in \chi$, we can define *entropy* of a BF as $H(X) = -\sum_{x \in \chi} p(x) \log p(x)$ [36]. Thus, the Entropy distance measures the amount of information stored in a BF, and we can measure the difference of information stored in the BF employing the following equation:

$$entropy(A, B) = 1 - \frac{min(H(A), H(B))}{max(H(A), H(B))}$$

5.3.3 Machine Learning Techniques

We tested our approach using four different machine learning classifiers: i) Logistic Classifier (Logistic), ii) Support Vector Machine Classifier (SVM), iii) Gradient Booster Classifier (GBC), and iv) Decision Tree (DT). We use these classifiers to train the separation model (line 6 of Algorithm 2) and the target classifier. It is worth mentioning that the Logistic, SVM, GBC, and DT classifiers are reported as the most used non-Deep Learning techniques in the context of RL [28; 84; 91].

5.3.4 Experimental Design

To execute our experiments, we first anonymize and calculate five similarity metrics (Jaccard, Dice, Overlap, Hamming, and Entropy) for all entity pairs in the datasets, in order to generate the output of the Comparison step of PPRL. Then, we vary the ML techniques for the separation model and the target classifier for each dataset.

A standard PPRL or RL process has to deal with an unbalanced instance of the label. In other words, typically, the RL process presents more non-matching entity pairs than matching ones; we extract samples (lines 4 and 5 of Algorithm 2) with, at least, 1:10 proportion of non-matching entities. We also consider $\mu = 10\%$ to filter the most relevant rows during the Classification Manufacturing stage.

We compare our approach against a threshold-based classifier (baseline) and competitors. As competitors, we employ the naive-based transfer learning (naive) [28], CORrelation ALignment (coral) [158], and the TransER [88]; we also compare the $d_{h\Delta h}$ against a state-of-the-art metric proposed in the work of Thirumuruganathan [165].

Our approach and all baseline approaches were implemented in Python 3, and we ran all experiments on a Linux server with 2.4 GHz CPUs and 24 GBytes of RAM. The programs and datasets are available at the authors' website[4].

5.3.5 Linkage Quality Results

In order to answer the first three RQs, we use as separation model and target classifier the Logistic classifier because it has produced the best result for the majority of the experiments. Latter, to assess the impact of different classifiers regarding the overall linkage quality (RQ.4), we tested our

[4]https://www.github.com/thiagonobrega/

approach considering all models presented in Section 5.3.3.

RQ.1) Is the $d_{h\Delta h}$ able to select a suitable source dataset compared to other methods, considering a PPRL scenario?

In our approach, we consider the $d_{h\Delta h}$ distance to measure if the candidate dataset is related (in terms of similarity metrics) to the target dataset. In its turn, the work of Thirumuruganathan et al. [165] recommended the usage of a source dataset that presented a MCC smaller than 0.2.

In order to investigate if $d_{h\Delta h}$ can provide good linkage results in a PPRL scenario, we plotted $d_{h\Delta h}$, MCC, and F1 by source and target datasets, considering all similarity metrics presented in Section 5.3.2. Figure 5.4 (left side) illustrates the $d_{h\Delta h}$ whilst the MCC is depicted on the right side.

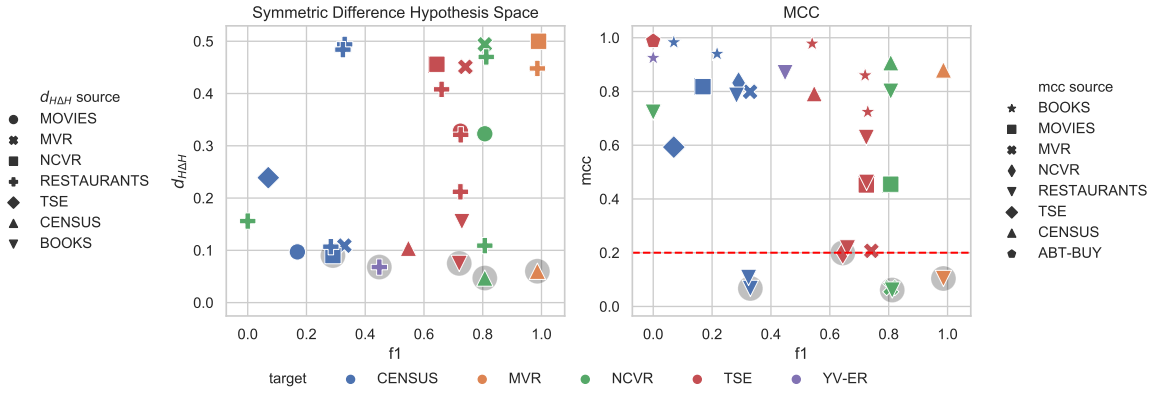


Figure 5.4: Comparison between the $d_{h\Delta h}$ and MCC regarding dataset relatedness distance.

The vertical and horizontal axes of Figure 5.4 represent the distance ($d_{h\Delta h}$ or MCC) and F1, respectively. The colors indicate the target dataset, while the shapes designate the source dataset used to train the classifier; notice that the shape legend is different on both sides of Figure 5.4. The shadowed shapes (grey circled) specify the source dataset that should be used by our approach; in other words, it indicates the dataset that presented the smallest distance between the candidate and target datasets. Also notice that we plotted a red line in the MCC, marking the threshold (0.2) pointed as the maximum values that should be considered by a TL approach that employs the MCC [165].

Observing Figure 5.4, it is possible to notice that, by employing the $d_{h\Delta h}$, we are able to execute AT-UC in all target datasets, while when we use MCC, we are unable to execute AT-UC on the YV-ER dataset. Notice that, by considering YV-ER as the target dataset, the MCC (purple triangle shape) is 0.87. Thus, with $MMC = 0.87$ above the 0.2 threshold, we disregard the restaurant dataset as the

source, preventing the linkage to this dataset when we employ MCC.

When we compare the source datasets selected by $d_{h\Delta h}$ and MCC, notice that, by considering the $d_{h\Delta h}$ distance, the linkage quality is higher than the MCC in two scenarios (TSE and YV-ER), and remains approximately the same in three scenarios (CENSUS, NCVR and MVR).

In order to provide a better illustration of the source selection impact over the linkage quality, we use the results plotted in Figure 5.4 to compare the linkage quality achieved by training the target classifier using the selected source (considering $d_{h\Delta h}$) against the remaining source candidates. The comparison results are depicted in Figure 5.5.

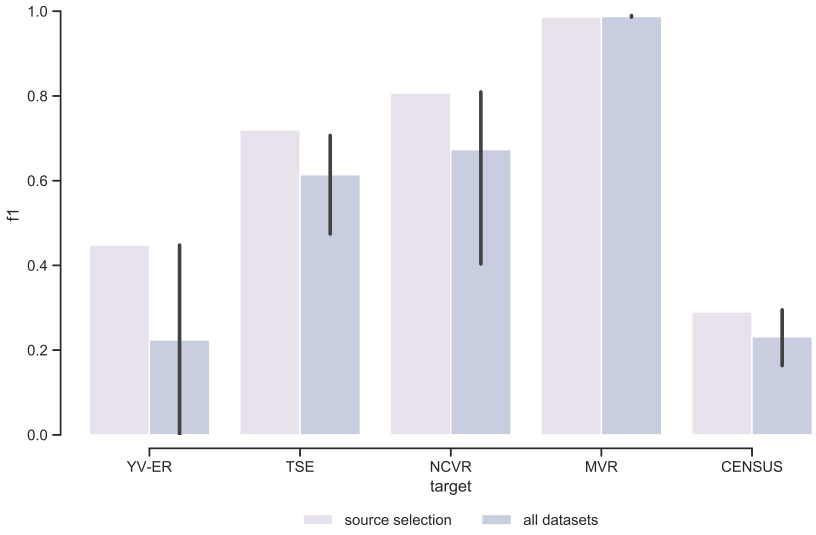


Figure 5.5: An illustration that the source selection step is able to chooses a source dataset that returns a F1 near the maximum possible value acquired by our baseline.

The vertical axis of Figure 5.5 represents F1 whilst the horizontal axis outlines the various target datasets. The colored bars illustrate the selected source dataset results and the mean quality of all remaining source candidate datasets. Notice that we plotted an error bar for all source candidate datasets; the top of this error bar represents the maximum F1 value of all source candidate datasets. Observing the selected datasets and the maximum possible F1, it is possible to establish that the source selection results displayed the same value of the highest F1 of all source candidates. In other words, our source selection approach (detailed in Algorithm 1) presents a F1 value compatible with the maximum F1 observed in our experiments.

The error bars in Figure 5.5 also illustrate a significant F1 variation for all datasets, except for MVR. This variation indicates the relevance of selecting an adequate source dataset for our approach.

Regarding our first research question, our source selection step overperformed the MCC method

and selected the source dataset that presented the highest possible F1 values in every experiment scenario. Thus, based on our experimental results, we can state that our source selection step is able to elect an adequate source dataset in a PPRL system.

RQ.2) Does feature (similarity metrics) selection improve the linkage quality of our approach?

Before we answer this research question, we need to verify the impact of different similarity metrics over the linkage quality. Figure 5.6 plots the F1 achieved by Algorithm 2 for each target dataset, considering the usage of distinct similarity metrics. In other words, for each target dataset, we examine all candidate datasets, varying the number of similarity metrics employed as features.

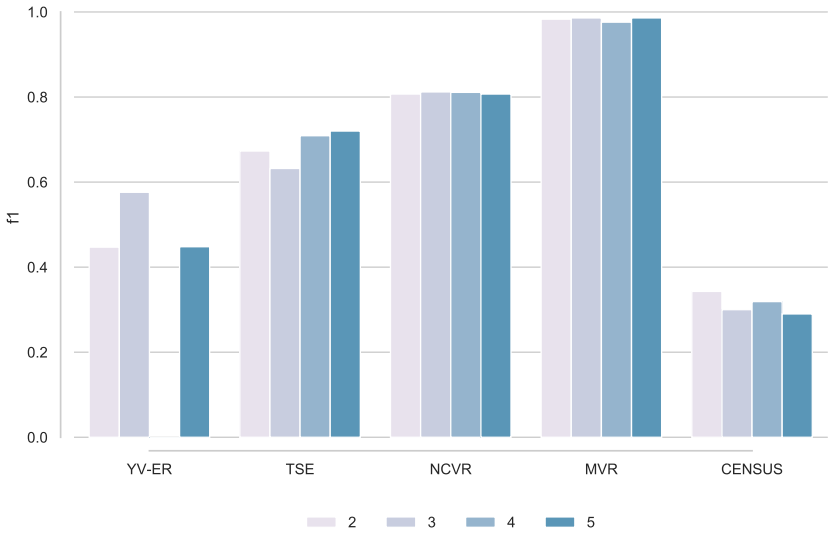


Figure 5.6: Quality (F1) impact of features selection per target dataset.

The horizontal and vertical axes depict the target dataset and F1, respectively. The colored bar represents the result achieved by our approach using different similarity metrics. In other words, we executed the source selection using all candidate datasets and a combination of 2, 3, 4, and 5 similarity metrics as input to Algorithm 2.

In Figure 5.6, it is possible to observe a variation of F1 for each target dataset depending on the number of similarity metrics used as features in our final classifier. It is worthwhile to mention that, for the most challenging datasets to perform the linkage (CENSUS and YV-ER), which presented the highest number of missing values (see Table 5.2) and lowest F1, the usage of specific features allows an improvement in terms of F1. These facts indicate the relevance of selecting adequate similarity

metrics for each source and target dataset [101; 192].

To estimate the impact of feature selection on the overall quality of our approach, we executed Algorithm 3 using the combination of all features (five similarity metrics), such that $\bigcup_{r=2}^{5} C_r^5 = 26$, and compared it (Algorithm 3) against the source selection (Algorithm 2) considering all available similarity metrics as features. Table 5.3 summarizes the results.

target	**source #1**	**source #2**	**selected features**	**gain**		
				f1	precision	recall
MVR	CENSUS	CENSUS	[dice, jaccard, overlap, hamming, entropy]	0	0	0
NCVR	CENSUS	CENSUS	[dice, jaccard, overlap, hamming, entropy]	0	0	0
YV-ER	RESTAURANTS	BOOKS	[jaccard, overlap, hamming]	13%	2%	17%
CENSUS	NCVR	MOVIES	[jaccard, entropy]	5%	6%	-18%
TSE	BOOKS	BOOKS	[dice, jaccard, overlap, hamming]	-1%	3%	-8%

Table 5.3: Feature Selection quality results

Table 5.3 presents the target dataset, the selected source considering all five similarity metrics (source #1), the chosen source dataset considering feature selection (source #2), the similarity metrics employed, and the quality metrics gains.

Regarding the NCVR and MVR target datasets, Algorithm 3 chose the same source dataset with all five similarity metrics as Algorithm 2. Therefore, no changes are observed in the quality metrics. For the remaining target datasets, different source datasets and similarity metrics were chosen by Algorithm 3.

Concerning the CENSUS and YV-ER datasets, it is possible to recognize an improvement on F1 when we employ the feature selection step. Regarding the YV-ER dataset, Table 5.3 exposes an increase in all quality metrics. For the CENSUS dataset, we witness an increase in precision and a decrease in recall. In other words, the selected source was able to train a classifier that produced more accurate predictions (+6% precision). It decreased the number of detected duplicate entities (-18% recall), resulting in a 5% gain in the overall quality (F1), making the target classifier predictions more accurate.

Regarding the TSE dataset, notice that the selected source dataset is the same for the two test scenarios; however, the features are different. The source dataset and features chosen by Algorithm 3 present a reduction of 1% in terms of the overall quality (F1). Nevertheless, we would like to remark that the feature selection step increases the prediction accuracy in terms of precision at a recall cost.

In practical terms, the feature selection step in the TSE dataset and, similarly to the CENSUS and YV-ER datasets, improved the target classifier predictions. In other words, when we trained the target classifier using every similarity metric available, the target classifier was able to recognize

140 entity pairs as duplicated correctly and 95 (40%) pairs wrongly. In turn, when we employed feature selection, the target classifier was able to distinguish 128 entity pairs as duplicated correctly and 50 (28%) pairs wrongly.

Regarding our second research question, the feature selection step increases the overall quality. Furthermore, it makes the target classifier more reliable and identifies the duplicated entities with more precision.

RQ.3) Is AT-UC able to improve the PPRL quality results when compared to the baseline and the competitors?

As detailed in Section 5.3.4, we compare the results of our approach against a baseline and three competitors. As a baseline, we consider the threshold-based classifier. As competitors, we employ the naive-based transfer learning (naive), CORrelation ALignment (coral) [158], and the TransER [88]. Table 5.4 displays the results (Precision, Recall, and F1), the source used to train the model, the target, and the approach.

Before describing the Naive classifier results, we would like to highlight that the reported results represent the best linkage result for every tested scenario since, otherwise, the mean result would be zero for almost every target. Comparing the AT-UC result against the Naive approach, we can observe that our approach overcomes the naive approach for every metric. Furthermore, by observing the precision and recall metrics, it is possible to verify that the naive-based classifier wrongfully marks a large number of pairs as similar due to the high value of recall (near 100% of coverage) and low value of precision in every case.

To evaluate the transER, we employed all possible source datasets because the approach does not have a methodology for selecting a source. However, transER was able to execute only in three cases. Thus, we report the results as best in the source column. AT-UC overcame the results achieved by transER, especially for the TSE dataset.

These results can be explained by the fact that transER considers homogeneous domains (as we do). However, we assume that features represent different similarity metrics of the complete record, while transER considers the similarity metrics of the record's attributes. Moreover, due to this difference, the techniques employed to minimize the different class conditional probability distributions in AT-UC demonstrate more adequate to be employed in a PPRL context.

approach	target	source	precision	recall	f1
	census	restaurants	14%	77%	**22**%
	mvr	census	97%	99%	**98**%
at-uc	ncvr	census	100%	78%	**87**%
	tse	books	84%	87%	**85**%
	yv-er	tse	66%	56%	**61**%
	census	best	1%	100%	2%
	mvr	best	5%	99%	9%
naive	ncvr	best	2%	91%	4%
	tse	best	0%	13%	0%
	yv-er	best	43%	87%	58%
	census	-	-	-	-
	mvr	best	83%	99%	90%
transER	ncvr	best	74%	99%	85%
	tse	best	3%	100%	6%
	yv-er	-	-	-	-
	census	5-best	9% ±6%	49% ±6%	15% ±9%
	mvr	5-best	96% ±1%	88% ±1%	91% ±8%
coral	ncvr	5-best	99%	49%	66%
	tse	5-best	81% ±1%	80% ±1%	80% ±8%
	yv-er	5-best	93% ±15%	40% ±15%	46% ±45%
	census	5-best	8% ±5%	61% ±5%	14% ±9%
	mvr	5-best	49% ±46%	71% ±46%	51% ±44%
threshold	ncvr	5-best	49% ±47%	67% ±47%	49% ±42%
	tse	5-best	27% ±24%	71% ±34%	32% ±31%
	yv-er	5-best	60% ±42%	64% ±42%	60% ±39%

Table 5.4: Instance Alignment Impact

As mentioned in Chapters 2 and 3, the usage of the CORAL in a RL context could be damaged due to the nature of the data distribution of a typical RL process. We report the CORAL results for the five best linkage results because: i) it does not have a methodology to select a source candidate, and ii) if we use the complete set of results, the mean metric values will drop, making it hard to compare with our approach.

The usage of CORAL in non-normal data damages the knowledge transfer [88]. This fact can be seen in the reported error for each target dataset. Notice that there is no error for NCVR. This result was achieved because the CORAL could be executed only for one pair of datasets. Regarding the linkage quality, AT-UC overcame the mean result of the best five results of CORAL.

The most used classification technique in a PPRL context is the threshold-based classifier. To evaluate this classifier, we tested similarity values from .6 to 1 with an increment of 0.5 and reported the five best results. Prior to confronting the AT-UC result against the threshold classifier,

we remark the challenge of guessing the threshold value for each target dataset. In our experiments, each dataset presented a different optimal value. Particularly, for the TSE and CENSUS datasets, the threshold value was sensitive, and a slight increment (or decrement) in the threshold presented an important impact over F1.

Regarding the comparison of our approach and threshold-based classifier, it is possible to recognize that our approach overcomes the mean quality of the baseline and produced results near the optimal threshold, i.e., the error upper bound. Moreover, AT-UC overwhelms the optimal baseline value for the CENSUS, MVR, and TSE datasets.

Finally, we can state that our approach can improve the linkage quality compared to the competitors and the baseline, answering the third research question. Furthermore, AT-UC has other aspects that should be mentioned. Similar to other approaches [158; 88], AT-UC requires minimal parametrization. Moreover, it eliminates the need for an expert to guess a threshold value or select a source dataset employed in the TL approach.

It is worthwhile to mention that all of the competitors were proposed to a typical RL scenario, where the privacy of the data is not an issue, while AT-UC is specifically designed to operate in a PPRL scenario. To evaluate the privacy risk, we employ a state-of-the-art privacy attack, the results and our considerations are reported in Section 5.3.7.

RQ.4) Which are the most adequate classifiers to be used as the separation and the target classifiers?

To answer the previous research questions, we used classifiers that presented the best quality result in terms of F1. Thus, to answer this research question, we first investigate the impact of different ML methods on separation and target classifiers.

Figure 5.7 illustrates the impact of different techniques as the separation classifier (Algorithm 2, line 3). For each target dataset, we vary the ML techniques on the separation and the target classifiers. As a consequence of this experimental setup, an error bar is plotted for each colored bar.

In Figure 5.7, it is possible to observe that the SVM and the Logistic techniques employed as separation classifiers presented the best quality result. Moreover, when we compare SVM against the Logistic classifier, we observe an overlap between the confidence intervals, preventing us from stating that one technique is statistically better than the other [38].

The result achieved by the SVM and Logistic classifiers can be explained by the fact that SVM tries to find the margin that separates the classes (source or target), and this fact reduces the risk of error on the data [72; 73]. In turn, Logistic regression delineates different decision boundaries with distinct weights that are near the optimal point. In other words, Logistic and SVM techniques fit a single

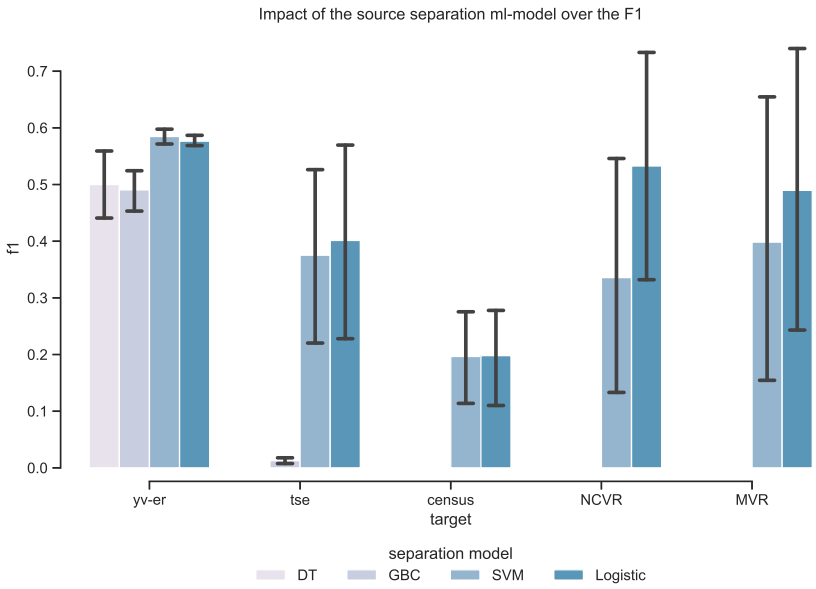


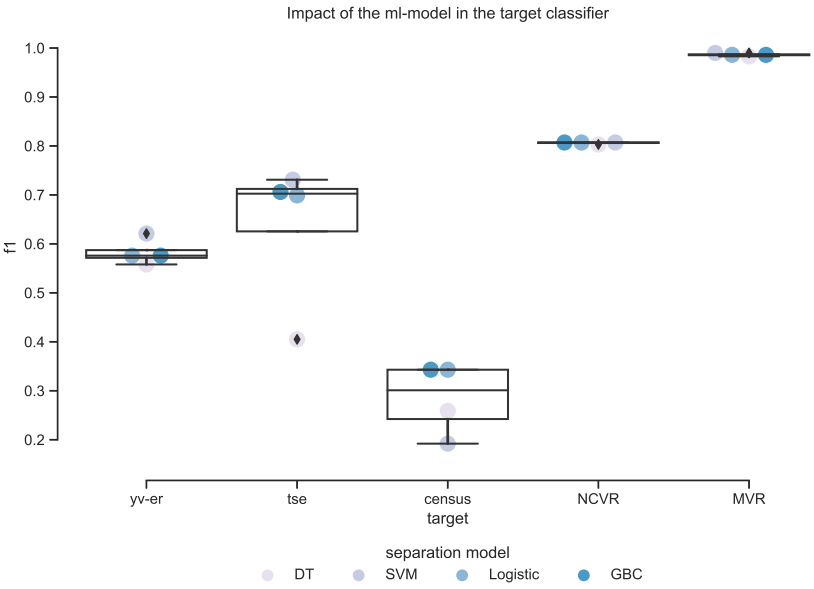
Figure 5.7: Quality results by target and separation classifiers.

line to divide the space precisely into two, whereas DT and GBC bisect the space into smaller regions.

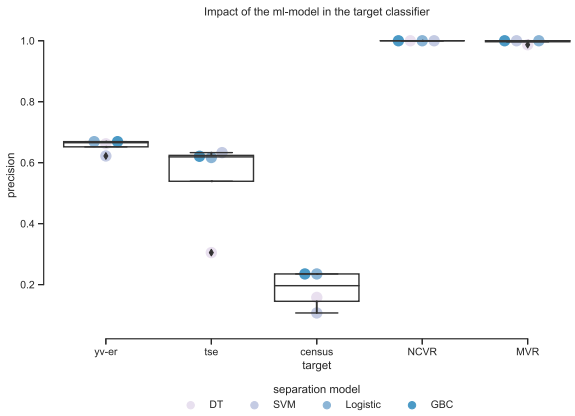
It is worth mentioning that techniques that bisect the decision boundary are more susceptible to overfitting the training data when classifiers have difficulty to separate the classes [73]. We believe this fact explains the quality result achieved by DT and GBC as separation classifier.

Considering the previous arguments, we chose Logistic Classifier as a separation classifier to investigate the impact of different ML techniques as target classifiers. Even though we cannot affirm that there is a statistical difference between the results produced by Logistic and SVM, the Logistic classifier presented an average value of F1 higher than the other competitors, and it is simpler than the SVM.

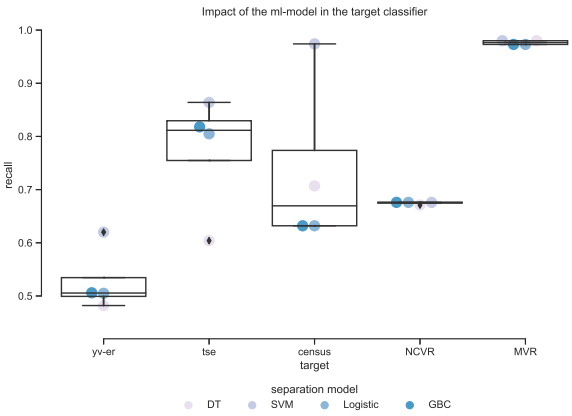
Figure 5.8 illustrates the quality results for various ML techniques as the target classifier, considering Logistic as separation classifier. A boxplot is depicted for the target datasets and, inside each boxplot, we plotted the result of the different ML techniques.



(a) F1



(b) Precision



(c) Recall

Figure 5.8: Quality results by dataset and target classifier.

Observing the NCVR and MVR datasets, notice that the quality result manifests almost the same value regarding the employed ML techniques. This result can be explained by the fact that the NCVR and MVR datasets presented similarity values with a significant difference between the classes (match and non-match). For instance, most of the matching instances have similarity metrics values near 0.9, while non-match instances presented mean similarity values below 0.6. In other words, the NCVR and MVR datasets are easy datasets to perform the linkage.

The use of Logistic, GBC, and SVM as target classifiers produced the best quality results. The SVM technique provides a good quality result for every tested scenario, except for the CENSUS dataset, where it wrongly classifies various instances as matches. This classification error can be perceived by the high recall (all duplicated entities are found) and low precision.

The GBC and Logistic classifiers achieve almost the same quality results for every target dataset. In our implementation, we use an exponential optimized loss function for the GBC that considers an AdaBoost algorithm with an exponential loss function. This exponential loss could explain the similar results achieved by the Logistic and GBC classifiers. Thus, considering the computational cost, quality results, and the Logistic Classifier simplicity, we used it as a target classifier algorithm in our experiments.

Several works indicate that Logistic and SVM are the ML techniques that provide the best linkage results in the RL context [28; 84; 91]. Our work corroborates this statement and suggests that the GBC technique can also supply satisfactory linkage quality in a PPRL scenario.

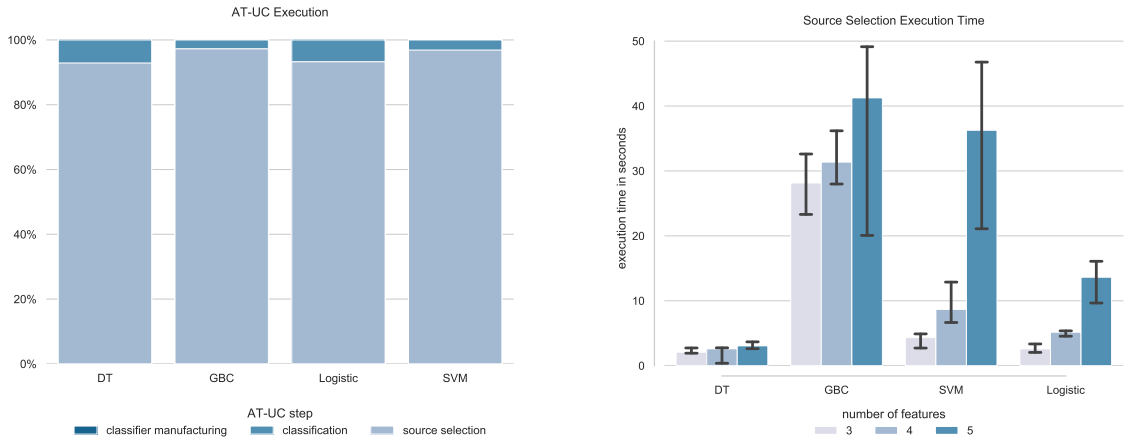
5.3.6 Efficiency Results

In this section, we evaluate the efficiency of AT-UC in a PPRL process. Moreover, we assess the computational cost, in terms of execution time, of AT-UC steps (Source Selection, Classifier Manufacturing and Classification) for each experiment executed in Section 5.3.

Figure 5.9a, depicts the mean execution time for each step of AT-UC considering different separation models (line 6 of Algorithm 2) and classifiers. Notice that, regardless of the separation model, the source selection step consumes 90% (on average) of the execution time. On the other hand, the classifier manufacturing and the classification steps consume (on average) 1% and 9% of the execution time, respectively.

Figure 5.9b shows the execution time of the source selection by the number of features and the

separation model. Notice that DT and Logistic present the shortest execution time while more complex models such as GBC and SVM present the longest execution time.



(a) AT-UC steps execution time.

(b) Source Selection execution time.

Figure 5.9: AT-UC Efficiency Results.

It is worthwhile to mention that the PPRL process execution time usually present a long execution time due to the comparison step of PPRL. For instance, the smallest personal dataset (census) depicted in Table 5.2 took more than 30 minutes to execute the comparison step. Therefore, AT-UC does not represent a significant impact on the PPRL execution time.

5.3.7 Privacy Sketch

As illustrated in Figure 5.2 and presented at the beginning of Section 5.2, AT-UC is executed by a semi-trusted third party, considering an HBC adversary model. Moreover, AT-UC considers as input the similarities of record pairs (i.e., the output of the Comparison step) and the public (or synthetic) datasets employed to train the classifier.

Public datasets are widely available on the Internet. However, it is not common to use these datasets in a privacy attack because it is hard to determine if a public dataset is related to the datasets used in a PPRL process. Therefore, the source candidates sent by the PPRL parties to STTP could make it easy for STTP to identify which public datasets were related to the private dataset and execute an attack.

Moreover, the Source Selection stage could be employed to reduce the search space of an attack performed by a STTP, regardless of the source candidate datasets sent by the PPRL parties. For instance, a STTP could have pre-stored several public and synthetic datasets and use the Source

Selection algorithm to choose the most related datasets to execute a privacy attack. In the following, we present arguments on how a STTP could execute an attack using our approach.

First, we would like to remark that, for the candidate's datasets, the PPRL parties send only the calculated similarity values of the records stored in a public dataset. The PPRL parties do not inform the semi-trusted third party any other details regarding the dataset, e.g., the dataset origin or the attributes (QID) used to calculate the similarity.

To leverage the information presented in the candidate's datasets, STTP must associate the similarity values sent by the PPRL parties to the actual dataset values to execute a privacy attack over the target dataset. This association is not an easy task because the PPRL parties could extract a sample, use privacy-preserving techniques such as Differential Privacy or L-Diversity [107] over the public datasets or use a privacy-preserving data generator [137] to create a custom syntactic dataset.

Our privacy sketch assumes that a STTP can associate the similarity values with the raw dataset pair. Moreover, we assume STTP can precisely identify the attributes used to generate the record.

In order to measure the privacy risk of our approach, we employ a privacy attack that uses similarity values of a known (source) dataset to re-identify entities from an unknown (target) dataset [180]. This attack employs the similarity values of both datasets (source and target) to generate a pair of graphs, where entities and similarity values are converted to nodes and edges. Then, these graphs are aligned using several Graph and Machine Learning-based techniques to identify entities with the same neighborhood (nodes and vertices). The results reported in [180] demonstrated an accuracy superior to 95% for the NCVR dataset.

We use the NCVR and MVR datasets because they presented the highest number of common attribute values to maximize the chance of success of the attack. We extracted a sample with 10,000 records from the datasets and executed the attack over the attributes firstname, lastname, and middlename.

In the initial result, we used all three attributes and executed the attack. As a result, the attack was unable to re-identify any entities. We did not expect this outcome. Thus, we used two attributes (firstname and lastname) and still obtained the same outcome. No entities were re-identified.

In order to properly evaluate the privacy risk, we used a single attribute (firstname) of the dataset to provide a vulnerable scenario. It is worth mentioning that it is strongly advised not to use only one attribute in the RL task because the low variability in the data is expected to have a high number of false-positive matches, compromising the linkage quality. In this scenario, in which one attribute is employed as QID, the attack could re-identify 139 (1,4%) entities of 10,000.

Notice that, to re-identify entities using a state-of-the-art attack, we had to use two datasets with a

similar set of attribute values. However, AT-UC considers only the similarity values; AT-UC is context agnostic. Therefore, it can use a source dataset with no common attributes (or attribute values) to the target dataset, making it hard for an attacker to use the information stored in the source candidates to attack the target dataset. For instance, for the CENSUS and TSE datasets (Table 5.4) AT-UC uses non-personal datasets as sources (Restaurants and Books). In these cases, it is unlikely that a book title and restaurants share the same attributes as a person in a PPRL process.

In summary, the usage of external datasets can increase privacy risk for PPRL. Even if PPRL does not use our approach, an attacker can employ our source selection stage to select external (e.g., public) datasets that are most suitable to execute an attack over the target dataset. Knowing these implications, the PPRL parties should consider the adoption of well-known actions to mitigate re-identification risk, such as i) avoid conducting the linkage process considering few attributes, ii) choose proper anonymization parameters, and iii) use an anonymization technique that provides Differential Privacy guarantees.

5.4 Summary

In this chapter, we have presented AT-UC, which enables the usage of ML-based classifiers in the PPRL process. AT-UC uses public (or synthetic) datasets to train a classifier to identify matching records pairs of private datasets. The use of AT-UC simplifies the PPRL process by employing a ML-based classification step, mitigating the problem introduced by the manual threshold selection. AT-UC was evaluated experimentally using several real-world data sources regarding efficiency and effectiveness (quality). The results demonstrated that AT-UC is able to overcome the quality of the most used technique (threshold) in the PPRL context.

It is important to mention that our contribution assumes that among public datasets exists a suitable dataset that can be used as a source dataset. However, we are aware that may not exist public datasets suitable for every PPRL scenario. Thus, to overcome this limitation, in the following chapter we investigate the usage of a Deep Learning-based classifier in the PPRL context.

Chapter 6

Deep Learning-based Classifiers for PPRL

Currently, data has been used in several domains, from government policies to business analytics [30]. Furthermore, services and enterprises use data as the primary input for their products and services. For instance, governments use the citizens' populational and medical information to define and fund the public health policy [30; 176]. Regarding the data as a business, companies merge multiple public/private datasets into a unique database to validate the consumer registration data to speed up the customer acquisition process and improve information quality. This process is named customer on-boarding service [147].

Moreover, due to the laws and regulations, the aforementioned data usage (e.g., on-boarding services and public policy) needs to be carried over private data. Thus, enforcing governments and private companies to employ privacy-preserving technology to preserve the privacy of the data (e.g., customers or populations). In this context, PPRL is used to identify entities (e.g., personal records) across multiples dataset that do not have a unique identifier (e.g., social security number). It is worth remarking that the PPRL process decides whether a record pair is a match using the similarity value, calculated by comparing the anonymized/encoded QIDs values [30].

In order to classify the record pairs, most of the PPRL solutions consider a simple threshold classifier. In summary, the PPRL parties elect the threshold value (which usually vary from 0 to 1) and compare it against the similarity value of each record pair. Notice that the threshold value definition is a complex task that requires expert operators to ”guess” the appropriate value. For instance, if the threshold value is too high (e.g., 0.9 or 1), PPRL will miss true match entities. On the other hand, if this value is too low, PPRL will likely classify false positive matches.

Several works have been proposed to improve the linkage quality in the PPRL context, including the usage of Machine Learning [25; 27] and Transfer Learning [126] techniques. These techniques preponderantly employ statistical machine learning models (e.g., the Support Vector Machine and Logistic classifier) over the record pairs' similarity values to classify whether or not the records are a match.

Moreover, according to recent work [105; 123], similarity measurements are dependent on the task (e.g., RL) and the data itself. For instance, by comparing events that happen across time (dates), we can select syntactical or chronological similarity measurements depending on the task performed. However, in the PPRL context, the PPRL solutions employ limited syntactical similarity measurements, such as Jaccard, due to the binary characteristics of the encoded records [30; 113; 177]. Thus, due to the limitation imposed by PPRL (e.g., binary similarity measures and privacy restrictions), it is hard to create/employ classifiers for PPRL.

In the RL context, where data privacy is not an issue, Deep Learning techniques were employed to mitigate the similarity measures bias [41; 46; 75; 91; 105; 121]. In summary, Deep Learning techniques were employed to perform linkage tasks using the non-linearity characteristics of unstructured data and other complex data structures. In other words, these techniques were employed to detect patterns that could indicate whether or not an object pair is a match or not. Moreover, the aforementioned works demonstrated that Deep Learning could improve the linkage quality when employed in non-tabular data, such as the PPRL encoded.

To our best knowledge, no work directly explores the non-linearity provided by the PPRL inputs - encoded records. In other words, in the PPRL context, the existing decision models rely upon an indirect similarity measure instead of the actual data. In order to explore the non-linearity of the encoded data in the PPRL context, we propose a novel methodology to compare encoded records in this thesis. The basic idea of the proposed methodology is to perform the record pairs classification without using predefined similarity metrics (e.g., Jaccard and Dice). In other words, the proposed methodology leverages Neural Networks (NN) and sophisticated statistical learning methods (e.g., SVM and Gradient Boosting) to learn patterns that could indicate whether or not an encoded record pair is a match.

Besides the aforementioned challenges of PPRL, the encoding/anonymization step presents itself as a complicating element to the use of raw encoded data (e.g., the Bloom Filter value of two records) as input to classifiers. Moreover, from the perspective of the classifier, the encoding process can be seen as a dimensionality reduction with noise addition. Furthermore, the dimensionality reduction happens when we transform a 128-bits word (considering UTF-8 encoding and six letters words) into 12 bits in one BF, losing the syntax and semantic representation of the word. In turn,

the noised addition occurs due to the hash collision and the hardening techniques employed in the PPRL encoding step.

Explaining the argument

The following example illustrates the dimensionality reduction of the information performed by the PPRL encoding step. Assume the we have to compare the sentences: "The mayor removed the bus top" and "The city authority moved the bus terminal". Using the plain sentences, we can employ the semantics of "city authority" and "mayor" to determine that these sentences are related. However, in PPRL, we need to transform this sentence into corresponding bit arrays and calculate their similarity - e.g., 0.50. This makes it hard for a classifier, using only the record pairs' similarity values, to decide whether these sentences are related, or not.

To mitigate the problems caused by the encoding technique, we propose the use of Recurrence Plot (presented in Section 6.1.1) to reencode the PPRL data. In other words, Recurrence Plot (RP) is employed to map PPRL encoded data (e.g., Bloom Filter) as a complex dynamical system [112] in order to evidentiate patterns that can be used by classifiers to decide whether record pairs are a match or not. Therefore, by representing encoded record pairs as RP, we are able to train classifiers using the encoded data. Moreover, we can use deep learning techniques to explore the hidden patterns and the non-linearity characteristics of the information held by the encoded representations of the records. We tested our approach using the PPRL's most common encoding technique, the Bloom Filter (BF) - detailed in Chapter 2.

In Chapter 5, we propose the AT-UC approach that employs Transfer Learning (Instance-based) to build classifiers for the PPRL process. AT-UC was conceived assuming that a suitable source dataset exists for every target dataset. However, in PPRL applications, it is possible that there is no suitable source dataset. To mitigate this problem, we make the following contributions: (i) we propose a novel representation of the records pairs for comparing PPRL anonymized/encoded data; ii) we propose and test several deep learning architectures, considering RPs in the PPRL context; iii) we evaluate our contributions using real-world datasets considering different parameter settings, and we compare our method with existing classification (hardening) techniques regarding linkage quality.

6.1 Deep Learning-based Classifiers

This section describes how to use an alternative representation of Bloom Filter pairs that enables a novel comparison methods in the PPRL context. First, we detail how to represent BF pairs as

RP. Then, in the subsequent section, to validate the use of RP in the PPRL context, we propose a methodology to incorporate RP in a PPRL workflow.

It is worthwhile to mention that, by representing BF pairs as Recurrence Plots (dynamical system), we are able to explore new classification strategies in PPRL. In other words, instead of using the similarity value between encoded record pairs, we use RP's to train a classifier (decision model) and employ this classifier to perform the linkage. Notice that this workflow is a secondary contribution employed to enable and assess our main contribution, i.e., the use of RP within the PPRL context.

6.1.1 Recurrence Plot as a Feature to Train PPRL classifiers

As presented in Chapter 2, Recurrence Plot (RP) is a well-known and widely used technique to explore and reveal a pattern in data. RP has been used in several contexts, from time series analysis to medical applications [112]. In this section, we discuss the use of RP to uncover patterns in pairs of encoded records (e.g., BF pairs). In other words, we use RP to highlight patterns in encoded record pairs that could indicate whether a record pair is a match, or not.

> **Explaining the Idea**
>
> *The use of RP to represent pair of encoded records intends to replace the standard similarity measures (e.g., Jaccard and Dice) by the RP. Furthermore, the RP aims to highlight patterns that will be employed as features to the PPRL Classification step. Moreover, the use of RP to represent record pairs will allow Machine-Learning techniques to learn from the encoded records, reducing the bias introduced by the standard similarity measures [105]. In summary, RP will enable the use of Deep Learning techniques (i.e., Neural networks) in the PPRL context.*

Before explaining how to employ RP within the PPRL context, first, we need to present assumptions regarding the PPRL encoded data. We assume that such data is represented by a 1-dimensional array of bits or integers. In our experiments and the formalization presented in this section, we consider a 1-dimensional array of bits, such as the Bloom Filter technique. Thus, each position of the encoded data array ($\hat{e}$) represents a state that needs to be mapped by RP, such as:

$$\hat{e} = [x_1, \cdots, x_l],$$

where x_i represents the value (bit) in the encoded data position whilst l is the length of the encoded data.

In a typical PPRL process, pairs of encoded data are used in the Comparison step. Thus, to highlight the states (encoded data patterns) that simultaneously occur in both records, we must operate a bivariate Recurrence Plot [112]. In this sense, we employed the Cross Recurrence Plot (CRP) as our bivariate technique because it can show which states in one dynamical system occur simultaneously in a second dynamical system [47]. Consequently, the aforementioned CRP characteristic could be employed to detect regions of the encoded data (i.e., BF segments) that coincide.

Explaining the Idea

CRP will be employed to identify similar regions of the encoded records. In other words, the CRP highlights common states (e.g., bits) over encoded record pairs. Moreover, these regions (the RPs) will be forwarded to a ML-based classifier to extract patterns that could identify matching records.

In Section 2.8, we presented a generic CRP formalization (Equation 2.7). In order to provide further details of the CRP in the PPRL context, we introduce a modified version of CRP. Equation 6.1 formalizes CRP using two encoded records ($\hat{e}_1$ and $\hat{e}_2$) and m neighbors. The m-neighbors is used to create and delimitate an area around each bit position of the encoded record that the Heaviside function will compute. For a graphical example of the m-neighbors, see the red dashed areas of Figure 6.1

$$CRP(\hat{e}_1, \hat{e}_2, m, \epsilon) = \sum_{i=0}^{l_1} \sum_{j=0}^{l_2} \Theta \left(\alpha, \sum_{w=0}^{m} ||\hat{e}_1[i+w] - \hat{e}_2[j+w]|| \right) \tag{6.1}$$

where l_1 and l_2 are the lengths of the encoded records, α is a threshold distance employed in the Heaviside function (Θ), and $||\hat{e}_1[i] - \hat{e}_2[j]||$ is the distance between two sets of elements of the encoded records $\hat{e}_1[i]$ and $\hat{e}_2[j]$.

Notice that CRP will generate a 2-dimensional matrix, $CRP_{i,j}$. In a typical PPRL encoding process, all parties encode their data using the same number of bits (l). Thus, we can calculate the $CRP_{n,n}$ dimensions using l and m. Equation 6.2 show the formula.

$$n = l - (m - 1) \tag{6.2}$$

As disclosed in Equation 6.1, CRP will iterate over the elements (e.g., bits) of the encoded records and compute the distance of these elements considering m neighbors. Later, this distance will be compared against a threshold (α) in a Heaviside function. In Equation 6.3, we define the Heaviside function used in our solution.

$$\Theta(\alpha_i, v) = \begin{cases} 1: & v \leq \alpha_i \\ 0: & v > \alpha_i \end{cases} \tag{6.3}$$

To illustrate the use of CRP over encoded data, consider the encoded records of ANA and ANE (depicted in Section 2.1.4). In Figure 6.1, we illustrate the CRP encoding process for the BF ($l = 8$) of ANA and ANE. Moreover, in the following example, we consider three neighbors ($m = 3$) and the Heaviside threshold of one ($\alpha = 1$).

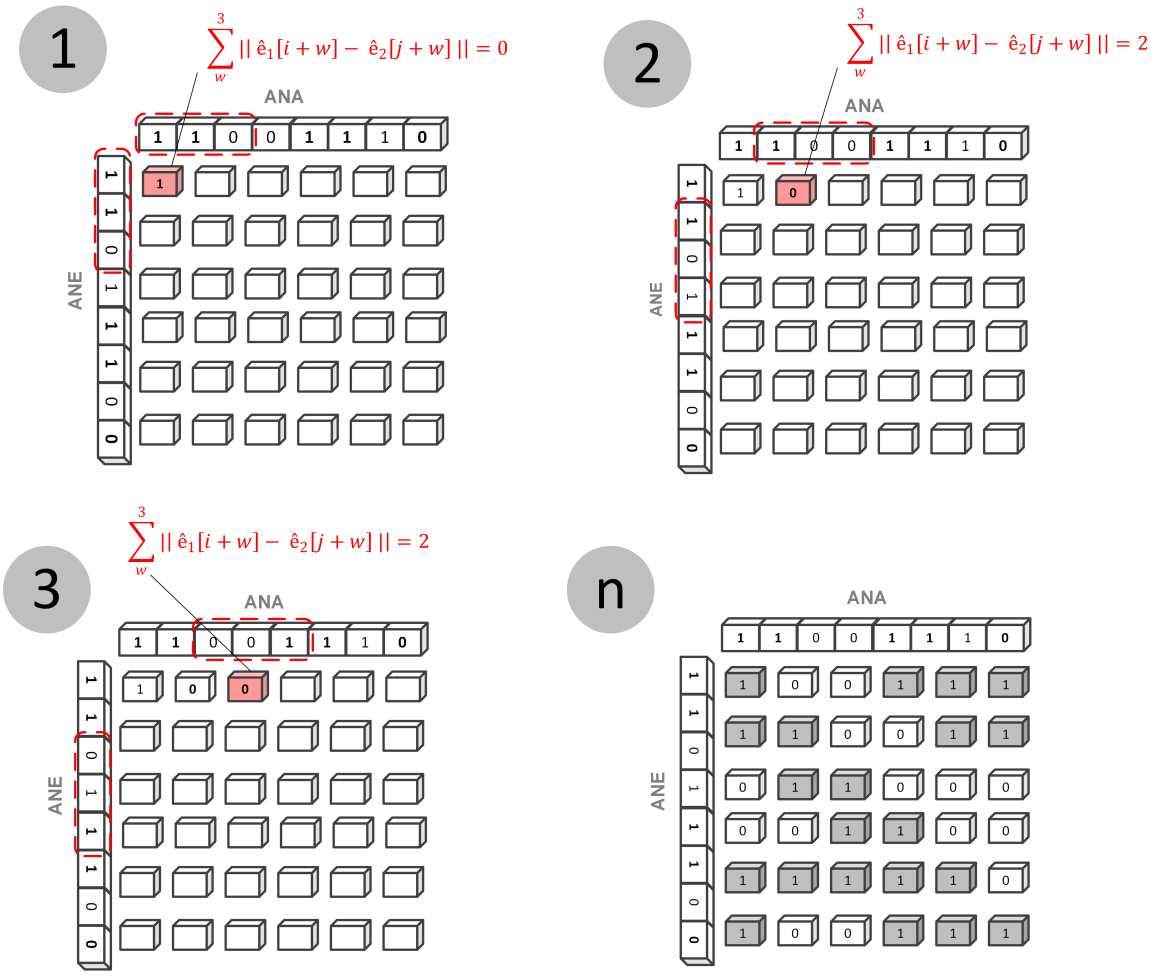


Figure 6.1: CRP encoding process

The CRP is represented as a 6×6 matrix. This dimension is defined by the filter length and the number of neighbors (Equation 6.2, $n = 8 - (3 - 1)$). Each element in the matrix is calculated using Equation 6.1. The Heaviside function ($\Theta(\alpha, \sum_{w=0}^{m} ||\hat{e}_1[i+w] - \hat{e}_2[j+w]||)$) in the last segment of Equation 6.1 is employed to define whether a filter position represents a co-occurrence in both

encoded records, or not. In other words, the first segment of the equation ($\sum_{i=0}^{l_1}\sum_{j=0}^{l_2}$) iterates over each position of the $CRP_{i,j}$, while the second one (the Heaviside function) marks the position with 1 or 0 if a region of both filters indicates a co-occurrence (or not), respectively.

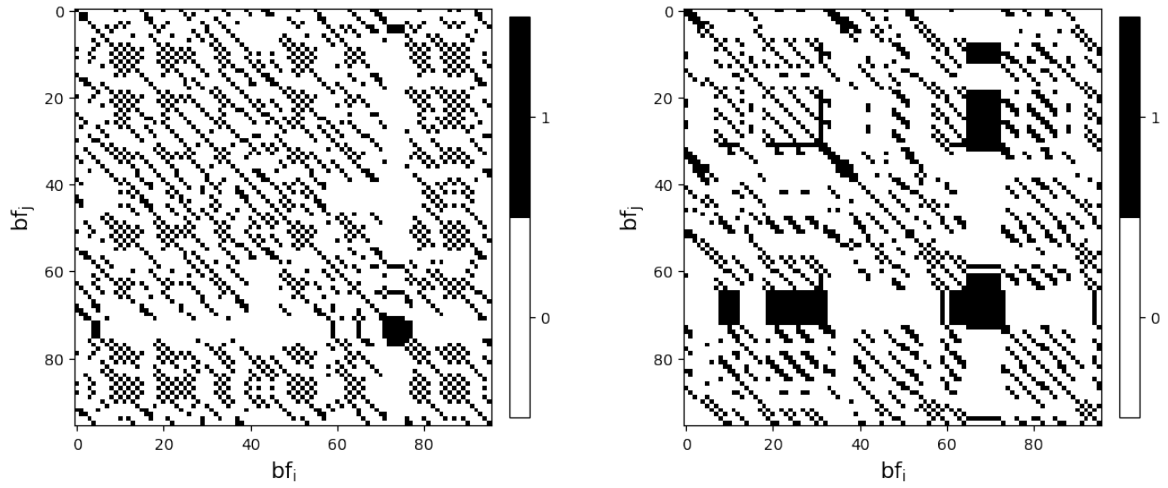
The first step of Figure 6.1 illustrates the computation of the element $CRP_{0,0}$. The red square illustrates the neighborhood (m) used to define whether a bit position co-occurs in both filters. Notice that the distance between the regions is calculated by the $\sum_{w=0}^{m}||\hat{e}_1[i+w] - \hat{e}_2[j+w]||$. This information is highlighted in red. The calculated distance (Manhattan distance) was zero for the first step because the regions are identical. This distance is compared against the Heaviside threshold ($\alpha = 1$) and the region is marked as co-occurrence in the CRP.

The process is repeated for every element of the CRP matrix, and the position is marked accordingly. For instance, in the second step, the distance (2) is greater than α. Therefore, the Heaviside function marks the region as zero. Finally, at the end of the CRP generation, the matrix will be filled with zeros and ones, evidencing similar regions in the encoded data.

By representing the encoded record pair as a CRP, we expect to highlight the local similarity/difference of the data. Moreover, following the work of Recurrence Quantification [112], a CRP of two systems (encoded data in our context) is represented as a series of parallel segments in the matrix. The frequency and length of these lines are related to a certain similarity between both encoded data. Thus, by employing CRP over PPRL encoded data, we expect similar encoded data (e.g., matching records) to generate a particular pattern (e.g., several bold and continuous diagonal lines). Figure 6.2 shows examples of a CRP of matching encoded data and non-matching encoded data.

Notice that the matching encoded data exhibits the expected pattern while the non-matching encoded data presents fewer diagonal lines and several black squares. It is worthwhile mentioning that several numeric methods, analogous to the binary data similarity metrics, have been previously proposed [112]. For instance, the RP can be assessed by its: i) diagonal, vertical, and white vertical lines frequency distribution, average length, longest length, and entropy; ii) pattern recurrence rate; iii) determinism; iv) divergence; v) laminarity; and vi) laminarity determinism ratio.

In this work, instead of using the previously mentioned numeric methods, CRP will be classified by Machine Learning-based classifiers. These classifiers will use the CRPs as input to the ML models. Instead of using a similarity value, or a small set of similarity values [126], as features, we will employ a 2-dimensional matrix of features, where the classifiers will look for partners that indicate matching encoded records. Moreover, instead of finding a decision boundary (line or hyperplane) based only on similarity values, our work employs CRP as input to extract patterns representing matching records. In Section 6.1.2, we detail a workflow that considers CRP in a PPRL process.



(a) Matching

(b) Non-matching

Figure 6.2: CRP representation of two encoded record pairs. The records were encoded considering a $l = 100$, $k = 5$, and a false positive rate of $.5$. Figure 6.2a illustrates a matching pair with 0.9 of similarity, while Figure 6.2b exemplifies a non-matching pair, with a similarity value smaller than 0.7.

6.1.2 MHT Workflow

This section presents a simplified workflow to incorporate CRP in a PPRL solution. Before introducing the workflow, first, we need to raise assumptions and constraints regarding the components of the PPRL and workflow functionalities. The workflow was conceived to be employed in a Semi-Honest (or HBC) adversary model. Moreover, we also consider the existence of a Semi-Trusted Third Party (STTP), which will carry out the classification step of PPRL. Furthermore, the MHT Workflow is entirely executed by the PPRL party (data owner). The only new information to STTP are the CRP encoding parameters and the trained classifier.

We also assume that the $CRP_{i,j}$ of the encoded record provides a matrix with $n \times n$ (detailed in Section 6.1.1) elements for each encoded record pair. Moreover, a classifier trained with the $CRP_{i,j}$ can recognize patterns that indicate whether a record pair is a match or not. It is worth mentioning that the existing ML classifier in the PPRL context employs a low-dimensional feature space (i.e., a similarity value); therefore, the existing ML classifiers are focused on finding an optimal threshold similarity value due to the limited information available to the classifier. Furthermore, regarding data privacy, the workflow was designed to provide as little additional information as possible to a malicious party. In other words, the workflow uses information already available to the STTP, considering

a standard HBC PPRL process. Moreover, the workflow can be used along with Differential Privacy tools, such as the privacy-preserving SGD (Section 2.7).

Explaining the Idea

The primary purpose of the CRP in the MHT Workflow is to provide an ML-based classifier with labeled instances to train a classifier to distinguish matching instances using patterns within the CRPs. Moreover, the use of the CRP during the classification step of PPRL aims to extract the patterns (over the unlabeled data, testing data) and classify them using the previously trained data.

In order to create the classifiers (models) mentioned above, the proposed workflow is divided into three phases: i) training data generation, ii) CRP encoding, and iii) model training. Figure 6.3 depicts the workflow.

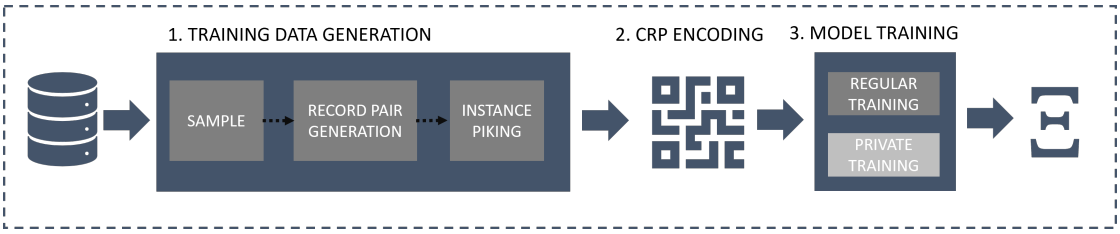


Figure 6.3: MHT Workflow

In the data generation phase, first, we extract a random sample with a fraction (Υ_s) of the original dataset length, $0 < \Upsilon_s < 1$. This sample is divided into two sub-datasets ($\bar{D}_a$ and $\bar{D}_b$) during the record pair generation phase. It is important to remark that we control the percentage of matching records (Υ_m) presented in sub-datasets. In the last step of the data generation phase (instance harvesting), we extract record pairs (instances) that could provide better insights for the classifier to reduce the number of training instances.

During instance harvesting, the similarity (e.g., Jaccard) between each record pair in the sub-datasets $\bar{D}_a$ and $\bar{D}_b$ are calculated to select the non-matching examples that present the highest similar values (Υ_h). For example, assume that an instance (a1 $\in \bar{D}_a$) is compared against the b1, b2, and b3 non-matching instances (such that, $[b1, b2, b3] \in \bar{D}_b$), resulting in similarities of 0.1, 0.4, and 0.6, respectively. Considering the instance harvesting with $\Upsilon_h = 33\%$, only b3 will be employed as a non-matching example. In other words, in this step, we select the closest non-matching instances (i.e., those ones that are harder to distinguish) to the matching instance. Moreover, we expect to produce a high-quality classifier with few record pairs by performing this selection of training examples.

In the encoding phase, the output of the data generation phase (the selected record pairs) will be converted to the CRP representation using the Heaviside function threshold (α) and the number of neighbors (m), defined by the PPRL parties. Finally, the classifier will be trained over the CRPs. Notice that, in Figure 6.3, the model training phase has two possible training methods: one that does not consider the privacy of the classifiers and the second that uses a Differential Privacy technique to mitigate privacy attacks through the trained models. Algorithm 4 formalizes the complete workflow process.

Algorithm 4: MHT Workflow

input : $\mathrm{D}_{raw}, \Upsilon_s, \Upsilon_m, \Upsilon_h, \alpha, m, l, n, \Xi_0, \epsilon$

output: Ξ

1 $D_{sample} \leftarrow$ `random_sample` $(\mathbf{D}_{raw}, \Upsilon_s)$

2 $\bar{D}_a, \bar{D}_b \leftarrow$ `pair_generation` $(\mathbf{D}_{sample}, \Upsilon_m)$

3 $T_i \leftarrow$ `instance_harvesting` $(\bar{D}_a, \bar{D}_b, \Upsilon_h)$

4 $CRP_{i,j} \leftarrow$ `crp_encoding` (T_i, α, m, l, n)

5 $\Xi \leftarrow$ `trainig` $(CRP_{i,j}, \Xi_0, \epsilon)$

The input of the MHT Workflow include: i) PPRL party database ($\mathbf{D}_{raw}$), ii) random sample length (Υ_s), iii) pair generation parameter (percentage of matching records Υ_m), iv) instance picking parameters (Υ_h), v) CRP encoding parameters (α, m, l, n), vi) the untrained classifier (Ξ_0) and vii) the privacy budget of the training parameter (ϵ). The output of the Algorithm 4 is the trained classifier Ξ. It is important to mention that the workflow is executed by the PPRL parties. In turn, the STTP only receives the trained model Ξ.

The first three lines of Algorithm 4 apprise the training data generation (detailed in early in this section). Yet, it is important to remark the computational cost of instance harvesting (line 3). This method presents quadratic computational cost considering the sample length (Υ_h). It is worth mentioning that, in our experiments, this method was executed over small samples ($100 \leq \Upsilon_h \leq 10,000$) and, therefore, does not represent the most costly operation in the workflow.

The most costly method is $crp\_encoding$ method (line 4). In Equation 6.1, it is possible to observe that the CRP encoding has an asymptotical cost of $O(n^2)$ in terms of the original encoding length (e.g., the Bloom Filter length l). Also notice that the $crp\_encoding$ method is $O(n^2)$ iff. $m \ll l$, if $m \equiv l$, otherwise the method will present an asymptotical cost of $O(n^3)$. Moreover, the encoding will be executed for each instance in T_i. This computational cost is partially mitigated by the instance harvesting method. Yet, the PPRL operator must be aware of the involved computational cost, mainly because STTP will employ this method to compare and classify each record pair. Thus,

considering the computational cost of the CRP encoding, it is strongly recommended to use filtering and blocking techniques in the PPRL blocking step to avoid unnecessary comparison by the STTP [30]. It is important to remark that blocking and filtering techniques are not in the scope of this work.

Finally, in line 5, the model (Ξ_0) will be trained over CRP (line 4) considering (or not) a privacy budget (ϵ). If $\epsilon > 0$, Ξ_0 will be trained using optimizers that consider the privacy (as presented in Section 2.7). In turn, if $\epsilon = 0$, no privacy constraint will be executed during the classifier training. At the end of the workflow, the PPRL parties send their trained classifier Ξ and the encoded/anonymized records to the STTP.

6.1.3 Privacy Considerations

In the PPRL process, the encoded records (e.g., BF representation of the records) are always sent to STTP, considering a three-party protocol. In the MHT Workflow, one new piece of information is sent to the STTP: the trained classifier (Ξ). As presented in Section 2, Ξ could represent a privacy risk. For instance, a STTP could use the model parameters to reveal the training data.

To mitigate the aforementioned attack, the MHT Workflow is compatible with Differential Privacy mechanism to train the classifier. This mechanism distances the classifier parameters to the training data at linkage quality cost, making it hard for attackers to use the model parameters to re-identify the instances used to train the classifier. Therefore, it provides privacy guarantees according to a customized privacy budget (ϵ). Furthermore, the data (the encoded record) employed to train the classifier is available to STTP. In other words, even if an STTP could infer the training data from the classifier, STTP will not be able to acquire new information. However, in future work that generates or uses an external dataset to train the classifier, Differential Privacy is needed.

Nevertheless, in the PPRL context, most ML-based classifiers employ statistical learning classifiers (e.g., Logistic Classifier and Support Vector Machine) and do not consider Differential Privacy over the trained models [30]. It is worthwhile to mention that our workflow is compatible with the Differential Privacy training mechanism, mainly because the CRP encoding enables the use of deep learning and Decision tree classification mechanisms [58], which are compatible with this training method. The Differential Privacy quality impact is reported in our experiments, Section 6.2.

6.1.4 Neural Network Architecture

Existing PPRL ML-based classifiers rely on standard binary similarity measures to classify whether an encoded record pair is a match or not [30; 177]. Moreover, a classifier that relies only on similarity

measures needs to define a decision boundary. In this context, there are several well-known statistical learning approaches, such as SVM and Logistic Classifier [25; 126]. However, within the PPRL context, to our best knowledge, no classifier was explicitly designed to recognize patterns. This fact was reflected in our experiments when we employed a state-of-the-art CNN for pattern recognition (ResNet50 [191]), and it was unable to learn the pattern using the available training data.

This section presents the CNN architectures to be employed as a classifier in a PPRL context that uses the CRP representation for the encoded data. Section 6.1.4 introduces a CRP Convolution Neural Network (CCN). CCN considers as input the CRP of two encoding records. Moreover, to assess the CRP encoding impact on the linkage quality, we created a competitor that uses the raw encoding data (e.g., the bit array of a BF) as input data. Notice that the CCN is also a secondary contribution to validate the use of CRP in the PPRL context. In the following section, we present the network architecture previously mentioned.

Deep Learning Network Architectures

Deep learning (connectionist architectures) has been studied for more than 70 years [73]. However, most deep learning architectures were built/proposed on empirical effort [62]. The architectures presented in the following section are not different. We detail the insights used to build the network architectures in each section. All networks presented in this section are based on several empirical tries.

CRP Convolution Neutral Network (CCN)

Convolutional neural networks are often employed for pattern recognition tasks, such as object and face detection [64], because of their capacity to extract important features in high-dimensional data [73]. Due to this capacity and the simplicity of the CNNs, we use this network to implement a classifier that is able to recognize patterns in CRP of encoded data.

Before presenting the architecture of our CCN, we must first detail its input. The input of the CCN is a CRP with $n \times n$ (width, height). It is important to remark that each CRP represents two encoded PPRL records. It is worth remarking that the classification label is generated in the training data generation phase of the MHT Workflow and passed to the CCN at training time as one-hot encoded vectors.

Figure 6.4 illustrates the CNN architecture. The CNN is divided into two parts: the convolutional layers, to extract the features of CRP, and the classification layer, to perform the classification of the CRP based on the extracted features.

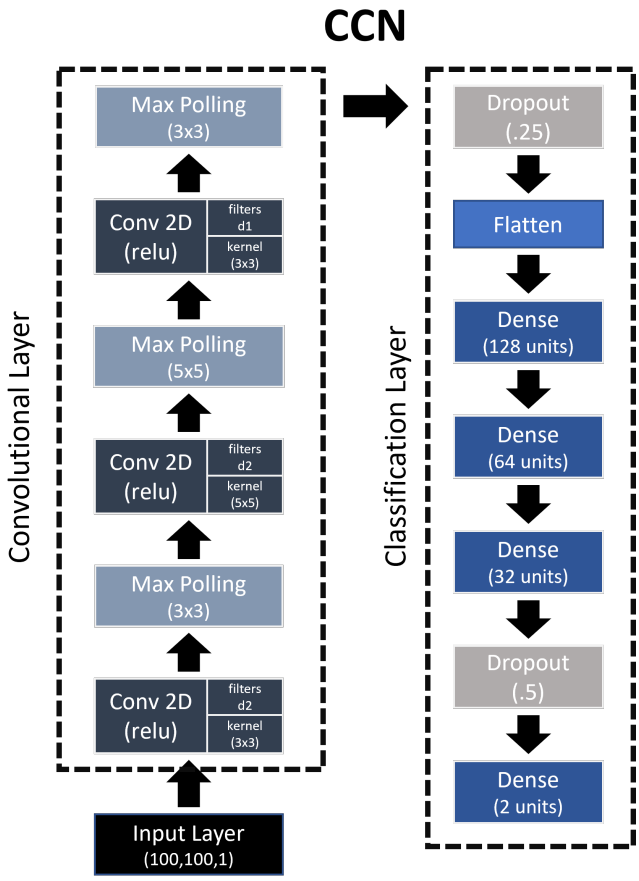


Figure 6.4: CRP Convolution Neutral Network (CCN)

The input layer passes the CRP to three consecutive convolutional layers, where max pooling and dropout (p = 0.25) are applied. The number of kernels for the convolutional layers was set to d1 and d2. The parameters d1 and d2 are defined according to the dimension of the CRP, such that $d1 = \frac{n^2}{5} \times 2$, and $d2 = \frac{n^2}{5} \times 3$ where n represents the width and height of CRP.

Explaining the Idea

The core insight of this network is to take the CRP and extract local features (i.e., the CRP diagonal lines) in the first convolutional layers and then combine the local features to create compound features (e.g., patterns recurrence rate, determinism, divergence, and laminarity) in the final layers. In summary, each convolution filter learns as much as possible from this feature map.

Pooling layers

We use the Pooling layers to reduce the size of the output (feature maps) of each convolutional layer, preserving the most relevant features. [62; 73]

Next, the data is flattened and passed through the classification layers. The classification is a fully connected layer of 128, 64, and 32 units with dropout (p = 0.50). The activation functions of the convolutional layers and the fully connected layer were ReLUs. Finally, a fully connected layer with two units (matching and non-matching labels) and a softmax activation function are used to produce the final output, i.e., the probability for each classification label.

Siamese Convolution Neutral Network for PPRL (SCN)

To demonstrate that CRP can improve the linkage quality, we propose a competitor to CCN that does not use the CRP representation of encoded record pairs. Instead of employing the CCN, we propose a Siamese Neural Networks (SNN) that utilize the raw encoded records as input.

Bromley et al. [18] introduced Siamese Neural Networks to verify signatures on credit cards. However, SSN has been used in many different areas, from one-shot learning to textual and facial recognition tasks [24; 90; 123]. As SNN perform well in areas where similarities between different entities need to be evaluated [123; 105], we opt to test this type of network to address the task of duplicate detection over encoded data. Figure 6.5 illustrates the network architecture.

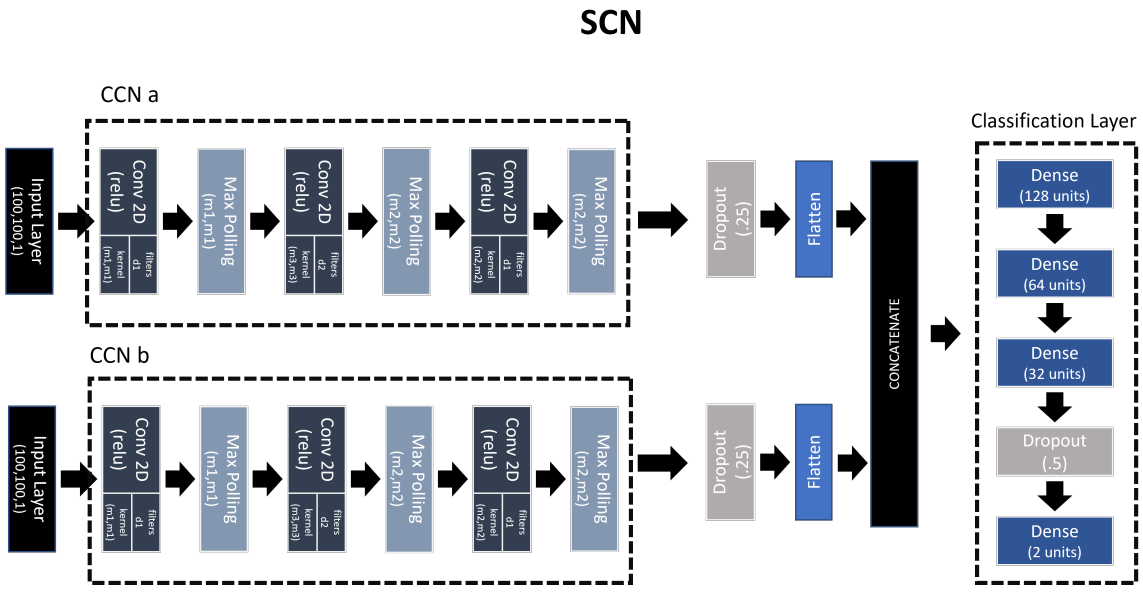


Figure 6.5: Siamese Convolution Neutral Network for PPRL (SCN)

The basic idea of this SCN is to use one CNN for each encoded record (e.g., bloom filter) to extract the patterns. Then, in the classification layers, we intend to use this pattern (extracted by the convolutional layers) to decide whether a record pair is a match or not. Notice that the encoded data makes the classification harder, mainly due to the reduced amount of information and the noise added by the encoding method. This fact is illustrated in our experiments and highlights the positive influence of the CRP representation in the PPRL classification step.

6.2 Evaluation

In this section, we evaluate the DLC effectiveness, i.e., the linkage and classifier quality. To this end, we present a discussion regarding the experimental results to answer the following Research Questions (RQ):

1. Is CRP able to improve the classifiers effectiveness?
2. What is the influence of the training parameters of the MHT workflow over the linkage quality?
3. What is the impact of using different classifiers - e.g., state-of-the-art NN (ResNet50) and classical Machine Learning classifiers (SVM and GBC) - in the MHT workflow?
4. Is DLC able to improve the PPRL quality results compared to the baseline and the competitor?
5. What is the linkage quality cost of the Privacy-Preserving Training step ($\epsilon > 0$) of the MHT workflow?

The linkage quality of the contributions presented in this chapter were evaluated in terms of Precision, Recall, and F1 metrics (detailed in Section 2.3.3). Next, we present considerations regarding the datasets, anonymization parameters, ML classifiers, baselines, and competitors employed in our experiments.

6.2.1 Experimental Design

To assess our contributions, we assume that the PPRL parties use a Balanced Bloom Filter (BBF) as their data's encoding/anonymization technique. Moreover, we employed four pairs of real-world and synthetic datasets to answer the aforementioned research questions. A summary of the dataset characteristics and their anonymization parameters are shown in Table 6.1.

	DATASET DETAILS						BF		
name	ds-type	input-type	record pair	matching pairs	attribute number	missing values	k	l	n
mvr	real	clean-clean	1×10^8	100	4	1,772	7	250	36
nvr	real	clean-clean	1×10^8	100	5	352	4	200	28
dblp-acm	real	clean-clean	6×10^6	500	3	14	3	450	94
census	synthetic	dirty	6.85×10^5	80	3	291	3	100	23

Table 6.1: Statistics of datasets

The datasets are detailed in Appendix A.5 and Chapter 5. We use different anonymization parameters in our experiments. Moreover, we also employ a dataset with '1:m' matching (census), one entity could be marked as similar to m other entities, to test our contribution in a Dirty-Clean RL task [33]. In other words, we tested our contribution in challenging linkage conditions.

The MHT workflow can use different ML techniques, from deep learning to statistical learning algorithms (e.g., SVM and Gradient Boosting Classifiers). We explored this characteristic and tested the CRP data representation as well as the MHT workflow, considering the CCN (detailed in Section 6.1.4). Moreover, we compared the CCN against two distinct families of classifiers, Deep Learning and Classical Machine Learning classifiers. Table 6.2 illustrates our experimental design.

parameter	values
Training Data Generation	
Random Sample Length Υ_s	10%
Percentage matching examples (Υ_m)	10%
Percentage non-matching examples (Υ_h)	50% to 300%
CRP encoding	
Heaviside function threshold (α)	1, 5, 10
CRP neighbors (m)	5, 10, 20, 30, 40
CRP distance function	Manhattan
Model training	
Classifiers	CCN, SCN, SVM[1], GBC[2], Resnet50
Metric Functions	Precision and Recall
Loss Functions	binary crossentropy, categorical crossentropy
epochs	50
Privacy Budget (ϵ)	.5 , 1 , 5 , 10

Table 6.2: Experimental Design

[1] trained with RBF kernel

[2] trained with 100 estimators, max depth of five, and learning rate 0.1

Regarding the **Deep Learning classifiers**, we tested the CCN against the ResNet50 [191], a state-of-the-art convolutional neural network for image partner recognition.In addition, we compare our CCN against well-known **classical Machine Learning classifiers**: Support Vector Machine (SVM) and Gradient Booster Classifier (GBC). We selected these techniques because: i) both of them are capable of dealing with high-dimensional data (such as CRP) [73], and ii) the simplicity and explainability provided by these techniques [73; 91]. Furthermore, we compare our approach against a **threshold-based classifier (baseline) and a competitor**. As the competitor, we employed AT-UC (Chapter 5).

We varied several parameters in the MHT workflow, i.e., the Training Data Generation, CRP, and Model training stages. To optimize the experimental design, we considered the same random sample length (Υ_s = 10% of the original dataset), percentage matching examples (Υ_m = 10% of the original dataset), metric functions, and CRP distance function.

In the Training Data Generation phase, we vary the percentage of non-matching examples (Υ_h) to investigate the influence of the ratio between the matching and non-matching examples in the final linkage quality.

We vary the threshold α and the number of BF neighbors (m), in the CRP encoding phase. The variation of parameters (α,m) is employed to investigate the existence of a relation between the anonymization parameters - bloom filter length (l) and hash functions (k) - and the CRP parameters. Regarding the CRP distance function (presented in Equation 2.7 and 6.1), we employed the Manhattan distance.

The classifiers were trained considering Precision and Recall as metric functions. We opted to combine these functions because PPRL (and RL) are usually unbalanced - e.g., more than 95% belong to one class (match or non-match). Thus, the use of other metrics (such as accuracy) could result in unbalanced classifiers [28]. Regarding the loss functions, we employed the Binary Cross Entropy (BCE) and Categorical Cross Entropy (CCE). CCE and BCE are defined in Equations 6.4 and 6.5, respectively.

$$CCE = -log(q(x)) \tag{6.4}$$

, where q(x) is the probability of class x in the prediction.

$$BCE = -(p(x) * log(q(x) + (1 - p(x)) * log(1 - q(x)) \tag{6.5}$$

, where p(x) is the probability of class x in the dataset (e.g., the probability of a record pair represents a match in the dataset).

BCE and CCE are reported as the most promising loss functions for linkage tasks [73].

Our approach and all baseline ones were implemented in Python 3, and we ran all experiments on a Linux server with 128 TensorCores (TPU), 2.4 GHz CPUs, and 48 GBytes of RAM. Moreover, considering the random sample in the first stages of the MHT workflow (Algorithm 4), we executed the experimental design five times to mitigate the influence of the sample in our results. Finally, it is worth mentioning that the programs and datasets are available at the authors' website[3].

6.2.2 Results

In this section, we answer the aforementioned research questions. The results presented in this section consider the experimental design shown in Table 6.2.

RQ.1) Is CRP able to improve the classifiers effectiveness?

To answer our first research question and investigate the influence of CRP (and its parametrization) on the linkage quality, we compared the results of SCN and CCN. SCN is a Siamese network that employs one CCN for each BF to extract relevant features in the encoded data. CCN receives as input a CRP of an encoded record pair. Figure 6.6 depicts the F1 for the complete experimental design.

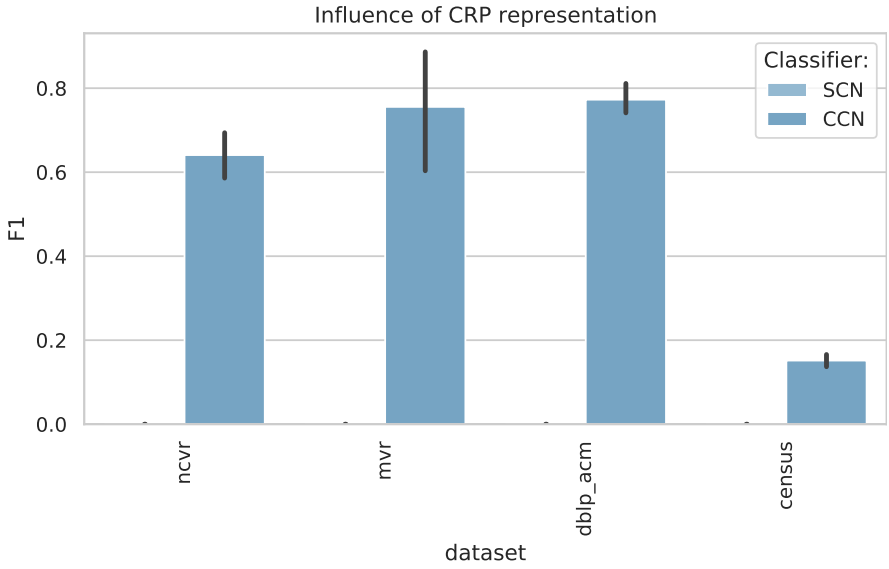


Figure 6.6: SCN vs. CCN linkage quality.

Notice that SCN achieved an F1 = 0. In other words, SCN could not learn patterns in the raw

[3]https://www.github.com/thiagonobrega/

encoded data and classified every record pair as a non-match for every observation. This result is expected mainly due to the dimensionality reduction and the noise addition performed by the PPRL anonymization step.

Considering the unsatisfactory results of SCN, we conducted additional experiments to investigate if SCN is capable of identifying matching instances of encoded records. We considered 200 epochs during the training phase and added two extra convolutional layers. Nevertheless, even with this modification, the F1 generated by SCN was smaller than 10% for the NCVR dataset. Observing these results, we can state that CRP highlights the encoded data patterns and simplifies the task of the convolutional layers of CCN. Therefore, the use of CRP improves the linkage quality in the MHT workflow.

In order to investigate the influence of the different CRP configurations (α and m) over the linkage quality, in Figure 6.7, we plotted the F1 for the CCN considering several CRP configurations.

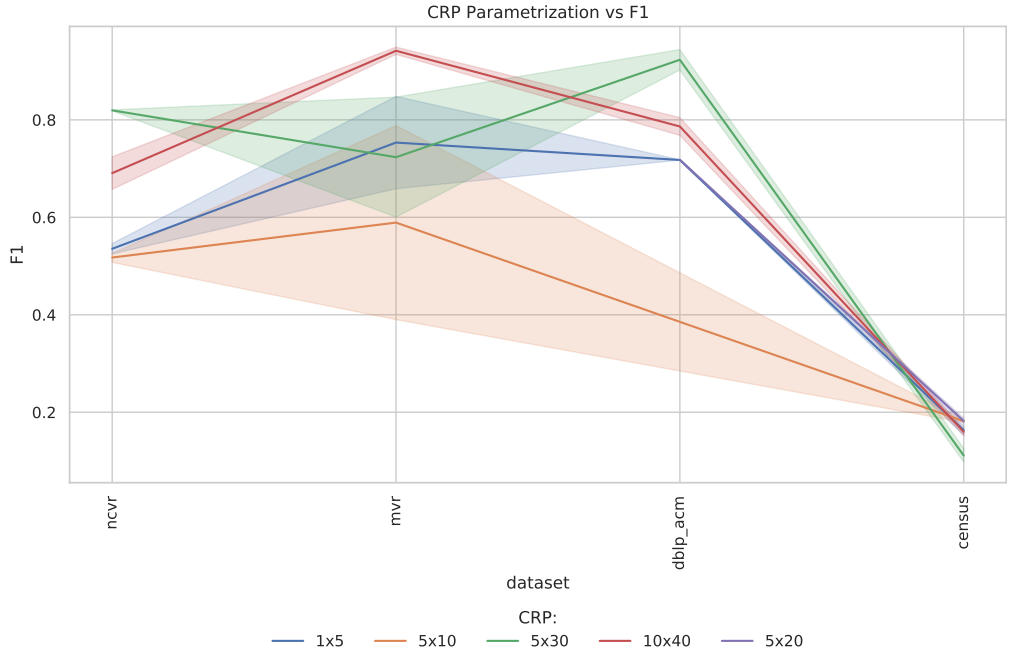


Figure 6.7: Linkage result for different CRP configurations.

Each colored line represents one CRP configuration ($\alpha \times m$). For instance, 10x40 represents a CRP with a Heaviside function threshold of 10 and 40 neighbors. In Figure 6.7, it is possible to notice a correlation between the anonymization (l, n, k) and CRP (α, m) parameters. Table 6.3 evidences this correlation. Table 6.3 presents the length of the filter (l), the mean number of bigrams of the filters (n), the number of hash functions (k), and the CRP configuration that achieved the best results in Figure 6.7.

dataset	l	k	n	best crp_conf
ncvr	200	4	28	5x30
mvr	250	7	36	10x40
census	100	3	23	5x20
dblp_acm	450	3	94	5x30

Table 6.3: Correlation between the encoding and CRP parameters.

The correlation between anonymization and CRP parameters is evidenced when we used a CRP configuration that employs: i) a Heaviside function threshold (α) close to the number of hash functions ($\alpha \approx k$); and ii) the number of neighbors (m) near to the mean number of n-grams of the BF ($m \approx n$). This parameter configuration ($\alpha \approx k$ and $m \approx n$) leaded to the best linkage results in our experiments.

The linkage results of this parametrization can be partially explained by the near-uniform distribution of bits ($fpr = .5$) performed by the Bloom Filter [151]. In other words, assume that each n-gram (n) of the encoded record is represented by k bits and the n-grams are almost uniformly distributed over l. For two similar BFs, it is expected (with a high probability) that among m bits, at least $m - k$ bits (considering the Manhattan distance in Equation 6.1) are identical. This insight requires further investigation and can be explored in future work.

In summary, configuring of the CRP parameters ($\alpha \approx k$ and $m \approx n$) results in enhanced linkage quality. Table 6.4 summarizes the mean gain of the aforementioned CRP parameterization for each dataset.

dataset	**precision**	**recall**	**F1**
mvr	27.5% $\pm$38.1	18.2% $\pm$33.8	23.8% $\pm$35.2
ncvr	-11.3% $\pm$14.4	51.2% $\pm$13.5	23.7% $\pm$9.6
dblp_acm	-0.6% $\pm$4.0	28.6% $\pm$8.5	18.8% $\pm$4.9
census	15.0% $\pm$22.3	1.7% $\pm$1.8	3.7% $\pm$4.0

Table 6.4: Influence of the CRP parameters over the linkage quality.

Table 6.4 demonstrated that the CRP parametrization produced (statistically relevant) gains for every tested metric and dataset. However, it is important to notice the limited gain for the census dataset. We believe that this result can be explained by the limitations of the MHT workflow (e.g., the

simplified data generation process, that provides only exact matching examples to the classifier) and the complexity of the dataset (presented in Appendix A.5). In the following research questions, we discuss the limitations of the MHT workflow. It is worth mentioning that, for the following research questions, we consider the CRP parametrization depicted in Table 6.3.

RQ.2) What is the influence of the training parameters of the MHT workflow over the linkage quality?

In this section, we investigate the influence of the percentage of non-matching examples (Υ_h) and loss functions (BCE and CCE) over the linkage quality. It is worth mentioning that we are considering only the CCN with CRP parameters shown in Table 6.3.

The ratio of negative (non-matching) and positive (matching) instances in the training data has the potential to influence the linkage metrics results [28]. In the RL context, depending on the classifier and the data distribution, it is recommended to use a 1:1 ratio of positive/negative examples to train the classifier [56]. Figure 6.8 illustrates the influence of the ratio between the matching and non-matching instances in the MHT workflow.

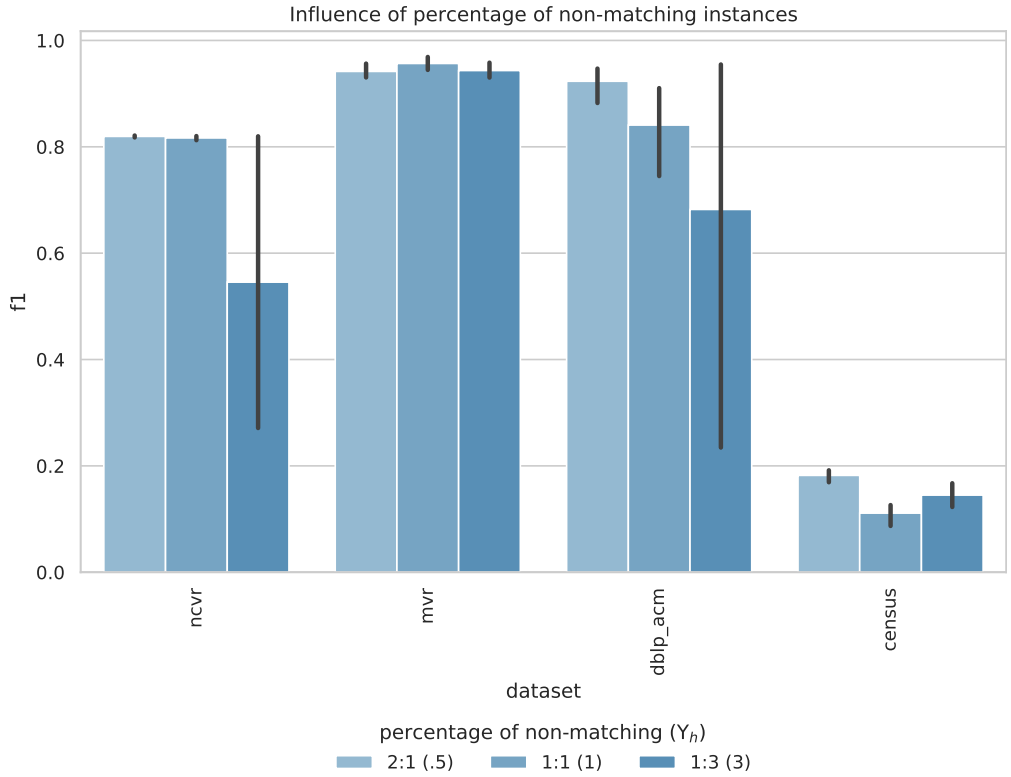


Figure 6.8: Influence of Υ_h and Υ_s parameters over the linkage quality.

The horizontal and vertical axes depict the F1 values and the datasets, respectively. The colored

bar represents the result of different matching and non-matching ratios. We use 2:1, 1:1, and 1:3. The first and second number of the ratio indicates the number of matching and non-matching examples, respectively. For instance, 2:1 means that we have two non-matching examples for each matching example.

In Figure 6.8, it is possible to observe that the ratios 2:1 and 1:1 are statically equivalent. The 1:3 ratio demonstrated that the F1 became more dependent on the sample (evidenced by the error bar for ncvr and dplp-acm datasets). Therefore, as in the RL context [56], we suggest using the 1:1 ratio in the MHT workflow parameters (Υ_s and Υ_h).

To assess the influence of the Loss functions, we tested our approach using two Loss functions (BCE and CCE) widely used in unbalanced classification tasks. Figure 6.9 exhibits our results.

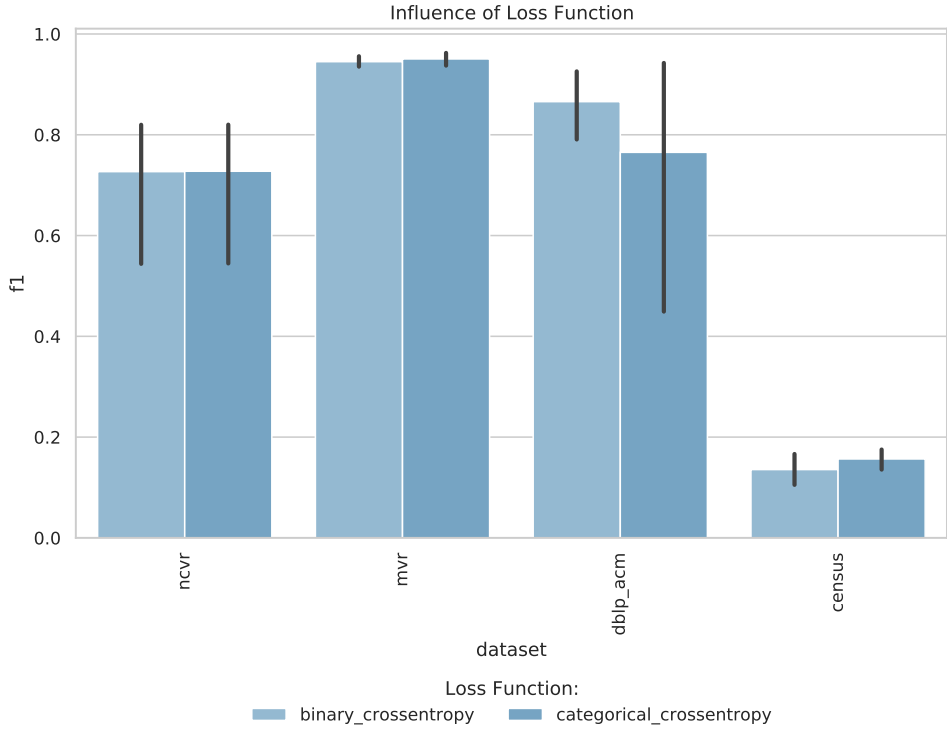


Figure 6.9: Influence of loss functions over the linkage quality.

According to Figure 6.9, the F1 measure is almost the same. However, for the dblp-acm dataset, the CCE presented a smaller F1. Moreover, the error bars for dplp-acm dataset indicate that F1 results were more dispersed than the BCE. In other words, depending on the sample, the use of CCE as a loss function could result in poor linkage results. This fact can be explained by the fact that BCE considers the probability of the target ($p(x)$) and prediction ($q(x)$) of the class, while CCE considers only the probability of the prediction. In other words, BCE provides a mechanism to adjust the loss in unbalanced classification tasks, similar to the RL task.

Considering the results of this research question, we use the 1:1 ratio and the BCE loss function for the following research question.

RQ.3) What is the impact of using different classifiers - e.g., state-of-the-art NN (ResNet50) and classical Machine Learning classifiers (SVM and GBC) - in the MHT Workflow?

The previous research question evaluated the linkage quality considering CCN. Now, we use different classifiers as competitors to CCN. Moreover, we compared CCN against a ResNet50 network, a convolutional neural network that is 50 layers deep, a widely used NN for image recognition tasks. We trained this network using the same input as CCN.

Recently, several studies [108; 109; 169] report that classical machine learning techniques (e.g., SVM and GBC) achieved results closer to the Deep Learning strategies. Moreover, the use of techniques such as SVM and GBC has the advantage of providing an explicable model (classifiers) [148] than Deep Learning models. Thus, considering the aforementioned studies, we tested the linkage capacities of SVM and GBC in our MHT workflow.

For each classifier, we plotted a Receiver Operating Characteristic curve (ROC curve) to assess the classifier's performance. The ROC curve illustrates a binary classifier diagnostic ability. The vertical axis of a ROC curve represents the True Positive Rate (or recall, defined in Chapter 2). The horizontal axis of the curve represents the False Positive Rate (FPR). Figure 6.10 depicts our results.

We plotted the Area Under the Curve (AUC) for each classifier. The AUC reflects the quality of the linkage. AUC value is within the range [0.5–1.0], where the minimum value represents the performance of a random classifier, and the maximum value corresponds to a perfect classifier (e.g., with a classification error rate equivalent to zero) [73].

The ResNet50 was unable to learn from the training data. In other words, the trained ResNet50 indicates that all test examples are non-matching, resulting in an F1=0. Therefore, the ResNet50 ROC curve can not be witnessed in Figure 6.10.

The performance of the CCN, SVM, and GBC classifiers for the ncvr, mvr, and dplp-acm datasets, is almost the same, aligned with the results reported by recent studies [108; 109; 169]. Moreover, for the mvr dataset, SVM and GBC overcame CCN, demonstrating that classical machine learning techniques could be employed considering the CRP representation of the encoded data to perform linkage tasks. It is worth remarking that, in RQ1, we argued that CRP could improve the linkage quality for Deep Learning-based classifiers. In this research question, we demonstrated that CRP could improve the linkage quality using classical machine learning classifiers.

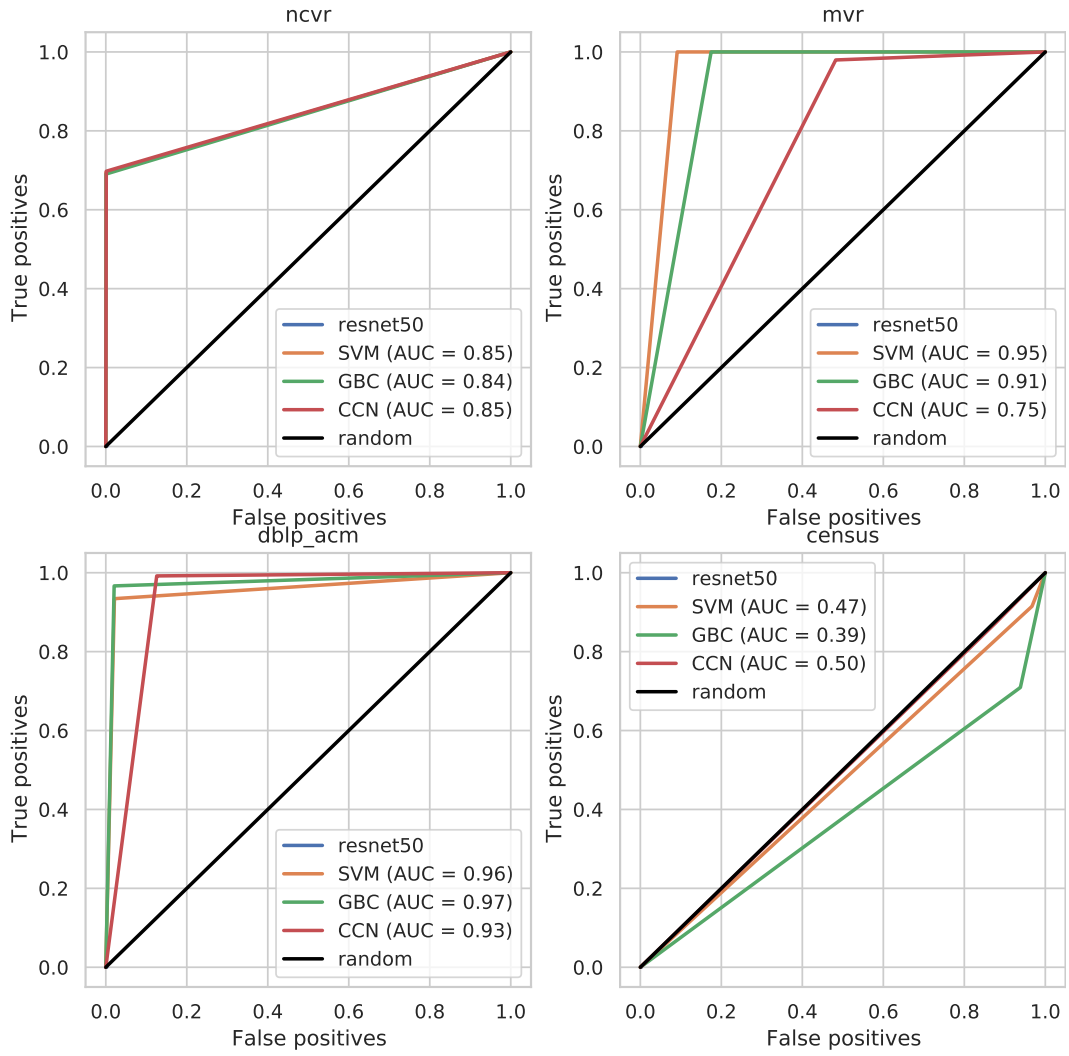


Figure 6.10: ROC Curves

It is important to comment on the results depicted for the census dataset. As mentioned in the previous chapter, the census dataset is a challenging dataset for record linkage. This fact is reflected in the low AUC of the classifiers. In addition, due to dirty and complex data characteristics presented in the dataset, the classifiers were unable to identify patterns that could distinguish matching from non-matching instances. These results are justified by the quality of the training examples for complex datasets. In other words, the MHT workflow does not provide a suitable training dataset due to the limitation of the training dataset generation phase. This limitation should be addressed in future work by exploring techniques such as Transfer Learning, Federated Learning, and Privacy-Preserving Data Generation in the training dataset generation phase.

Table 6.5 details the linkage results (Precision, Recall and F1) for the classifiers depicted in Figure 6.10.

dataset	model	precision	recall	f1
	CCN	87.23% ±0.0	10.38% ±0.08	18.55% ±0.13
census	**GBC**	69.86% ±1.5	7.94% ±0.13	14.26% ±0.25
	SVM	95.74% ±6.02	10.27% ±0.58	18.55% ±1.06
	CCN	98.95% ±0.34	90.82% ±0.19	94.71% ±0.26
dblp_acm	**GBC**	97.12% ±0.65	97.6% ±1.07	97.35% ±0.21
	SVM	93.4% ±0.04	98.21% ±0.04	95.74% ±0.04
	CCN	99.12% ±1.04	90.63% ±3.87	94.66% ±2.36
mvr	**GBC**	100.0% ±0.0	95.91% ±1.06	97.91% ±0.55
	SVM	100.0% ±0.0	97.52% ±0.14	98.75% ±0.07
	CCN	69.47% ±0.52	99.76% ±0.22	81.9% ±0.29
ncvr	**GBC**	68.52% ±0.76	99.9% ±0.09	81.28% ±0.51
	SVM	69.09% ±0.55	99.9% ±0.08	81.69% ±0.36

Table 6.5: Linkage metrics for different classifiers.

Table 6.5 shows the mean result and the standard error for all executions. The linkage metrics reflect the ROC curve results. GBC, SVM, and CCN have equivalent results for the tested data.

RQ.4) Is DLC able to improve the PPRL quality results compared to the baseline and the competitor?

To answer this research question, we compare the results of our approach against a baseline and a competitor. As a baseline, we consider the threshold-based classifier. As competitor, we employ AT-UC (Chapter 5). Figure 6.11, exhibit our results.

The most used classification technique in a PPRL context is the threshold-based classifier. To evaluate this classifier, we tested similarity values from .6 to 1 with an increment of 0.5 and reported the mean value for the three best results. Comparing DLC against the threshold-based classifier, it is possible to recognize that our approach overcomes the mean quality of the baseline.

Comparing DLC and AT-UC, the results (especially for the mvr and dblp-acm datasets) indicate that performing classification over the pattern extracted from the encoded data could significantly increase the linkage quality. On the other hand, the DLC results for the ncvr and census datasets could be explained by the limitations of the training data generation phase of the MHT workflow. In

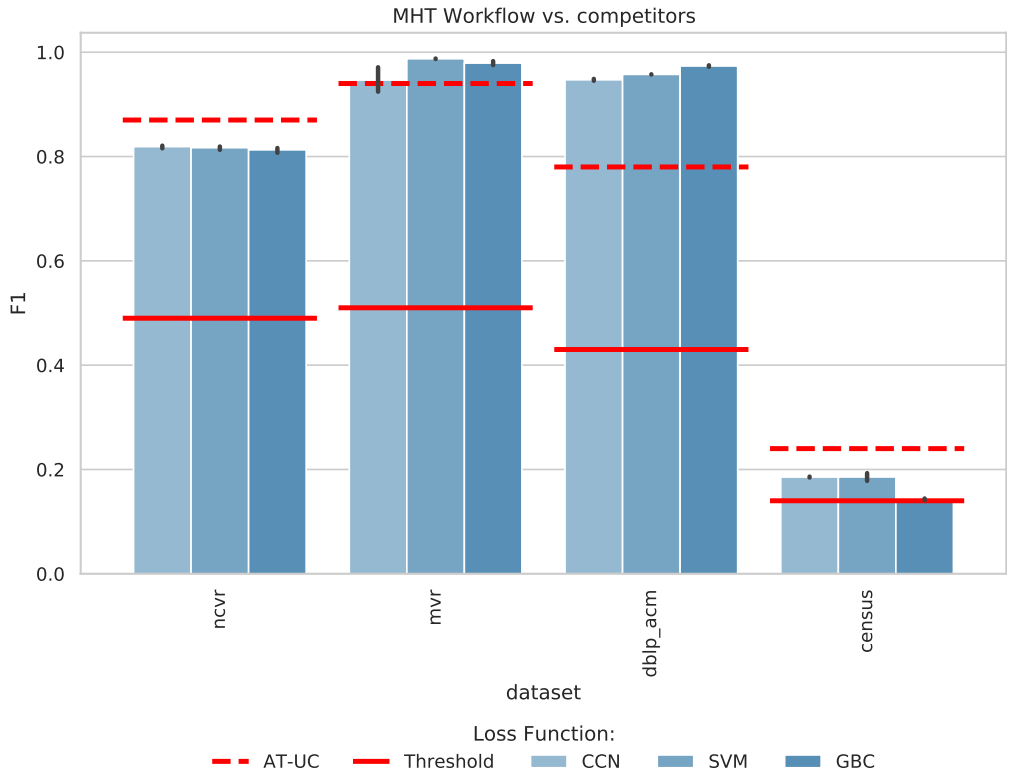


Figure 6.11: Comparison between DLC against competitor and baseline in terms of quality results.

summary, DLC overcame the baseline and produced results equivalent to its competitor.

The result depicted in this section is affected by the MHT workflow. We believe that the quality of the training data provided by the MHT workflow impacted the results. In other words, due to the simplicity of the training data generation stage, the training data is mainly exact matching. Therefore, the classifiers are trained with limited examples, making it hard to extract suitable patterns to distinguish between every matching/non-matching test data instance. This point is evidenced by the results of the census dataset, a complex synthetic dataset.

Notice that, even with limited training data, CRP along with ML techniques (e.g., CCN and GBC) was able to overcome the competitor or achieve results equivalent to its competitor, AT-UC. The limitation of the MHT workflow can be explored in future work by proposing novel methods for data augmentation and privacy-preserving data generation.

RQ.5) What is the linkage quality cost of the Privacy-Preserving Training step ($\epsilon > 0$) of the MHT workflow?

In Section 6.1.3, we argued that our contributions do not change the chance of success of the

existing attacks [182; 180; 181; 177]. However, in future work, Transfer Learning or Federated Learning techniques could be incorporated into our framework, and novel information can be used to train PPRL classifiers. In order to protect the privacy of the training data, we incorporated the Privacy-Preserving Training step, which provides Differential privacy guarantees to the trained classifier.

In order to evaluate the comprise between privacy and linkage quality, we executed the MHT workflow for the same input considering different privacy budgets ($\epsilon \in [1, 3, 10]$) for the MVR dataset. We calculated the ϵ using Equation 4.1[4]. Figure 6.12 depicts our results.

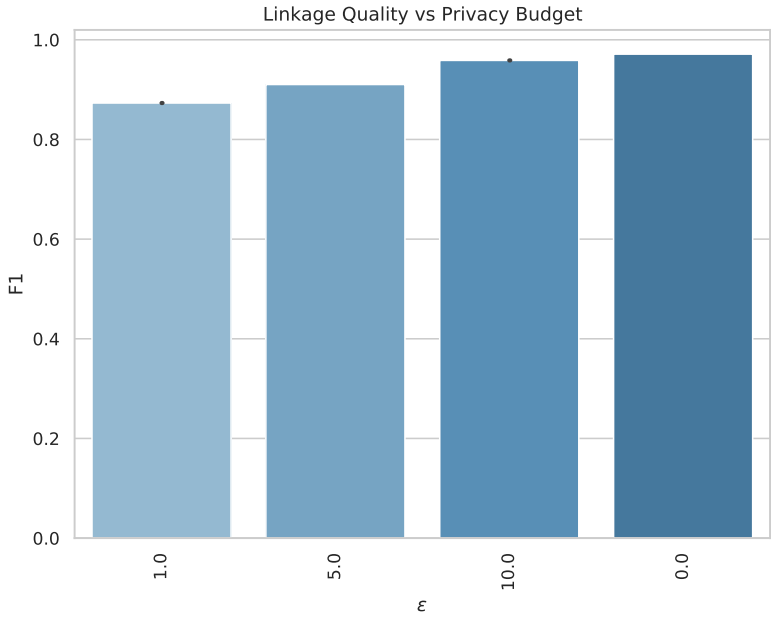


Figure 6.12: Privacy budget vs. linkage quality.

The colored bars represent F1 for each privacy budget. It is important to remark that the smaller ϵ values imply more privacy for the training data. Notice that the 0.0 indicates the value of F1 of the regular training (no privacy budget).

The results reported in Figure 6.12 indicate that the privacy budget impacts the linkage quality. Moreover, for a $\epsilon = 1$, a privacy budget capable of maintaining the privacy of the training data in a real-world application [44], F1 was reduced by almost 10%. Table 6.6 the impact of the privacy budget over the linkage quality metrics.

Table 6.6 explains the F1 results. Notice that F1 is impacted by the compromise of the recall imposed by the privacy budget. In other words, the privacy budget makes it harder for the classifier to extract and identify the patterns due to the noise added to the gradient of the optimizer.

This result was expected, and the PPRL parties should be aware of the linkage quality of the

[4]We use $C = 1.5$, and $\delta = 7.8^{-2}$,3.2^{-4},1.3^{-5}

ϵ	precison	recall	F1
1.0	0.61	-17.47	-9.81
5.0	0.61	-11.42	-6.09
10.0	0.61	-2.86	-1.26

Table 6.6: Linkage quality vs. Privacy budget.

privacy budget. Moreover, the PPRL parties must know that the privacy budget makes the classifier lose true match examples. However, the privacy budget in our experiments does not increase the false match. In other words, the classifier does not wrongly classify non-matching examples as matching. This fact is relevant because, depending on the context of the PPRL task, wrongly classifying non-matching as matching could have serious outcomes (e.g., marking a citizen as a terrorist).

6.3 Summary

In this chapter, we presented DLC, a methodology that operates a novel representation of the BF pairs (the CRP) that enables classifiers to distinguish between matching (and non-matching) BF pairs based on bit patterns of the encoded records. Moreover, by employing the CRP representation (our main contribution) as a feature space for ML-based classifiers, we are able to train classifiers to recognize patterns that indicate whether or not an encoded record pair represents a match instead of defining a separation boundary using standard similarity metrics. Finally, we proposed a workflow to test CRP and CCN.

The use of DLC simplifies the PPRL process by employing an ML-based classification step, mitigating the problem introduced by the selection of a proper similarity measure. Moreover, DLC was evaluated experimentally using several real-world data sources. The results demonstrated that DLC could overcome the quality of the most used technique (threshold) in the PPRL context. However, the results also demonstrated the limitations of our workflow, which we intend to address in future work and use techniques such as Transfer Learning, Federated Learning, and Privacy-Preserving Data Generation in the dataset generation phase.

Chapter 7

Conclusions and Future Work

In this chapter, we summarize the contributions presented in this thesis. Moreover, this chapter reveals the perspectives of future research topics by commenting on weaknesses and topics not addressed by the contributions present in this document.

7.1 Contributions

We presented our contributions to the PPRL process in Chapters 4, 5 and 6. We provide the contributions that: i) enable the usage of a novel adversary model and ii) improve the linkage quality by proposing an automatic classification approach to the PPRL process. Moreover, besides privacy and quality improvements, our work produces an impact on the adoption (usability) of PPRL by companies and governments by reducing the level of trust to execute PPRL and eliminating the need for an expert to define a classification threshold. Furthermore, the contributions presented in this thesis tackled our main objective - to improve the privacy and the linkage quality of the PPRL process (introduced in Section 1.2). In the following, we emphasize our contributions.

In Figure 1.2 (Chapter 1), we highlight our contributions within a traditional PPRL workflow. To provide a better explanation and show how our contributions can be fitted into a PPRL process, we expanded Figure 1.1 and, in each step of the process, we highlight the most relevant contributions (techniques, approaches, and protocols) of our work in Figure 7.1.

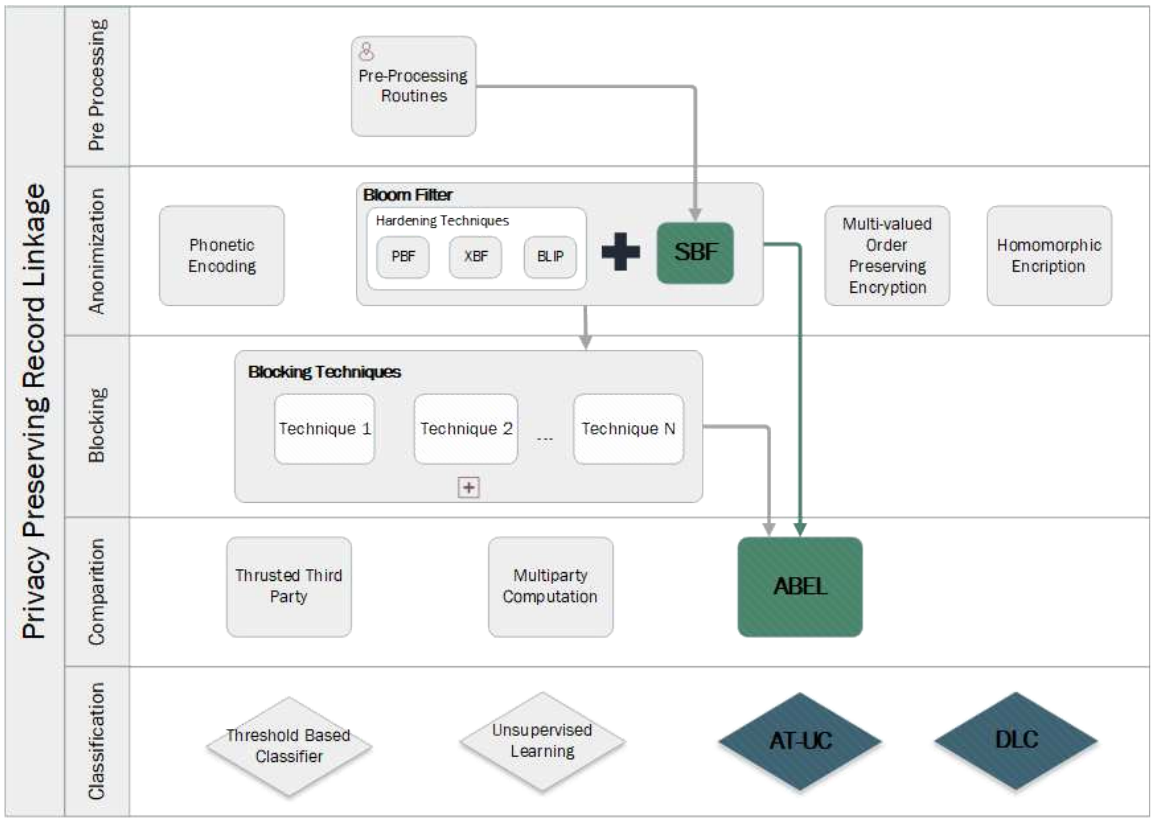


Figure 7.1: Detail of our contributions (SBF, ABEL, AT-UC, and DLC) to the PPRL workflow.

Figure 7.1 illustrates all steps of a generic PPRL process. Notice that our contributions are highlighted in different colors. Our first contribution (SBF) is located in the Anonymization step. Notice that SBF does not require any modification in the Pre-processing and Blocking steps. In other words, SBF can receive the input of any Pre-processing step and can be used as input to any Blocking step.

The SBF approach is designed to work with the majority of the current BF implementations, including BF with privacy-preserving enhancing mechanisms, such as the Bloom and Flip (BLIP) [153], Xor-Folding BF (XBF) [154], among others. The output of SBF is a set of splits that are consumed by the ABEL module to calculate the entity pairs' similarity. In the Comparison step, we propose ABEL, which uses SBF to perform the record similarity computation under the covert adversary model. ABEL's output is a similarity graph, where the edges are the entities' IDs, and the vertices represent the similarity values. It is worthwhile to mention that the PPRL output does not contain any entities' private data (anonymized or plain). Instead, it contains the entities' IDs and the similarities employed in the Classification step.

Therefore, with SBF and ABEL, we tackle our **first and second specific goals**, which are to

improve the privacy-preserving capabilities of the Bloom Filter and consider a novel adversary model to PPRL, respectively. Moreover, our experiments demonstrated a relationship between the SBF's error and split length with the linkage's privacy, efficiency, and effectiveness. In summary, smaller splits lead to privacy increase and efficiency, and effectiveness cost.

This similarity graph can be used as input to any of the available classification techniques. For instance, we use a threshold-based classification in the evaluation presented in Chapter 4. However, to address the problems related to the Threshold-based and Unsupervised Learning Classifiers, we propose AT-UC (Chapter 5).

AT-UC is a methodology to provide ML-based classifiers to the PPRL process. AT-UC exploits Transfer Learning techniques, which leverage the information available in public (or synthetic) datasets to train classifiers in a privacy-preserving context. In summary, in the AT-UC, we implemented the following contributions: i) a dataset selection method based on the dataset's resemblance and ii) the usage of an ML-based classifiers for PPRL.

AT-UC receives the entity similarity graph and a set of non-private training datasets as input. Finally, at the end of the AT-UC execution, a ML-based classifier will indicate whether a record pair is a match or not. Notice that the ML-based classifier is trained without the need of an expert to label instances or define a similarity threshold. Furthermore, the **AT-UC addresses our third specific goal**, the use of a machine learning-based classifier in a PPRL context.

Examining the AT-UC results, we conclude that the use of Transfer Learning techniques can improve the Classification step by leveraging the knowledge of public datasets to train ML-based classifiers to the PPRL, overcoming the threshold-based classifier's linkage results (the most used classifier in the PPRL context).

The Deep learning-based classifier for PPRL (**fourth specific goal**) is addressed by DLC (Chapter 6). The use of Deep learning classifiers in the PPRL process seeks to mitigate the issues of employing predefined binary similarity measures. Instead of utilizing the similarity measures (such as AT-UC), DLC trains classifiers to identify patterns over encoded data (e.g., Bloom Filter pairs) that indicate whether an encoded record pair represents a match or not.

In order to enable the usage of Deep learning techniques, our first contribution was a novel representation of encoded data (e.g., Bloom Filter pairs). This representation uses the Cross Recurrence Plot (CRP) to highlight patterns in the encoded data. Our second contribution was two deep learning models based on Convolutional Neural Networks that leverage the CRP of the anonymized data to identify matching record pairs. We also proposed a workflow to validate the use of our classifier in a PPRL context.

Analyzing the DLC results, we conclude that by representing the encoded record pairs as CRPs, it was possible to highlight patterns in the data that could be employed classifiers to identify matching entities. Moreover, our experiments demonstrated that the CRPs patterns could provide a suitable feature space for the ML-based classifier instead of standard binary similarity measures.

It is important to mention that DLC also contributes to PPRL linkage quality, with no or minimal cost to the privacy guarantee of the PPRL process. Moreover, DLC (and AT-UC) also contributes to adopting PPRL in a real-world application.

Notice that we illustrated the inputs and outputs of each contribution presented in this document. Furthermore, we explained that each contribution impacts only a specific PPRL step, preserving the input and output of all PPRL original steps. Thus, by preserving the original input and output of the PPRL process, our contributions can be easily plugged into a novel or existing PPRL workflow.

In summary, besides the practical aspects of our contributions (linkage quality and privacy), this work may contribute to the usage of PPRL solutions in real-world problems (such as epidemiologic, populational, law enforcement, and medical applications) by presenting a review of such application in each chapter of this document.

7.2 Future Work

In this section, we highlight the main open areas, research opportunities, and future work in the context of this thesis. We divided the research opportunities into two groups, the Privacy aspects and Linkage Quality & Novel PPRL Applications.

7.2.1 Privacy aspects

This section presents three viable research opportunities to improve the PPRL privacy guarantees.

Improve the privacy guarantees of the PPRL process (Privacy-Preserving Blockchain): existing Blockchain systems present low transactional privacy. All transactions, including the smart contract execution, are exposed on the Blockchain. Thus, by exposing the data on the Blockchain, an attacker could exploit the exposed data to attack the privacy of the individuals. To tackle this problem, several researchers [93; 187; 43; 149] and companies (such as Microsoft [1]) have been working on novel strategies in the last few years.

[1] Confidential Consortium Framework https://github.com/microsoft/CCF which employ SGX technology to execute the Blockchain nodes in secure enclaves

For instance, in 2016, Kosba et al. [93] proposed a decentralized smart contract system that encrypts the data on the Blockchain, thus retaining transactional privacy from the public's view. Moreover, in 2019, Ddwivedi et al. [43] proposed the use of a blockchain to provide secure management and analysis of healthcare big data. They proposed a modified blockchain with additional privacy and security properties based on advanced cryptographic primitives. The aforementioned works make data and transactions more secure and anonymous over a blockchain-based network.

Employ the aforementioned Privacy-Preserving Blockchain approach [93; 187; 43; 149] could improve the privacy guarantees of the PPRL process. Furthermore, such an approach could mitigate several privacy attacks, such as the attack mentioned in Appendix A.2.3.

Novel Privacy Attacks: several PPRL attacks [181; 180; 182; 177] employ statistical machine learning strategies to reidentify entities. These attacks exploit the similarity graph of the entities and the bit patterns of the encoded data to gain information. The CRP and RP representation (presented in Chapters 5 and 6) could be used to conceive novel PPRL privacy attacks. The Recurrence Plots could be employed to evidence patterns in the encoded data. Moreover, deep learning algorithms could be employed to facilitate the re-identification of the entities.

Differential Privacy in PPRL: Differential Privacy is under-explored within the PPRL context. This technique can be employed in SBF and ABEL to mitigate several PPRL attacks. Moreover, Differential Privacy can be employed in the Transfer Learning context to reduce the privacy risk of transferred knowledge and private data, such as described in Section 5.3.7.

7.2.2 Linkage Quality and Novel PPRL Applications

This section presents three viable research opportunities to improve the PPRL process linkage quality and novel PPRL application.

Distributed Representation of Words (DR) in PPRL: DR has been used in Natural Language Processing (NLP) applications as a feature representation of the input to Deep Learning and other sophisticated processing tools [73]. In the traditional RL context, NLP has been employed with meaningful results [41] to perform linkage tasks over non-structured data (e.g., text). One possible research opportunity is to employ privacy-preserving techniques to perpetuate (encode) the DR and propose a PPRL method to perform linkage over non-structured data. For instance, a privacy-preserving NLP solution could link medical records. Moreover, a privacy-preserving NLP solution could identify patients with the same condition (e.g., diabetes) and suggest treatments with privacy guarantees.

Federated Learning (collaborative learning): is a machine learning technique that trains an algorithm across multiple data owners without the need for data exchanging during the training pro-

cess [73]. In Chapter 6, we propose a method (DLC) that enables the possibility of using Federated Learning in the PPRL context. However, we did not explore this possibility. Future research can be conducted to propose novel Federated Learning strategies to PPRL or assess the existing Federated Learning in the PPRL context.

Deep Unsupervised Domain Adaptation: in this work, we presented two contributions to the classification step of PPRL: AT-UC and DLC. The first transfers the knowledge from non-private datasets, and the second employs deep learning classifiers to match entity pairs. Proposing a technique that transfers the knowledge of non-private datasets to be employed in the MHT workflow could improve the linkage quality of the PPRL process - e.g., employing privacy-preserving data augmentation techniques (further detailed in Section 6.3). Therefore, a combination of AT-UC and DLC could be considered a future work opportunity.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Daniel Abril, Vicenç Torra, and Guillermo Navarro-Arribas. Supervised learning using a symmetric bilinear form for record linkage. *Information Fusion*, 26:144–153, 2015.

[3] Mohammad Al-Rubaie and J. Morris Chang. Privacy-Preserving Machine Learning: Threats and Solutions. *IEEE Security & Privacy*, 17(2):49–58, 3 2019.

[4] CA Ardagna, M Cremonini, E Damiani, S De Capitani di Vimercati, and P Samarati. Privacy-enhanced location services. digital privacy: Theory, technologies and practices, 2007.

[5] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. A Survey of Attacks on Ethereum Smart Contracts (SoK). In Matteo Maffei and Mark Ryan, editors, *Kyoto Daigaku kokukagaku kiyo. Bulletin of stomatology, Kyoto University*, volume 10204 of *Lecture Notes in Computer Science*, pages 164–186. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017.

[6] Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. *Journal of Cryptology*, 23(2):281–343, 2010.

[7] Yonatan Aumann and Yehuda Lindell. Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries. *Journal of Cryptology*, 23(2):281–343, 4 2010.

[8] Zeinab Bahmani, Leopoldo Bertossi, and Nikolaos Vasiloglou. ERBlox: Combining matching dependencies with machine learning for entity resolution. *International Journal of Approximate Reasoning*, 83:118–141, 2017.

[9] Pierre Baldi, Søren Brunak, Yves Chauvin, Claus AF Andersen, and Henrik Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412–424, 2000.

[10] Carlo Batini and Monica Scannapieco. *Data and Information Quality*. Data-Centric Systems and Applications. Springer International Publishing, 1 edition, 2016.

[11] Roberto J Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *21st International conference on data engineering (ICDE'05)*, pages 217–228. IEEE, 2005.

[12] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *Proceedings of the 2012 ACM conference on Computer and communications security - CCS '12*, page 784, New York, New York, USA, 2012. ACM Press.

[13] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175, 5 2010.

[14] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. *Advances in Neural Information Processing Systems*, pages 137–144, 2007.

[15] James H Boyd, Anna M Ferrante, Christine M O'Keefe, Alfred J Bass, Sean M Randall, and James B Semmens. Data linkage infrastructure for cross-jurisdictional health-related research in australia. *BMC health services research*, 12(1):1–8, 2012.

[16] Justin Brickell and Vitaly Shmatikov. Privacy-preserving classifier learning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5628 LNCS:128–147, 2009.

[17] Andrei Broder and Michael Mitzenmacher. Network applications of bloom filters: A survey. *Internet Mathematics*, 1(4):485–509, 2004.

[18] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6, 1993.

[19] Ursin Brunner and Kurt Stockinger. Entity Matching on Unstructured Data: An Active Learning Approach. *Proceedings - 6th Swiss Conference on Data Science, SDS 2019*, pages 97–102, 2019.

[20] LA Bygrave. Data protection pursuant to the right to privacy in human rights treaties. *International Journal of Law and Information Technology*, 6(3):247–284, 01 1998.

[21] Jason Castiglione, Dusko Pavlovic, and Peter-Michael Seidel. *Privacy Protocols*, pages 167–191. Springer International Publishing, Cham, 2019.

[22] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. *Advances in Neural Information Processing Systems 21 - Proceedings of the 2008 Conference*, pages 289–296, 2009.

[23] Hsinchun Chen, Edna Reid, Joshua Sinai, Andrew Silke, and Boaz Ganor. *Terrorism Informatics: Knowledge Management and Data Mining for Homeland Security*. Springer, 2008.

[24] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[25] Peter Christen. A two-step classification approach to unsupervised record linkage. *Conferences in Research and Practice in Information Technology Series*, 70(Clarke):111–119, 2007.

[26] Peter Christen. Automatic record linkage using seeded nearest neighbour and support vector machine classification. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 151–159, 2008.

[27] Peter Christen. Automatic record linkage using seeded nearest neighbour and support vector machine classification. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 151–159, 2008.

[28] Peter Christen. *Data Matching*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[29] Peter Christen. Preparation of a real voter data set for record linkage and duplicate detection research. Technical report, 2013.

[30] Peter Christen, Thilina Ranbaduge, and Rainer Schnell. *Linking Sensitive Data*. Springer International Publishing, Cham, 2020.

[31] Peter Christen, Thilina Ranbaduge, Dinusha Vatsalan, and Rainer Schnell. Precise and Fast Cryptanalysis for Bloom Filter Based Privacy-Preserving Record Linkage. *IEEE Trans. Knowl. Data Eng.*, 31(11):2164–2177, 2019.

[32] Peter Christen, Rainer Schnell, Dinusha Vatsalan, and Thilina Ranbaduge. *Efficient Cryptanalysis of Bloom Filters for Privacy-Preserving Record Linkage Peter*, volume 10235 of *Lecture Notes in Computer Science*. Springer International Publishing, Cham, 2017.

[33] Peter Christen and Dinusha Vatsalan. A flexible data generator for privacy-preserving data mining and record linkage. 2012.

[34] Peter Christen, Dinusha Vatsalan, and Vassilios S Verykios. Challenges for privacy preservation in data integration. *Journal of Data and Information Quality*, 5(1-2):1–3, 2014.

[35] Peter Christen, Anushka Vidanage, Thilina Ranbaduge, and Rainer Schnell. Pattern-mining based cryptanalysis of bloom filters for privacy-preserving record linkage. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 10939 LNAI:530–542, 2018.

[36] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. 2006.

[37] Mary Cryan. *Probability and Computing Randomized Algorithms and Probabilistic Analysis*. JSTOR, 2006.

[38] Geoff Cumming and Sue Finch. Inference by eye: confidence intervals and how to read pictures of data. *The American psychologist*, 60(2):170–180, 2005.

[39] Tamraparni Dasu, Yaron Kanza, and Divesh Srivastava. Unchain Your Blockchain. In *Foundations and Applications of Blockchain (FAB)*, number 1, pages 16–23, 2018.

[40] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. BLOCKBENCH: A Framework for Analyzing Private Blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data - SIGMOD '17*, pages 1085–1100, New York, New York, USA, 2017. ACM Press.

[41] Xin Luna Dong and Theodoros Rekatsinas. Data Integration and Machine Learning. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1645–1650, New York, NY, USA, 5 2018. ACM.

[42] Elizabeth Ashley Durham. *A framework for accurate, efficient private record linkage*. PhD thesis, Vanderbilt University, 2012.

[43] Ashutosh Dhar Dwivedi, Gautam Srivastava, Shalini Dhar, and Rajani Singh. A decentralized privacy-preserving healthcare blockchain for iot. *Sensors*, 19(2):326, 2019.

[44] Cynthia Dwork. Theory and Applications of Models of Computation. 4978:1–19, 2008.

[45] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[46] Muhammad Ebraheem, Saravanan Thirumuruganathan, Shafiq Joty, Mourad Ouzzani, and Nan Tang. DeepER – Deep Entity Resolution. 11(11), 2017.

[47] Jean-Pierre Eckmann, S Oliffson Kamphorst, David Ruelle, et al. Recurrence plots of dynamical systems. *World Scientific Series on Nonlinear Science Series A*, 16:441–446, 1995.

[48] Mohammadreza Ektefa, Fatimah Sidi, Hamidah Ibrahim, Marzanah A. Jabar, and Sara Memar. A comparative study in classification techniques for unsupervised record linkage model. *Journal of Computer Science*, 7(3):341–347, 2011.

[49] Khaled El Emam, Fida Kamal Dankar, Romeo Issa, Elizabeth Jonker, Daniel Amyot, Elise Cogo, Jean-Pierre Corriveau, Mark Walker, Sadrul Chowdhury, Regis Vaillancourt, et al. A globally optimal k-anonymity method for the de-identification of health data. *Journal of the American Medical Informatics Association*, 16(5):670–682, 2009.

[50] Muhammad El-Hindi, Martin Heyden, Carsten Binnig, Ravi Ramamurthy, Arvind Arasu, and Donald Kossmann. BlockchainDB - Towards a Shared Database on Blockchains. In *Proceedings of the 2019 International Conference on Management of Data - SIGMOD '19*, volume 12, pages 1905–1908, New York, New York, USA, 2019. ACM Press.

[51] Mohamed G. Elfeky, Vassilios S. Verykios, and Ahmed K. Elmagarmid. TAILOR: A record linkage toolbox. *Proceedings - International Conference on Data Engineering*, pages 17–28, 2002.

[52] The Faculty, Manitoba Winnipeg, and Canada August. Privacy-preserving Biomedical Data Sharing and Computation. (August), 2020.

[53] Ivan P Fellegi and Alan B Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

[54] Niels Ferguson, Bruce Schneier, and Tadayoshi Kohno. *Cryptography Engineering Design Principles and Practical Applications*. Wiley, 2010.

[55] Maria Isabel Mou Sequeira Fernandes. *Reconciling data privacy with sharing in next-generation genomic workflow*. PhD thesis, University of Luxembourg, 2020.

[56] Diego Fernandes de Araújo, Carlos Eduardo Santos Pires, and Dimas Cassimiro Nascimento. Leveraging active learning to reduce human effort in the generation of ground-truth for entity resolution. *Computational Intelligence*, 36(2):743–772, 2020.

[57] Jeffrey Fisher, Peter Christen, and Qing Wang. Active Learning Based Entity Resolution Using Markov Logic. volume 7301, pages 338–349. 2016.

[58] Sam Fletcher and Md Zahidul Islam. Decision tree classification with differential privacy: A survey. *ACM Computing Surveys (CSUR)*, 52(4):1–33, 2019.

[59] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

[60] Srivatsava Ranjit Ganta, Shiva Prasad SP Kasiviswanathan, and Adam Smith. Composition attacks and auxiliary information in data privacy. *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '08*, pages 265–274, 2008.

[61] Aris Gkoulalas-Divanis, Grigorios Loukides, and Jimeng Sun. Publishing data from electronic health records while preserving privacy: A survey of algorithms. *Journal of biomedical informatics*, 50:4–19, 2014.

[62] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[63] Inc. Google. Google Flu Trends.

[64] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern recognition*, 77:354–377, 2018.

[65] Lifang Gu and Rohan Baxter. Decision models for record linkage. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3755 LNAI:146–160, 2006.

[66] Xi He, Ashwin Machanavajjhala, Cheryl Flynn, and Divesh Srivastava. Composing Differential Privacy and Secure Computation. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security - CCS '17*, number 1, pages 1389–1406, New York, New York, USA, 2017. ACM Press.

[67] Ziyue He. Probabilistic Graphical Model for Entity Resolution. 2017.

[68] Boyi Hou, Qun Chen, Jiquan Shen, Xin Liu, Ping Zhong, Yanyan Wang, Zhaoqiang Chen, and Zhanhuai Li. Gradual machine learning for entity resolution. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, pages 3526–3530, 2019.

[69] IBGE. Nomes no Brasil, 2010.

[70] Ali Inan, Murat Kantarcioglu, Gabriel Ghinita, and Elisa Bertino. Private record matching using differential privacy. In *Proceedings of the 13th International Conference on Extending Database Technology - EDBT '10*, page 123, New York, New York, USA, 2010. ACM Press.

[71] Rohit Jain and Sunil Prabhakar. Trustworthy data from untrusted databases. *Proceedings - International Conference on Data Engineering*, pages 529–540, 2013.

[72] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*, volume 103 of *Springer Texts in Statistics*. Springer New York, New York, NY, 2013.

[73] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer Texts in Statistics. Springer New York, New York, NY, 2021.

[74] Matthias Jarke and Christoph Quix. *Federated Data Integration in Data Spaces*, pages 181–194. Springer International Publishing, Cham, 2022.

[75] Delaram Javdani, Hossein Rahmani, Milad Allahgholi, and Fatemeh Karimkhani. DeepBlock: A Novel Blocking Approach for Entity Resolution using Deep Learning. *2019 5th International Conference on Web Research (ICWR)*, pages 41–44, 2019.

[76] Somesh Jha, Louis Kruger, and Vitaly Shmatikov. Towards practical privacy for genomic computation. *Proceedings - IEEE Symposium on Security and Privacy*, pages 216–230, 2008.

[77] Wei Jiang, Chris Clifton, and Murat Kantarcioğlu. Transforming semi-honest protocols to ensure accountability. *Data and Knowledge Engineering*, 65(1):57–74, 2008.

[78] Anna Jurek, Jun Hong, Yuan Chi, and Weiru Liu. A novel ensemble learning approach to unsupervised record linkage. *Information Systems*, 71(June):40–54, 2017.

[79] Murat Kantarcioglu and Elena Ferrari. Research Challenges at the Intersection of Big Data, Security and Privacy. *Frontiers in Big Data*, 2(February):1–6, 2 2019.

[80] Alexandros Karakasidis and Georgia Koloniari. Scalable Blocking for Privacy Preserving Record Linkage. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 527–536, 2015.

[81] Alexandros Karakasidis, Vassilios S. Verykios, and Peter Christen. Fake Injection Strategies for Private Phonetic Matching. pages 9–24. 2012.

[82] Dimitrios Karapiperis, Aris Gkoulalas-Divanis, and Vassilios S. Verykios. Summarization algorithms for record linkage. In *Advances in Database Technology - EDBT*, volume 2018-March, pages 73–84. OpenProceedings.org, 2018.

[83] Dimitrios Karapiperis, Aris Gkoulalas-Divanis, and Vassilios S. Verykios. Summarizing and linking electronic health records. *Distributed and Parallel Databases*, 2019.

[84] Jungo Kasai, Kun Qian, Sairam Gurajada, Yunyao Li, and Lucian Popa. Low-resource Deep Entity Resolution with Transfer and Active Learning. pages 5851–5861, 2019.

[85] Youngeun Kim, Donghyeon Cho, and Sungeun Hong. Towards Privacy-Preserving Domain Adaptation. *IEEE Signal Processing Letters*, 27:1675–1679, 2020.

[86] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[87] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 1d convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151:107398, 2021.

[88] Nishadi Kirielle, Peter Christen, and Thilina Ranbaduge. Transer: Homogeneous transfer learning for entity resolution. In Julia Stoyanovich, Jens Teubner, Paolo Guagliardo, Milos Nikolic, Andreas Pieris, Jan Mühlig, Fatma Özcan, Sebastian Schelter, H. V. Jagadish, and Meihui Zhang, editors, *Proceedings of the 25th International Conference on Extending Database Technology, EDBT 2022, Edinburgh, UK, March 29 - April 1, 2022*, pages 2:118–2:130. OpenProceedings.org, 2022.

[89] Denis Kleyko, Abbas Rahimi, Ross W. Gayler, and Evgeny Osipov. Autoscaling Bloom filter: controlling trade-off between true and false positives. *Neural Computing and Applications*, 32(8):3675–3684, 2020.

[90] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille, 2015.

[91] Pradap Konda, Sanjib Das, Paul Suganthan, Anhai Doan, Adel Ardalan, Jeffrey R Ballard, Han Li, Fatemah Panahi, Haojun Zhang, Jeff Naughton, Shishir Prasad, Ganesh Krishnan, Rohit Deep, and Vijay Raghavendra. Magellan: Toward Building Entity Matching Management Systems. Technical report, 2018.

[92] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 839–858. IEEE, 5 2016.

[93] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)*, pages 839–858. IEEE, 2016.

[94] Chakravanti Rajagopalachari Kothari. *Research methodology: Methods and techniques*. New Age International, 2004.

[95] Nick Koudas, Sunita Sarawagi, and Divesh Srivastava. Record linkage: similarity measures and algorithms. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 802–803, 2006.

[96] Sanjay Krishnan, Jiannan Wang, Eugene Wu, Michael J. Franklin, and Ken Goldberg. ActiveClean : Interactive Data Cleaning For Statistical Modeling. *Proeedings of the VLDB Endowment*, 9(12):948–959, 2016.

[97] Hye-chung Kum, Ashok Krishnamurthy, Ashwin Machanavajjhala, and Stanley C Ahalt. Social Genome : Putting Big Data to Work. pages 56–63, 2014.

[98] M. Kuzu, M. Kantarcioglu, E. a. Durham, C. Toth, and B. Malin. A practical approach to achieve private medical record linkage in light of public resources. *Journal of the American Medical Informatics Association*, pages 285–292, 2012.

[99] Mehmet Kuzu, Murat Kantarcioglu, Elizabeth Durham, and Bradley Malin. A Constraint Satisfaction Cryptanalysis of Bloom Filters in Private Record Linkage. *Privacy Enhancing Technologies*, 6794:226–245, 2011.

[100] Lingli Li, Xiaodan Shang, Jinbao Li, and Jin Hu. Learning distance metrics for entity resolution. *IEEE Access*, 6(8):54900–54909, 2018.

[101] Lun Li, Jiqiang Liu, Lichen Cheng, Shuo Qiu, Wei Wang, Xiangliang Zhang, and Zonghua Zhang. CreditCoin: A Privacy-Preserving Blockchain-Based Incentive Announcement Network for Communications of Smart Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 19(7):2204–2220, 2018.

[102] Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[103] Yehuda Lindell. *Tutorials on the Foundations of Cryptography*. Information Security and Cryptography. Springer International Publishing, Cham, 2017.

[104] Yehuda Lindell. *Tutorials on the Foundations of Cryptography*. Springer, 2017.

[105] Michael Loster, Ioannis Koumarelas, and Felix Naumann. Knowledge transfer for entity resolution with siamese neural networks. *Journal of Data and Information Quality (JDIQ)*, 13(1):1–25, 2021.

[106] Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. Transfer learning using computational intelligence: A survey. *Knowledge-Based Systems*, 80:14–23, 2015.

[107] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. L-diversity: privacy beyond k-anonymity. In *22nd International Conference on Data Engineering (ICDE'06)*, pages 24–24, 2006.

[108] Satyajit Mahapatra, Vivek Raj Gupta, Sitanshu Sekhar Sahu, and Ganapati Panda. Deep neural network and extreme gradient boosting based hybrid classifier for improved prediction of protein-protein interaction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(1):155–165, 2021.

[109] Anurag Malik, Mandeep Kaur Saggi, Sufia Rehman, Haroon Sajjad, Samed Inyurt, Amandeep Singh Bhatia, Aitazaz Ahsan Farooque, Atheer Y Oudah, and Zaher Mundher Yaseen. Deep learning versus gradient boosting machine for pan evaporation prediction. *Engineering Applications of Computational Fluid Mechanics*, 16(1):570–587, 2022.

[110] Koray Mancuhan and Chris Clifton. Statistical Learning Theory Approach for Data Classification with l-diversity. 2016.

[111] Norbert Marwan and Jürgen Kurths. Nonlinear analysis of bivariate data with cross recurrence plots. *Physics Letters A*, 302(5-6):299–307, 2002.

[112] Norbert Marwan and Charles L Webber. Mathematical and computational foundations of recurrence quantifications. In *Recurrence Quantification Analysis*, pages 3–43. Springer, 2015.

[113] David McClure and Jerome P. Reiter. Differential privacy and statistical disclosure risk measures: An investigation with binary synthetic data. *Transactions on Data Privacy*, 5(3):535–552, 2012.

[114] Srujana Merugu and Joydeep Ghosh. Privacy-preserving distributed clustering using generative models. In *Third IEEE International Conference on Data Mining*, pages 211–218. IEEE, 2003.

[115] Demetrio Gomes Mestre. *Uma Abordagem para Aprimoramento do Balanceamento de Carga do Método de Resolução de Entidades Standard Blocking baseado em MapReduce*. PhD thesis, Universidade Federal de Campina Grande, 2013.

[116] Gerome Miklau and Dan Suciu. A formal analysis of information disclosure in data exchange. *Journal of Computer and System Sciences*, 73(3):507–534, 2007.

[117] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. 10 2013.

[118] Kato Mivule, Claude Turner, and Soo Yeon Ji. Towards a differential privacy and utility preserving machine learning classifier. *Procedia Computer Science*, 12:176–181, 2012.

[119] Hirofumi Miyajima, Noritaka Shigei, Syunki Makino, Hiromi Miyajima, Yohtaro Miyanishi, Shinji Kitagami, and Norio Shiratori. A proposal of privacy preserving reinforcement learning for secure multiparty computation. *Artificial Intelligence Research*, 6(2):57, 2017.

[120] Noman Mohammed, Benjamin C M Fung, and Mourad Debbabi. Anonymity meets game theory: Secure data integration with malicious participants. *VLDB Journal*, 20(4):567–588, 2011.

[121] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, Anhai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. Deep learning for entity matching: A design space exploration. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 19–34, 2018.

[122] Senthil Nathan, Chander Govindarajan, Adarsh Saraf, Manish Sethi, and Praveen Jayachandran. Blockchain meets database. *Proceedings of the VLDB Endowment*, 12(11):1539–1552, 7 2019.

[123] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, 2016.

[124] Sahand Negahban, Benjamin I P Rubinstein, and Jim Gemmell. Scaling Multiple-Source Entity Resolution using. *Cikm*, pages 2224–2228, 2012.

[125] Frank Niedermeyer, Simone Steinmetzer, Martin Kroll, Rainer Schnell, Mehmet Kuzu, Murat Kantarcioglu, Elizabeth Durham, and Bradley Malin. Cryptanalysis of Basic Bloom Filters Used for Privacy Preserving Record Linkage. *Journal of Privacy and Confidentiality*, 6(2):59–79, 2014.

[126] Thiago Nóbrega, Carlos Eduardo S Pires, and Dimas Cassimiro Nascimento. Towards auditable and intelligent privacy-preserving record linkage. In *Anais Estendidos do XXXVI Simpósio Brasileiro de Bancos de Dados*, pages 99–105. SBC, 2021.

[127] Thiago Pereira Nóbrega. Pareamento privado de atributos no contexto da resolução de entidades com preservação de privacidade. 2018.

[128] Thiago Pereira Nóbrega, Carlos Eduardo S. Pires, Tiago Brasileiro Araújo, and Demetrio Gomes Mestre. Blind attribute pairing for privacy-preserving record linkage. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing - SAC '18*, pages 557–564, New York, New York, USA, 2018. ACM Press.

[129] Thiago Pereira da Nóbrega, Carlos Eduardo Santos Pires, and Tiago Brasileiro Araujo. Avaliação Empírica de Técnicas de Comparação Privada Aplicadas na Resolução de Entidades. In *Proceedings of the 31 st of the Brazilian Symposium on Databases (SBBD16)*, pages 121–126, 2016.

[130] Thiago Nóbrega, Carlos Eduardo S. Pires, and Dimas Cassimiro Nascimento. Blockchain-based privacy-preserving record linkage: enhancing data privacy in an untrusted environment. *Information Systems*, 102:101826, 2021.

[131] Sinno Jialin Pan. Transfer learning. *Data Classification: Algorithms and Applications*, pages 537–570, 2014.

[132] Sinno Jialin Pan and Qiang Yang Fellow. A Survey on Transfer Learning. *IEEE Transactions on knowledge and data engineering*, 22:1345–1359, 2009.

[133] Sinno Jialin Pan, Xiaochuan Ni, Jian Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pages 751–760, 2010.

[134] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. 2009.

[135] George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. A Survey of Blocking and Filtering Techniques for Entity Resolution. (May), 2019.

[136] George A Papadakis, Dimitrios Skoutas, and Emmanouil Thanos. A Survey of Blocking and Filtering Techniques for Entity Resolution. (May), 2019.

[137] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10):1071–1083, jun 2018.

[138] Payal Parmar, Shraddha B. Padhar, Shafika N. Patel, Niyatee I. Bhatt, and Rutvij H. Jhaveri. Survey of Various Homomorphic Encryption algorithms and Schemes. *International Journal of Computer Applications*, 91(8):26–32, 2014.

[139] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. 2 2018.

[140] Robespierre Pita, Clicia Pinto, Pedro Melo, Malu Silva, Marcos Barreto, and Davide Rasella. A Spark-based workflow for probabilistic record linkage of healthcare data. *CEUR Workshop Proceedings*, 1330:17–26, 2015.

[141] Hans-Ulrich Prokosch, Thomas Bahls, Martin Bialke, Jürgen Eils, Christian Fegeler, Julian Gruendner, Birger Haarbrandt, Christopher Hampf, Wolfgang Hoffmann, Hauke Hund, et al.

The covid-19 data exchange platform of the german university medicine. In *Challenges of Trustable AI and Added-Value on Health*, pages 674–678. IOS Press, 2022.

[142] Arun Rajkumar and Shivani Agarwal. A differentially private stochastic gradient descent algorithm for multiparty classification. *Journal of Machine Learning Research*, 22:933–941, 2012.

[143] Thilina Ranbaduge and Peter Christen. Privacy-Preserving Temporal Record Linkage. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 377–386, 2018.

[144] Thilina Ranbaduge, Peter Christen, and Rainer Schnell. Secure and Accurate Two-Step Hash Encoding for Privacy-Preserving Record Linkage. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2020.

[145] Thilina Ranbaduge, Dinusha Vatsalan, Peter Christen, and Vassilios Verykios. Hashing-Based Distributed Multi-party Blocking for Privacy-Preserving Record Linkage. volume 7301, pages 415–427. 2016.

[146] Fang Yu Rao, Jianneng Cao, Elisa Bertino, and Murat Kantarcioglu. Hybrid private record linkage: Separating differentially private synopses from matching records. *ACM Transactions on Privacy and Security*, 22(3), 2019.

[147] Alex Rawson, Ewan Duncan, and Conor Jones. The truth about customer experience. *Harvard business review*, 91(9):90–98, 2013.

[148] Scott Robbins. A misdirected principle with a catch: explicability for ai. *Minds and Machines*, 29(4):495–514, 2019.

[149] Mark Russinovich, Manuel Costa, Cédric Fournet, David Chisnall, Antoine Delignat-Lavaud, Sylvan Clebsch, Kapil Vaswani, and Vikas Bhatia. Toward confidential cloud computing. *Communications of the ACM*, 64(6):54–61, 2021.

[150] S Schinkel, O Dimigen, and Norbert Marwan. Selection of recurrence threshold for signal detection. *The european physical journal special topics*, 164(1):45–53, 2008.

[151] Rainer Schnell. Privacy-preserving record linkage. pages 201–225. 2 2016.

[152] Rainer Schnell, Tobias Bachteler, and Jörg Reiher. Privacy-preserving record linkage using Bloom filters. *BMC Medical Informatics and Decision Making*, 9(1):41, 12 2009.

[153] Rainer Schnell and Christian Borgs. Randomized Response and Balanced Bloom Filters for Privacy Preserving Record Linkage. *IEEE International Conference on Data Mining Workshops, ICDMW*, pages 218–224, 2017.

[154] Rainer Schnell, Christian Borgs, and Filter-based Encryptions. XOR-Folding for Bloom Encryptions for Record Linkage. 2016.

[155] Rainer Schnell, Dorothea Rukasz, Christian Borgs, Stefan Brumme, William B Brogden, Tim O'Brien, Stephen Lacy, and Maintainer Dorothea Rukasz. R PPRL TOOLBOX, 2018.

[156] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[157] Jordi Soria-Comas and Josep Domingo-Ferrer. Big Data Privacy: Challenges to Privacy Principles and Models. *Data Science and Engineering*, 1(1):21–28, 2015.

[158] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[159] Latanya Sweeney. Weaving technology and policy together to maintain confidentiality. *The Journal of Law, Medicine & Ethics*, 25(2-3):98–110, 1997.

[160] Latanya Sweeney. Simple demographics often identify people uniquely. *Health (San Francisco)*, 671(2000):1–34, 2000.

[161] LATANYA SWEENEY. k-ANONYMITY: A MODEL FOR PROTECTING PRIVACY. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 10 2002.

[162] Fengyi Tang, Wei Wu, Jian Liu, Huimei Wang, and Ming Xian. Privacy-preserving distributed deep learning via homomorphic re-encryption. *Electronics (Switzerland)*, 8(4), 2019.

[163] Marco Thiel, M Carmen Romano, and Jurgen Kurths. Analytical description of recurrence plots of white noise and chaotic processes. *arXiv preprint nlin/0301027*, 2003.

[164] Marco Thiel, M Carmen Romano, Jürgen Kurths, Riccardo Meucci, Enrico Allaria, and F Tito Arecchi. Influence of observational noise on the recurrence quantification analysis. *Physica D: Nonlinear Phenomena*, 171(3):138–152, 2002.

[165] Saravanan Thirumuruganathan, Shameem A Puthiya Parambath, Mourad Ouzzani, Nan Tang, and Shafiq Joty. Reuse and Adaptation for Entity Resolution through Transfer Learning. 2018.

[166] Eliza Rebeca Tomodan and Cătălin Daniel Căleanu. Bag of features vs deep neural networks for face recognition. *2018 International Symposium on Electronics and Telecommunications (ISETC)*, pages 1–4, 2018.

[167] Juan Ramón Troncoso-Pastoriza, Stefan Katzenbeisser, and Mehmet Celik. Privacy preserving error resilient DNA searching through oblivious automata. *Proceedings of the ACM Conference on Computer and Communications Security*, pages 519–528, 2007.

[168] Tien Tuan, Anh Dinh, Rui Liu, Meihui Zhang, Gang Chen, Tien Tuan Anh Dinh, Rui Liu, Meihui Zhang, Gang Chen, Beng Chin Ooi, and Ji Wang. Untangling blockchain: A data processing view of blockchain systems. *IEEE Transactions on Knowledge and Data Engineering*, 30(7):1366–1385, 2018.

[169] Hayat Ullah, Bashir Ahmad, Iqra Sana, Anum Sattar, Aurangzeb Khan, Saima Akbar, and Muhammad Zubair Asghar. Comparative study for machine learning classifier recommendation to predict political affiliation based on online reviews. *CAAI Transactions on Intelligence Technology*, 6(3):251–264, 2021.

[170] Dinusha Vatsalan. *Scalable and Approximate Privacy-Preserving Record Linkage*. PhD thesis, 2014.

[171] Dinusha Vatsalan, Dimitrios Karapiperis B, and Aris Gkoulalas-divanis. *An Overview of Big Data Issues in Privacy-Preserving Record Linkage*, volume 2. Springer International Publishing, 2019.

[172] Dinusha Vatsalan and Peter Christen. Multi-Party Privacy-Preserving Record Linkage using Bloom Filters. 12 2016.

[173] Dinusha Vatsalan, Peter Christen, Christine M O'Keefe, and Vassilios S Verykios. An Evaluation Framework for Privacy-Preserving Record Linkage. *Journal of Privacy and Confidentiality*, 6(1):75, 2014.

[174] Dinusha Vatsalan, Peter Christen, and Erhard Rahm. Scalable privacy-preserving linking of multiple databases using counting Bloom filters. In *ICDMW, Barcelona*, 2016.

[175] Dinusha Vatsalan, Peter Christen, and Erhard Rahm. Scalable Multi-Database Privacy-Preserving Record Linkage using Counting Bloom Filters. 2017.

[176] Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Information Systems*, 38(6):946–969, 2013.

[177] Dinusha Vatsalan, Peter Christen, and Vassilios S Verykios. Ef fi cient Two-Party Private Blocking based on Sorted Nearest Neighborhood Clustering. pages 1949–1958, 2013.

[178] Dinusha Vatsalan, Dimitrios Karapiperis, and Vassilios S Verykios. Privacy-Preserving Record Linkage. (January), 2018.

[179] Dinusha Vatsalan, Ziad Sehili, Peter Christen, and Erhard Rahm. Privacy-Preserving Record Linkage for Big Data : Current Approaches and Research Challenges. In *Big Data Handbook*. Springer, 2016.

[180] Anushka Vidanage, Peter Christen, Thilina Ranbaduge, and Rainer Schnell. A Graph Matching Attack on Privacy-Preserving Record Linkage. *Int. Conf. Inf. Knowl. Manag. Proc.*, pages 1485–1494, 2020.

[181] Anushka Vidanage, Thilina Ranbaduge, Peter Christen, and Rainer Schnell. Efficient Pattern Mining based Cryptanalysis for Privacy-Preserving Record Linkage. *Proceedings - International Conference on Data Engineering*, pages 1698–1701, 2019.

[182] Anushka Vidanage, Thilina Ranbaduge, Peter Christen, and Rainer Schnell. A taxonomy of attacks on privacy-preserving record linkage. *Journal of Privacy and Confidentiality*, 12(1), 2022.

[183] Isabel Wagner and David Eckhoff. Technical privacy metrics: a systematic survey. *ACM Computing Surveys (CSUR)*, 51(3):1–38, 2018.

[184] Wei Wang, Meihui Zhang, and Gang Chen. Database Meets Deep Learning : Challenges and Opportunities. pages 17–22.

[185] Lifei Wei, Haojin Zhu, Zhenfu Cao, Xiaolei Dong, Weiwei Jia, Yunlu Chen, and Athanasios V. Vasilakos. Security and privacy for storage and computation in cloud computing. *Information Sciences*, 258:371–386, 2014.

[186] Karl Weiss, Taghi M. Khoshgoftaar, and Ding Ding Wang. *A survey of transfer learning*, volume 3. Springer International Publishing, 2016.

[187] Jiasi Weng, Jian Weng, Jilian Zhang, Ming Li, Yue Zhang, and Weiqi Luo. Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive. *IEEE Transactions on Dependable and Secure Computing*, 18(5):2438–2455, 2019.

[188] William E Winkler. Methods for Record Linkage and Bayesian Networks. *Proceedings of the Section on Survey Research Methods, American Statistical Association*, (Jasa 1969):29, 2002.

[189] GAVIN(ETHEREUM) WOOD. Ethereum: a Secure Decentralised Generalised Transaction Ledger. *Ethereum Project Yellow Paper*, 32(10):1365–1367, 2018.

[190] Primer Writing, Team Dixie, Mark Phillips, David Van Enckevort, Peter Christen, Maximilian Haeussler, Cenk Sahinalp, and Adrian Thorogood. Technology Primer : Overview of Technological Solutions to Support Privacy-Preserving Record Linkage. pages 1–23, 2017.

[191] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133, 2019.

[192] Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. *Transfer Learning*. Cambridge University Press, 2020.

[193] Heung Youl Youm. An overview of de-identification techniques and their standardization directions. *IEICE TRANSACTIONS on Information and Systems*, 103(7):1448–1461, 2020.

[194] Diego Zardetto, Monica Scannapieco, and Tiziana Catarci. Effective automated object matching. *Proceedings - International Conference on Data Engineering*, pages 757–768, 2010.

[195] Joseph P Zbilut, Alessandro Giuliani, and Charles L Webber Jr. Detecting deterministic signals in exceptionally noisy environments using cross-recurrence quantification. *Physics Letters A*, 246(1-2):122–128, 1998.

[196] Fuzhen Zhuang, Keyu Duan, Tongjia Guo, Yongchun Zhu, Dongbo Xi, Zhiyuan Qi, and Qing He. Transfer Learning Toolkit: Primers and Benchmarks. pages 17–23, 11 2019.

[197] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A Comprehensive Survey on Transfer Learning. 11 2019.

Appendix A

Appendix

A.1 Notation Summary

Symbol	Description
PPRL	
P	Participant of PPRL
$\|P\|$	number of participants
e	Entity
$\hat{e}$	Anonimyzed entity
D_p	Dataset of participant p
$\hat{D}_p$	Anonimyzed dataset of participant p
Ξ	Decision model
Bloom Filter	
l	Bloom filter length
n	set of elements (q-grams) to be inserted in BF
k	Bloom filter hash functions
b_m	Value of bit m in the filter
Spliting Bloom Filter	

s	Number of splits
ϕ	Split

3PAC/ABEL

Ψ	Blockchain public readable database
α	Alpha threshold
β	Beta threshold ($\beta = \alpha - error$)
ζ	List of entities (ids) pairs with their similarity values
φ	A set of splits (ϕ)

AT-UC

D_{target}	Target Dataset
D_{source}	Source Dataset
$T_{source/target}$	Target/Source Learning Task

DLC

$\bar{D}_a, \bar{D}_b$	Sub-dataset of the original
Υ_s	sub-dataset sample size in percentage
Υ_m	Percentage of non-matching isntance in the sub-dataset
α	CRP threshold
m	CRP neighbors
Θ	Heaviside function
ϵ	privacy budget

A.2 Limitation: Blockchain-based privacy-preserving record linkage

The "Blockchain-based Privacy-Preserving Record Linkage – Enhancing Data Privacy in an Untrusted Environment" (BC-PPRL) uses Blockchain technology to provide accountability to the computation performed during the comparison step of PPRL. The BC-PPRL utilizes small fragments (splits) of the encoded records to iterative compute the similarity of the records and classify them into matches and non-matches, without sharing the complete information of the encoded records. Christen et al. propose a novel attack that leverages the exchanged information by the BC-PPRL. In this section, we acknowledge the Christen et al. findings and provide a detailed explanation of how the privacy of BB-PPRL could be compromised. We also make available a simplified version of the BC-PPRL, the datasets, and version (ported to python 3) of the attack that could be executed in the google cloud environment at: https://github.com/thiagonobrega/bcpprl-simplified.

A.2.1 Introduction

In recent times, companies and government significantly increased the amount of collected data. Much of this data is about personal information, such as shopping transactions, browsing history, telecommunication records, financial information, or electronic health records. This data has been employed in data mining and analytic techniques that can provide relevant information for several areas of knowledge. For instance, personal data can i) be employed to perform crime and fraud detection [176], ii) lead to better patient outcomes or to detect a disease outbreak in the health sector [10], iii) be of vital importance to national security [179] or iv) be a competitive edge to a commercial enterprise [30].

Data mining and analysis often require information from multiple data sources to be integrated in order to enable precise and useful analysis [10]. However, to execute data integration, first, we have to identify and aggregate records that relate to the same entity (e.g., people, restaurants, publications, products, among others) from one or more data sources [30]. This process is known as Record Linkage (RL), Data Matching (DM), or Entity Resolution (ER) [30; 10]. Although the process receives several names in the literature, in this work, we will adopt RL.

The RL process is composed of four major steps. The first one is data pre-processing, which ensures the data from several data sources are in the same format. The second step, indexing, intends to reduce the number of comparisons performed by selecting record pairs to be matched (compared)

in the subsequent step. In the third step, the actual record pair comparison occurs. In the comparison step, each record pairs receives a similarity value. These pairs are compared using various attributes (for a person, it can include name, sex, and age) and comparison functions. Finally, in the last step (classification), the record pairs are classified into matches, non-matches, or potential matches, depending on the decision model used [10].

A recurring problem that Record Linkage faces is the absence of attributes capable of uniquely identifying records, which refer to the same entity, in the different data sources. The absence of a unique identifier, such as an ID, makes the use of simple comparison operations (e.g. SQL joins) impossible, making the linkage to be carried out with sophisticated comparisons involving a set of common attributes to all records in the different data sources. Such a set of attributes is called quasi-identifiers (QIDs) [30].

Currently, Record Linkage not only faces computational and operational challenges intrinsic to the comparison and classification methods, but it also has to address privacy preservation challenges due to recent laws and regulations such as European General Data Protection Regulation (GDPR), Brazilian General Data Protection Law (LGPD) and the US HIPAA Privacy Rule. In this context, Privacy-Preserving Record Linkage (PPRL) emerges, aiming to identify matching records across private data sources, ensuring that the data's privacy and confidentiality are preserved throughout the linkage process.

In order to address privacy-related issues, the basic idea of Privacy-Preserving Record Linkage (PPRL) is to execute the linkage process in anonymized data (by perturbing the original data with the use of encryption, hash functions, and noise additions), ensuring that the privacy and confidentiality of the data are preserved during the linkage process. PPRL reveals only a limited amount of information. For instance, a party only knows which of its own records exist in the other party's data source or the number of duplicated records presented in the datasets used as input to the PPRL process [179]. Moreover, a PPRL solution should address efficiency, linkage quality, and privacy characteristics. Furthermore, the PPRL solution needs to provide a comprised among privacy, efficiency, and quality according to the needs of the PPRL parties' requirements.

In this context, the Blockchain-based Privacy-Preserving Record Linkage (BC-PPRL) [130] introduces a novel protocol that enables the auditability of the computations performed by the PPRL parties. Furthermore, the BC-PPRL compares the records iteratively (using a fraction of the original information) to audit the computation performed by different parties and reduce the information shared during the PPRL process. A filtering process explains the information reduction in the BC-PPRL. In other words, in each iteration, records pairs (represented by a fraction of the origi-

nal information) that present a similarity value below a predefined threshold are removed from the comparison step of PPRL.

Christen et al. proposed a novel attack that leverages the exchanged information during the BC-PPRL execution to re-identify the encoded records. The attack proposed by Christen et al. can successfully re-identify records using small piece information.

Thus, considering the novel attack, this work has the objective of acknowledging the limitation of the BC-PPRL and clarifying in which circumstances the attack will succeed. Moreover, this work intends to make clear to other researchers how the BC-PPRL can (or can not) be safely used.

Outline. In Section A.2.2, we provide details regarding BC-PPRL. Section A.2.3 details the limitations of BC-PPRL. Finally, Section A.2.4 introduces our final arguments.

A.2.2 Blockchain-based Privacy-Preserving Record Linkage

This section explains the assumptions and intuitions employed to design the BC-PPRL. Thus, before presenting the assumptions, it is important to remember that the BC-PPRL introduces a novel protocol that enables the auditability of the computations performed by the PPRL parties, named Auditable Blockchain-based PPRL (ABEL). To implement the protocol, we employ a decentralized environment (Blockchain networks) where untrusted (or semi-trusted) parties perform the computations required by PPRL.

It is important to mention that the Blockchain does not provide a mechanism to preserve the privacy of the records during the PPRL process. Actually, the Blockchain reduces the privacy of PPRL by replicating the entire data amongst the untrusted parties. To overcome this limitation, we introduced the Splitting Bloom Filter (SBF), an improvement for the most prevalent anonymization technique used in PPRL applications: Bloom Filter (BF) [152]. SBF enables an iterative comparison of the BF similarity by breaking the original BF in splits. It is important to remark that each BF is the anonymized representation of one record.

The basic idea of ABEL is to iterative execute the comparison step of PPRL by employing small segments (splits) of the original BF. In each iteration, the split similarity will be used to validate the similarity computations and filter the BFs that will be forwarded to the next iteration. In other words, BFs that present a similarity value below a predefined classification threshold are removed from the protocol in each iteration. Moreover, this filtering process could reduce the amount of information regarding BFs that probably will not be classified as a match from being shared during the linkage process.

The iterative comparison step, the filtering process, and the use of small splits of the BFs can difficulty BF re-identification, resulting in privacy-enhancing to the PPRL process. It is important to remark that, in the BC-PPRL, i) a significant part of the BF is removed from the PPRL execution (due to the filtering process), and ii) the filtering process is conducted over a small fraction of the information.

By removing BFs, in the early stages of the protocol, it is expected that privacy attacks face an additional challenge to re-identify BFs using a fraction of the original information, for the BFs that probably will not be classified as a match. Regarding the BFs that probably will be classified as matches, the BC-PPRL will forward all splits.

In order to explain why the share of a fraction of the original could improve the privacy of PPRL, in the following, we detail the filtering process performed by ABEL. Furthermore, we detail two important characteristics of ABEL: i) the indistinguishability and ii) the uncertainty about the splits.

ABEL filtering process

As previously argued, ABEL removes (i.e., filters) BFs from the PPRL process that probably will not be classified as a match. This filtering process occurs because ABEL only shares the splits of a BF if the similarity is above a certain threshold to the other records. This filtering process is important to reduce the amount of information shared during PPRL.

To illustrate how much information (split) is filtered (retained) by ABEL in each iteration, we used a sample of the NCVR dataset to count the number of BF retained in each iteration of the ABEL execution. Figure 1 illustrates the filtering process over splits with 3.125% (32bits) of the original BF length. To provide a realistic scenario, we use standard blocking to define the computation that will be considered during PPRL. By employing the standard blocking, we reduce the number of total computations in the first iteration (using the first split) by 80%.

It is important to mention that, to plot Figure 1, we do not use the pair of BFs (or records). Instead, we count the number of BFs pruned from the PPRL by our protocol. Therefore, Figure 1 represents the number of BFs removed in each iteration.

In Figure 1, it is possible to notice that more than 65% of the BFs are filtered in the first iteration. In other words, for 65% of the BFs in the original dataset, we share only 3% of the original BF (information). This number represents a very small fraction of the original information, and we estimate that it refers to no more than three bigrams of the original information. Moreover, only the splits that are more likely to be matched in the subsequent iterations are shared. It is worthwhile to mention that ABEL filters more than 90% of the BFs until the 8th iteration.

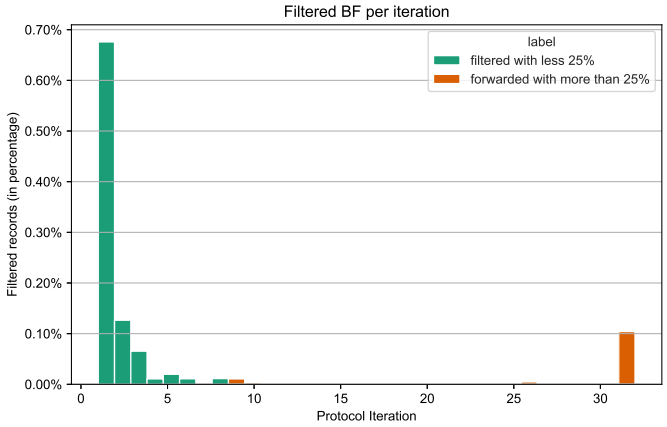


Figure A.1: The number of filtered records, considering similarity threshold = 0.4 and 0.05 error.

In summary, depending on the BC-PPRL parameters (e.g., threshold, error, and split length), the filtering process can reduce the shared information of records that will not be classified as match by the PPRL process.

Indistinguishability

The indistinguishability could be employed as a privacy metric to indicate whether the adversary can distinguish between two items of interest (such as recipients of a message or sensitive attributes in a database) [183].

In order to explain how ABEL and SBF can promote a degree of indistinguishability, consider that a dataset (D_1) contains three distinct BFs: A=101111, B=101110, and C=110011. The owner of D_1 intends to execute the PPRL process with D_2, which contains only one BF, T=110011. If an observer receives all BFs from D_1, he or she can recognize that these filters represent three distinct entities. In Figure A.2, we illustrate how ABEL and SBF increase the indistinguishability until a certain point.

Notice in Figure A.2 that the BFs are split into three parts. Also, ABEL shares one split per iteration if the split satisfies the adjusted threshold (α) defined by the PPRL parties. In the first iteration, the first splits of all records are shared. Notice that the first split of A and B are the same, making it hard to distinguish the origin of the splits. Moreover, due to the difference of the first split of A and B (11) to the first split of T (01), the remaining splits of A and B are not shared in the subsequent iterations.

It is worth mentioning that, at the end of the ABEL execution, an observer receives a reduced amount of information from the filtered records. In Figure A.2, the colored splits represent the avail-

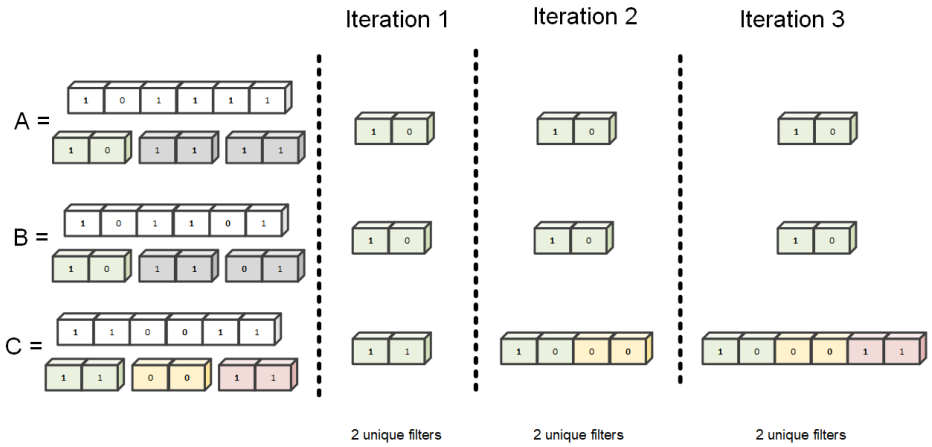


Figure A.2: BF indistinguishability over ABEL iterations

able information to an observer. Notice that this observer knows all bits of C. Therefore, for filter C, no indistinguishability is provided. However, for BFs A and B, the observer has limited information, i.e., the observer may infer that A and B share the same bigram. Furthermore, the observer doesn't know for sure if one split is relative to one or multiples records. This raises uncertainty about the cardinality of the record and the origin of the split. Briefly, the example illustrates how ABEL and SBF promote a degree of indistinguishability to PPRL.

To illustrate the indistinguishability in a real-world dataset, we plotted, in Figure A.3, the number of unique split values over the NCVR dataset considering different combinations of attributes and splits. It is important to remark that the BF parametrization is exposed in Table A.2, and all BF depicted in Figure A.3 consider the same false-positive rate.

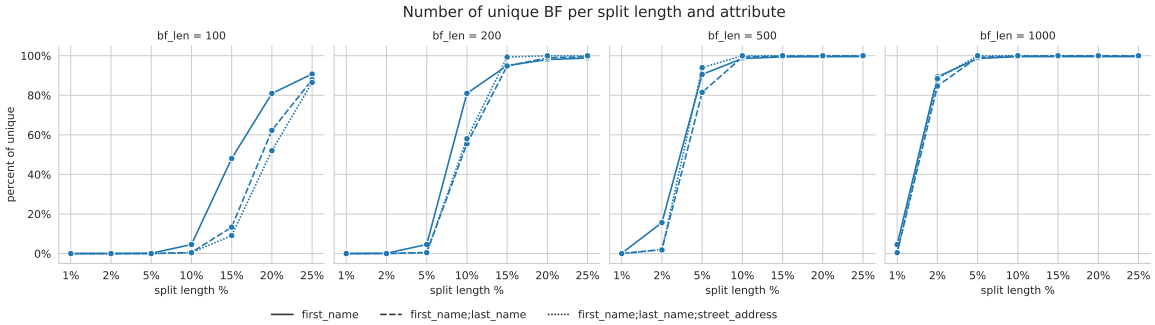


Figure A.3: Number of unique values BF per split length.

The indistinguishability in the NCVR dataset can be noticed in the BF with 100 bits. On average, a split with 10% of the original BF represents 10 BF. Therefore, because each split represents multiples records, an attacker will have an extra challenge to distinguish records represented by the same split.

Figure A.3 also shows that the original BF length (l), split length (s), and the dataset characteristic

(e.g., the attributes values diversity) are important to the indistinguishability. In summary, a bigger BF (i.e., l=1000 bits) that requires a significant number of hash functions to represent each n-gram (i.e., k¿100) demands a split with 1% of the original BF length to reduce the number of unique splits values. Thus, while we reduce the original BF length, it is possible to observe a reduced number of unique values over bigger splits.

Uncertainty

Another possible outcome of using a fraction of the original information to represent a record is an increase in uncertainty. Uncertainty and indistinguishability are highly correlated in the PPRL context; however, they are different. Indistinguishability investigates whether the adversary is able to distinguish between two outcome mechanisms [183]. Indistinguishability metrics are usually binary; they indicate whether two outcomes are indistinguishable or not but do not quantify the privacy levels in-between.

Uncertainty normally considers the probability/estimation that the adversary makes about the information. These metrics assume that high uncertainty from an adversary is associated with high privacy because the adversary cannot base his/her guesses on information known with certainty [183]. However, it is worthwhile to mention that even guesses with complete uncertain information can be correct. This results in privacy loss even in scenarios with a highly uncertain adversary.

Ordinarily, uncertainty is measured by employing the Entropy ($H(x)$) to measure the amount of information and therefore quantify the uncertainty of the information. Cross-Entropy was used in the Privacy-preserving Distributed Clustering [114] to measure the uncertainty from the clustered data, and therefore measure the privacy of the information stored in a cluster.

Cross-Entropy measures the amount of information needed to identify an object in the dataset if the original data is coded in terms of the distribution Q, rather than their true distribution P. Cross-Entropy can be expressed using the entropy of P and the KL-distance, such that:

$$CE(P,Q) = H(P) + KL(P||Q) \tag{A.1}$$

It is important to remark that the KL-distance (a.k.a., relative entropy) measures how one probability distribution differs from a second distribution. The KL-distance is defined as:

$$KL(P||Q) = \sum_{x \ni X} P(x) \times log\left(\frac{P(x)}{Q(x)}\right) \tag{A.2}$$

such that X represents a set of entities (discrete random variable) that can be encoded with different probabilities P(X) and Q(X).

Merugu and Ghosh [114] argued that there is a linear dependency between CE and KL-distance. They employ the KL-distance to estimate uncertainty and, therefore, the privacy in the context of their work. We are able to use the same idea to measure the uncertainty produced by BC-PPRL. Thus, consider that P is the original distribution of bit patterns in BFs and Q is the distribution of bit patterns in splits, i.e., the first split used by Semi-Trusted Third Party (STTP).

It is important to explain how KL-distance is interpreted as an uncertainty metric. In summary, a $KL(P||Q) = 0$ indicates that the two distributions in question have identical quantities of information. Thus, considering that high uncertainty provides high privacy [183; 114], lower KL-distance means low uncertainty and, therefore, less privacy.

In Section A.2.3 we correlate the uncertainty, indistinguishability, and the filtering process on the results achieved by Christen et al.

A.2.3 BC-PPRL Limitation

In this section, we intend to make clear the limitations exposed in BC-PPRL [130]. Moreover, we intend to confirm and acknowledge the results of Christen et al.

In Section 8 of BC-PPRL [130], there is a discussion regarding the limitation of BC-PPRL. It is argued that BC-PPRL faces challenges regarding execution time (efficiency) and privacy. Regarding the efficiency, it was demonstrated that BC-PPRL has an elevated execution time due to the usage of Blockchain networks. On average, each Blockchain computation was 20 times slower than the regular computation. Moreover, knowing that ABEL requires multiple iterations of the protocol, the execution of the PPRL process using BC-PPRL could be 20 to 200 times slower than a standard PPRL approach.

Concerning the privacy capabilities in Section 8.2.4, it was explicitly stated that *"... we believe that our approach is unable to preserve the privacy of the entities when the parties choose inappropriate parameters..."*. Section 8.2.5 presented a comprising between the BC-PPRL parameters and the efficiency, privacy, and linkage quality. Moreover, it was explained that bigger splits negatively impact privacy, while smaller splits have a positive impact on privacy.

Christen et al. demonstrate this privacy limitation. Furthermore, Christen et al. evidence that BC-PPRL could reveal a significant amount of information. In the following sections, we comment on the result of the novel attack, the *pprlSegmentAtomAttack*. We also present arguments to make clear in which case the aforementioned attack is more likely to succeed. Finally, we compare the *pprlSegmentAtomAttack* result against other attacks.

Comments on *pprlSegmentAtomAttack* results

The results reported by Christen et al. demonstrated that BC-PPRL reveals a significant amount of information when more than 5% of the BF (with 1,000 bits) are shared. Moreover, Christen et al. test the attack considering different combinations of encoded attributes, number of hash (k) functions, and segment length(s).

In the tested scenarios, the capabilities of BC-PPRL are limited. Moreover, BC-PPRL showed a discrete improvement on privacy when a non-optimal BF occupation (fewer hash functions) is employed. It is worthwhile to mention that, ideally, a BF needs to have 50% of its bits set to 1 to reduce the collision probability of a BF element (e.g., n-gram) [30].

However, we believe that this experiment has an issue related to the amount of information used to represent each bi-gram. To provide a simple illustration of our argument, let's consider that each character represented by ASCII (or UTF-8) has 8 bits; therefore, a bi-gram is represented by 16 bits. For instance, in the experiment, each bigram of the attributes 'First name' and 'First Name + Last Name' is represented by 123 bits and 57 bits, respectively. This encoding represents an increase of information from seven to three times of the original information. Entropy-based metrics of the encoded values could indicate this extra information.

This argument is corroborated by the results shown in Table 3 generated by Christen et al. Table 3 shows that, when the occupation of BF diminishes (smaller k), the attack reduces the number of 1-to-1 correct re-identification rates. It is important to remark that the extra information does not change the findings of Christen et al.; instead, it just modifies the limits that BC-PPRL is able to work properly.

To provide a more precise definition of the BC-PPRL limitation, we re-executed the attack employing a sample with 1,000 records of the dataset used in the experiment of Christen et al. We also consider different BF lengths with an optimal occupation of BF (50% of bits set 1).

Table A.2 shows the re-identification results obtained with *pprlSegmentAtomAttack* on the same three attribute combinations employed in the original attack. In this attack, we randomly selected 1,000 records of one dataset to be attacked and used all records of the other dataset (224,073 records) as the source of information. It is worth mentioning that we consider the amount of shared information (si) instead of considering BF segment length(s). For instance, when we consider that 10% of the BF is shared, it could be yielded in one split of 10%, two splits of 5%, or ten splits of 1%. The results reported in Table A.2 are the mean value of 10 rounds of re-identifications.

Table A.2: Counts of the five different outcomes of the atom-based attack on 1,000 randomly selected BF splits for different encoded attribute combinations. Each row shows the results of 10 repletion's of the attack.

Encoded attributes	l	shared information	k	1-to-1 correct	1-to-many correct	1-to-1 wrong	1-to-many wrong	No matches
first name + last name + street address	1000	25%	24	998	2	0	0	0
		20%	24	996 ±1	3 ±1	0 ±1	0	0
		10%	24	327 ±28	117 ±17	386 ±6	170 ±20	0
		5%	24	3 ±2	6 ±2	756 ±31	234 ±27	0
		2%	24	0	0	910	90	0
	500	25%	12	931 ±7	33 ±10	29 ±4	6	0
		20%	12	606 ±8	121 ±7	199 ±12	74 ±11	0
		10%	12	7 ±2	7 ±2	665 ±10	320 ±11	0
		5%	12	0	0	816 ±10	184 ±10	0
		2%	12	0	0	1000	0	0
	200	25%	5	15 ±2	10 ±4	672 ±17	302 ±16	0
		20%	5	0 ±1	3 ±2	728 ±17	269 ±17	0
		10%	5	0	0	795 ±21	205 ±21	0
		5%	5	0	0	966 ±5	34 ±5	0
		2%	5	0	0	1000	0	0
	100	25%	2	0	0	865 ±12	135 ±12	0
		20%	2	0	0 ±1	788 ±11	211 ±11	0
		10%	2	0	0	998 ±2	2 ±2	0
		5%	2	0	0	1000	0	0
		2%	2	0	0	1000	0	0
first name + last name	1000	25%	57	936 ±6	64 ±6	0	0	0
		20%	57	933 ±4	67 ±4	0 ±1	0	0
		10%	57	862 ±6	119 ±5	17 ±2	2 ±2	0
		5%	57	84 ±3	120 ±11	443 ±12	354 ±7	0
		2%	57	0	0	593 ±24	407 ±24	0
	500	25%	28	914 ±10	82 ±9	2 ±1	1 ±1	0
		20%	28	860 ±6	124 ±6	13 ±3	3 ±2	0
		10%	28	110 ±9	161 ±9	399 ±1	331	0
		5%	28	0 ±1	2 ±1	550 ±16	447 ±16	0
		2%	28	0	0	494 ±8	506 ±8	0
	200	25%	11	144 ±2	184 ±11	364 ±15	307 ±7	0
		20%	11	28 ±3	60 ±5	450 ±16	462 ±19	0
		10%	11	0	0	526 ±9	474 ±9	0
		5%	11	0	0	562 ±10	438 ±10	0
		2%	11	0	0	325 ±7	675 ±7	0
	100	25%	6	0 ±1	3 ±1	572 ±12	424 ±13	0
		20%	6	0 ±1	1	565 ±7	433 ±7	0
		10%	6	0	0	560 ±10	440 ±10	0
		5%	6	0	0	443 ±5	557 ±5	0
		2%	6	0	0	789 ±1	211 ±1	0
first name	1000	25%	123	477	523	0	0	0
		20%	123	460	540	0	0	0
		10%	123	469	529	2	0	0
		5%	123	291	547	99	63	0
		2%	123	5	18	403	574	0
	500	25%	61	485	515	0	0	0
		20%	61	468	525	5	2	0
		10%	61	309	493	107	91	0
		5%	61	19	91	342	548	0
		2%	61	0	0	425	575	0
	200	25%	25	338	552	64	46	0
		20%	25	212	488	158	142	0
		10%	25	5	41	385	569	0
		5%	25	0	1	381	618	0
		2%	25	0	0	457	543	0
	100	25%	12	12	80	353	555	0
		20%	12	5	19	427	549	0
		10%	12	0	0	522	478	0
		5%	12	0	0	697	303	0
		2%	12	0	0	850	150	0

As shown in Table A.2, the re-identification results to the BF with 1,000 bits are equivalent to the result reported by Christen et al. However, when we reduce the number of bits used to represent each n-gram (denoted by k), it is possible to notice that re-identification rate diminishes. It is worthwhile to remark that all tested BF consider a 50% occupation rate of the filter.

To illustrate the re-identification rate, consider the two distinct BF configuration BF with 1,000 bits and 100 bits regarding the attributes first and last name. Notice that for the 1,000 bits BF, with 25% of the original information, the attack was able to re-identify 99% of the records. However, for the 100 bits BF, the attack was unable to re-identify none of the records. We believe that these results can be explained by the amount of information used to represent the n-grams.

As segments get shorter, less correct 1-to-1 assignments can be identified regarding the BF length, which is expected as the number of unique bit patterns in BF segments gets smaller and, therefore, less accurate atom assignments are possible.

It is important to emphasize that the amount of information required to correctly and uniquely identify a substantial number of the plain-text values encoded into the BFs of the other database owner depends on the configuration of the original BF. However, regardless of the original BF length, attacks will successfully re-identify the records when a certain amount of information is shared.

Thus, the amount of information required to correctly and uniquely re-identify a substantial number of records depends on several parameters, such as the encoded attributes and BF parameters. For instance, considering the results shown in Table A.2, related to the attributes First and Last name encoded with a BF with 1,000 bits, if 10% of the original BF are shared, almost all records can be re-identified. However, considering a BF with 100 bits, when 25% of the original BF is shared, less than 1% of the records are re-identified.

In order to provide a clear definition of the BC-PPRL privacy limitation, in the next section, we present metrics that could be used to estimate the amount of information needed to re-identify the encoded records.

BC-PPRL privacy limitation

As argued in the previous section, BC-PPRL could reveal a significant amount of information depending on several parameters. This section ratifies Christen et al. finding, that the probability of n-gram being hashed in one split does not provide an accurate privacy metric to the encoded record. Furthermore, we provide a discussion that could indicate a more suitable method to estimate the

privacy limitation of BC-PPRL.

BC-PPRL [130] correlates the probability $P(ng)$ of all encoded bits of an n-gram be hashed in the shared information (split) to the chance of successfully re-identifying an encoded record. However, Christen et al. demonstrated that using $P(ng)$ or $P(\exists h_i(ng) \ni \phi, 1 \leq i \leq k)$ is not a precise method to estimate the re-identification rate of an attack. Moreover, $P(ng)$ and $P(\exists h_i(ng) \ni \phi, 1 \leq i \leq k)$ fails to predict the chance of re-identification in several cases (e.g., if the original BF uses too much information to represent each n-gram). In summary, the inadequate privacy estimation considering the $P(ng)$ and $P(\exists h_i(ng) \ni \phi, 1 \leq i \leq k)$ is justified by the fact that both metrics do not take into the amount of information used to represent each n-gram.

In order to provide a better indication of the shared amount of information (splits) that reduces the chance of re-identification, we propose the usage of indistinguishability and uncertainty metrics. As presented in Section A.2.2, the indistinguishability could be expressed in terms of the unique values of the shared information (splits). The uncertainty brought by the usage of a fraction of the information to represent could be expressed in terms of relative entropy ($KL(P||Q)$).

It is worthwhile to mention that indistinguishability ($PU(P,Q)$) and uncertainty ($KL(P||Q)$) could be calculated by the database owner. Assuming that P is the original BF and Q is the split (or amount of information), a database owner could calculate the $KL(P||Q)$ and count the percent of unique values ($PU(P,Q)$) that a specific amount of information produces.

Figure A.4 illustrates the relation amongst the aforementioned metrics and the re-identification rate achieved by the *pprlSegmentAtomAttack*. Notice that the vertical axis is used to plot three metrics, the re-identification rate and two privacy metrics ($KL(P||Q)$ and $PU(P,Q)$). The colored lines represent the $KL(P||Q)$ and $PU(P,Q)$, and the grey bars represent the percentage of the re-identified records.

It is worthwhile to mention that we use an interval from 0 to 1 to represent all results. For instance, if a $PU(P,Q)$ or the percentage of re-identified records is 100%, we represent it by 1. To normalize the $KL(P||Q)$ scores between 0 (identical) and 1 (maximally different), we employ the Jensen–Shannon divergence (JS). The Jensen–Shannon divergence is defined by $JS(P||Q) = \frac{1}{2} \times KL(P||M) + \frac{1}{2} \times KL(Q||M)$, where $M = \frac{(P+Q)}{2}$. In other words, $JS(P||Q) = 0$ has the same meaning of $KL(P||Q) = 0$, that the P and Q are similar; therefore, the shared splits have the same amount of information as the original ones.

Regarding the indistinguishability ($PU(P,Q)$), it is possible to observe that, when the shared information has multiple BF, represented by one split (or set of splits), the tested attack is unable to re-identify the encoded records. It is important to mention that in the results depicted in Figure

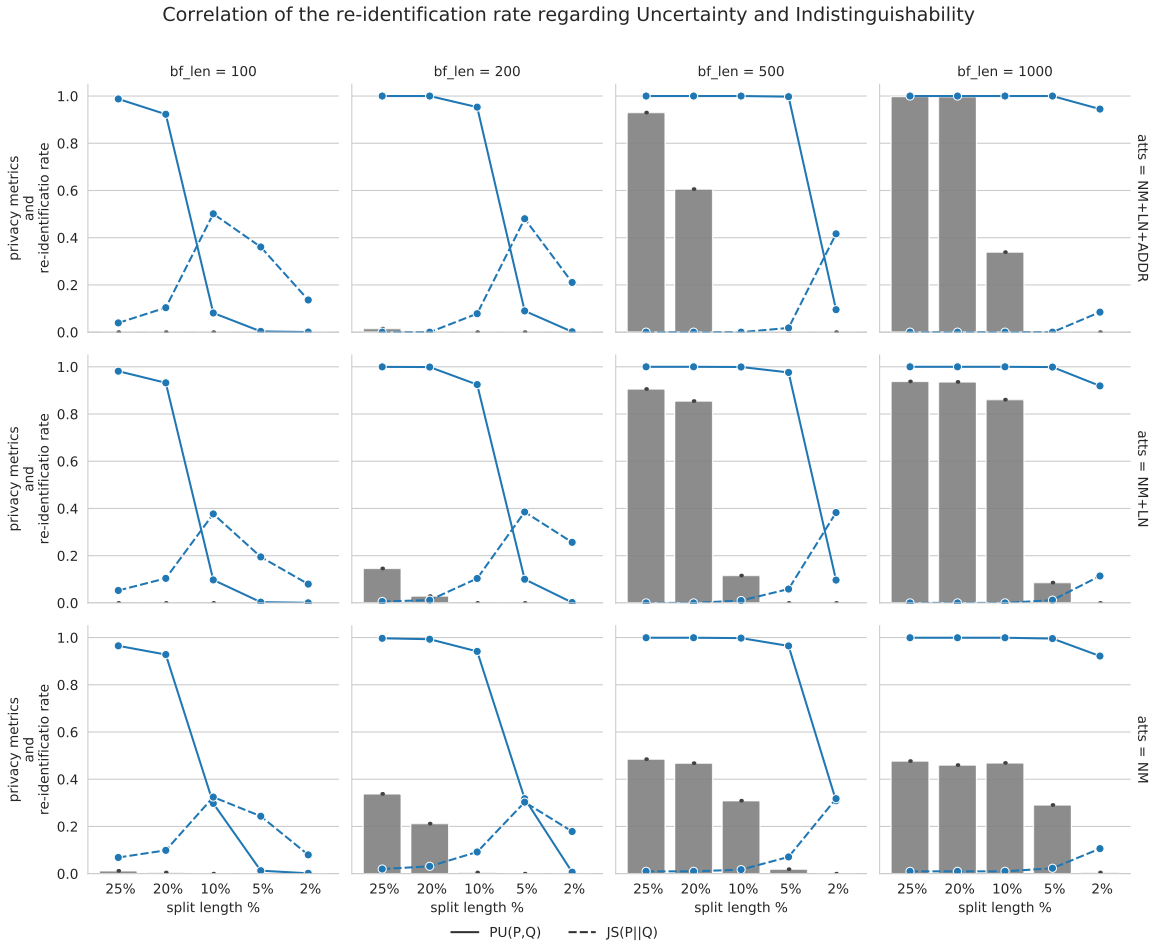


Figure A.4: Re-identification results and privacy metrics for different attribute combinations, BF length, and shared information. The NM, LN, and ADDR represent the First name, Last name, and street address.

A.4 when we have $PU(P,Q) < 95\%$, the attack is unable to re-identify the records, including the BF encoded with 1,000 bits.

We are aware that this attack can be improved, and this result might vary depending on the attack and the data. Therefore, we suggest that the Indistinguishability should be smaller than 50%. In other words, by considering $PU(P,Q) < 50\%$, we ensure that, on average, each split represents two encoded values. Thus, an attacker will be unsure and will have difficulty in distinguishing which original encoded value generates the split, as explained in Section A.2.2.

Regarding the uncertainty, it is possible to observe that when we consider an amount of information (Q) that differs from the original value (P), the attack is unable to re-identify the encoded records. In other words, when the shared split set presents the uncertainty metric bigger than zero, the tested attack is unable to properly re-identify the records.

It is important to comment on the result related to the BF with 200 bits, specifically to the three attributes (First name, Last name, and Street address) when more than 20% of the original information is shared. Notice that, in this scenario, the uncertain metric is zero. We believe that a slight improvement on the attack or a better data source could result in a higher re-identification rate.

In summary, we believe that the combination of indistinguishability and uncertainty metrics could indicate the amount of information that could be shared, depending on the BF parameter. Based on the experimental results, we believe that $PU(P,Q) < .5$ and $KL(P||Q) > 0.1$ provide a suitable configuration parameter to BC-PPRL. However, it is important to mention that further investigation should be conducted to determine the indistinguishability and uncertainty metrics values that should be used. In Section A.2.3, we use the Brazilian Politician Registration to illustrate the re-identification (and the privacy metrics) results considering a different dataset.

Re-identification results considering the complete BC-PPRL

In BC-PPRL [130], the approach was tested considering five attributes against a different attack. Thus, assuming that *pprlSegmentAtomAttack* provides a more accurate test scenario, we employ this attack to re-assess the privacy of the BC-PPRL. This section intends to provide further details on the re-identification rate that an attack could produce when executed over the BC-PPRL outputs. To evaluate the re-identification accuracy, we consider two attributes (first and last names) of a sample with 10,000 records of the NCVR dataset. The plain records values were encoded considering a BF with l=200 bits; the anonymization details are shown in Table A.3.

First, we calculate indistinguishability and uncertainty. The indistinguishability and uncertainty results are described in Table A.3. Notice that the metrics indicate that we can share 20% of the original information. Therefore, considering the indistinguishability and uncertainty, we select a 5% split length (s=10bits) to be used in each iteration of BC-PPRL. In other words, only in the fifth iteration, when 25% of the encoded information is shared, a portion of the encoded records will be re-identified.

As mentioned in Section A.2.2, BC-PPRL filters the anonymized records (BFs) according to a predefined threshold. In Figure A.5, we plotted the amount of records filtered in each iteration considering the pre-agreed threshold ($\alpha = 0.85$) and error ($error = 0.05$).

In each iteration of BC-PPRL, we executed the attack against the shared information. For instance, we executed an attack in the first iteration, considering 5% of all BF. While in the second iteration, we ran the attack over 10% of the original BF to the forwarded records (33% of the remaining encoded records). We executed this attack until the last iteration of the protocol. Table A.4 shows

Table A.3: Privacy metrics

Encoded attributes	l	shared information	k	percent 1-to-1 correct	$PU(P,Q)$	$KL(P\|\|Q)$	$JS(P\|\|Q)$
first name + last name + street_address	1000 bits	25%	24	99%	100%	0.000	0.000
		20%	24	99%	100%	0.000	0.000
		10%	24	33%	100%	0.000	0.000
		5%	24	0%	100%	0.000	0.000
		2%	24	0%	94%	0.023	0.085
	500 bits	25%	12	93%	100%	0.000	0.000
		20%	12	60%	100%	0.000	0.000
		10%	12	0%	100%	0.000	0.000
		5%	12	0%	99%	0.001	0.018
		2%	12	0%	9%	0.505	0.416
	200 bits	25%	5	1%	100%	0.000	0.000
		20%	5	0%	100%	0.000	0.000
		10%	5	0%	95%	0.019	0.078
		5%	5	0%	8%	0.722	0.480
		2%	5	0%	0%	0.121	0.210
	100 bits	25%	2	0%	98%	0.005	0.039
		20%	2	0%	92%	0.035	0.104
		10%	2	0%	8%	0.776	0.501
		5%	2	0%	0%	0.349	0.361
		2%	2	0%	0%	0.051	0.136
first name + last name	1000 bits	25%	57	93%	100%	0.000	0.000
		20%	57	93%	100%	0.000	0.000
		10%	57	86%	100%	0.000	0.000
		5%	57	8%	99%	0.000	0.012
		2%	57	0%	91%	0.044	0.114
	500 bits	25%	28	90%	100%	0.000	0.000
		20%	28	85%	100%	0.000	0.000
		10%	28	11%	99%	0.000	0.010
		5%	28	0%	97%	0.011	0.059
		2%	28	0%	9%	0.421	0.383
	200 bits	25%	11	14%	99%	0.000	0.006
		20%	11	2%	99%	0.000	0.012
		10%	11	0%	92%	0.035	0.104
		5%	11	0%	10%	0.444	0.385
		2%	11	0%	0%	0.188	0.257
	100 bits	25%	6	0%	98%	0.009	0.053
		20%	6	0%	93%	0.037	0.104
		10%	6	0%	9%	0.406	0.377
		5%	6	0%	0%	0.101	0.195
		2%	6	0%	0%	0.018	0.080
first name	1000 bits	25%	123	47%	99%	0.000	0.010
		20%	123	46%	99%	0.000	0.010
		10%	123	46%	99%	0.000	0.010
		5%	123	29%	99%	0.002	0.023
		2%	123	0%	92%	0.037	0.106
	500 bits	25%	61	48%	99%	0.000	0.010
		20%	61	46%	99%	0.000	0.010
		10%	61	30%	99%	0.001	0.017
		5%	61	1%	96%	0.017	0.071
		2%	61	0%	30%	0.298	0.318
	200 bits	25%	25	33%	99%	0.001	0.020
		20%	25	21%	99%	0.003	0.031
		10%	25	0%	94%	0.027	0.092
		5%	25	0%	31%	0.266	0.303
		2%	25	0%	0%	0.087	0.179
	100 bits	25%	12	1%	96%	0.015	0.069
		20%	12	0%	92%	0.031	0.099
		10%	12	0%	29%	0.308	0.324
		5%	12	0%	1%	0.161	0.243
		2%	12	0%	0%	0.018	0.080

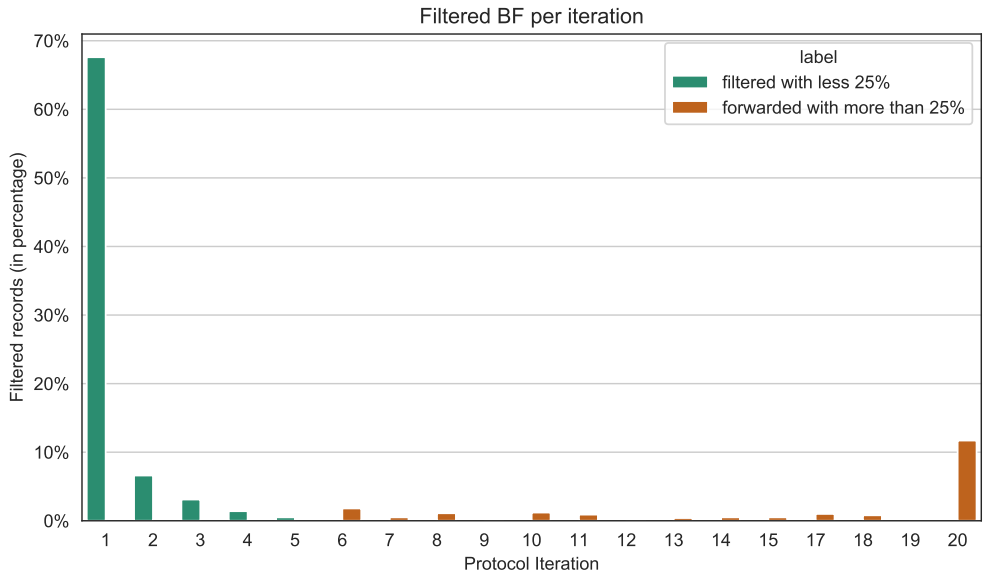


Figure A.5: ABEL filtering process.

the re-identification results achieved in each iteration of the BC-PPRL execution.

Table A.4: Re-identification accuracy per iteration

iteration	shared information	1-to-1 correct	1-to-many correct	1-to-1 wrong	1-to-many wrong	No matches
Iteration 1	5%	1	0	5,588	4,411	0
Iteration 2	10%	1	1	1,855	1,392	0
Iteration 3	15%	8	17	1,157	975	0
Iteration 4	20%	48	137	804	759	0
Iteration 5	25%	214	312	614	450	0
Iteration 6-20	100%	1,346	125	0	0	0

It is worthwhile to remark that from the sixth iteration onwards, we did not execute the attack. Moreover, we consider that all encoded recorded forwarded before 25% could be re-identified. At the end of the BC-PPRL execution, 13% of the records could be re-identified.

It is important to mention that 85% of the re-identified records were classified as match by the classification step (similarity value higher than the predefined threshold). Therefore, even if no records are re-identified by the attack (e.g., in an ideal PPRL process) at the end of the PPRL process, the parties will know that these records are presented in the other databases. In other words, in this attack, a malicious party could re-identify 2% of the records that will not be classified as a match.

To illustrate the indistinguishability and uncertainty metrics on the re-identification accuracy, we plotted the re-identification results and the metrics in Figure A.6. The axis and elements of the figure

are the same of Figure A.4.

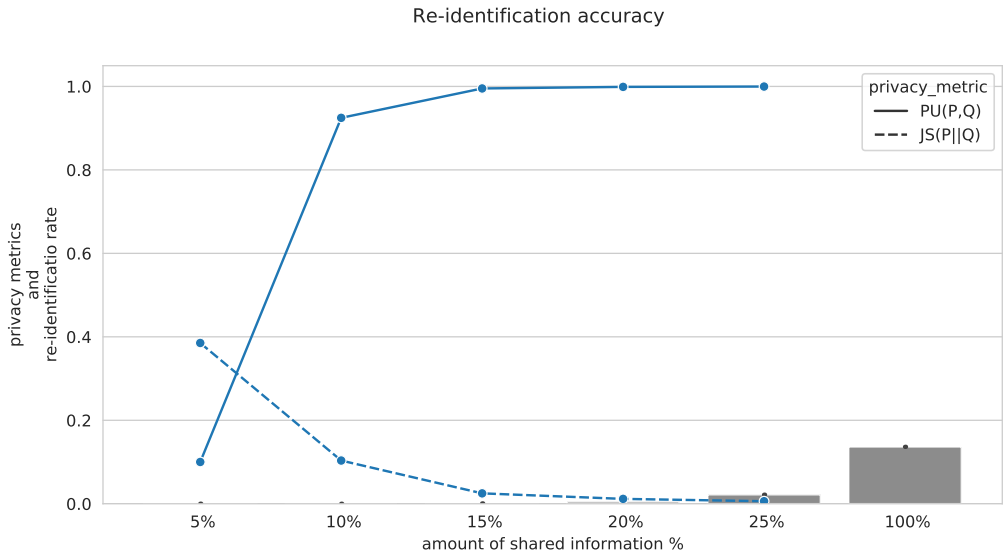


Figure A.6: Number of unique BF values per amount of shared information

Notice that as prognosticated by the indistinguishability and uncertainty metrics, the attack begins to re-identify the encoded records at the fifth iteration.

Comparison of re-identification results with existing BF attack

As mentioned by [130] and Christen et al., BC-PPRL makes available information regarding the anonymization parameters. This information could be employed to create highly effective attacks (as the *pprlSegmentAtomAttack*). It is worth mentioning that highly effective attacks exist [35; 31; 180], even in scenarios where a third party is involved. In these scenarios, the third party is unaware of the anonymization parameter employed and faces different challenges in re-identifying the encoded records.

In order to provide a clear measurement of the BC-PPRL limitation to the PPRL process, we compare the result shown in Section A.2.3 against the result reported in recent BF re-identification attacks [32; 35; 31; 180]. The results reported in Section A.2.3 are compared against an attack that does not consider information regarding the anonymization parameters.

Table A.5 shows the re-identification results of different attacks. It is important to mention that the majority of the compared attacks assume that the third party has different conditions to be properly executed, such as i) guess certain parameters used during BF encoding (e.g., BF length), ii) guess the domain of the sensitive databases depending on its knowledge about the database owners, and iii)

Table A.5: Comparison of re-identification results

Publication	**Dataset**	**Num BF**	**1-to-1 correct**	**1-to-1 correct %**
[35]	NCVR diff. attr.	>200k	>49k	25%
[31]	NCVR First + Last name	>200k	-	20.7%
[180] (with out sim adjust)	NCVR First + Last name	100k	-	>50% accuracy
[180] (with sim adjust)				>90% accuracy
this work	NCVR First + Last name	10k	1,346	13%

access to an encoded dataset with values that occur frequently.

Table A.5 illustrates that the executed attack was able to re-identify fewer records than the others (in percentual), considering the experimental setup that we tested. However, it is important to mention that the re-identification results depend on the anonymization parameters, data, and thresholds employed in the BC-PPRL.

Brazilian politicians re-identification results

We re-executed the experiment considering the public Brazilian Politician Registration (BRP), available from: https://www.tse.jus.br. We employed all records from the 2014 and 2018 federal and estate elections, comprising 26,093 and 29,010 records, respectively.

Table A.6 shows different attributes combinations, and for each combination, we outline a set of unique values, V_A for the attacker and V_B for the victim dataset. The number of unique q-grams extracted from values in V_A is shown as $|Q_A|$, and from V_B as $|Q_B|$. The intersection of the common q-grams is denoted as $|Q_C|$.

Table A.6: Characteristics of the encoded attribute combinations and q-gram sets used in the attack

Encoded attribute	$\|V_A\|$	$\|V_B\|$	$\|V_C\|$	Jaccard overlap	$\|Q_A\|$	$\|Q_B\|$	$\|Q_U\|$	$\|Q_C\|$
FULL_NAME;BIRTH_CITY	26,093	29,010	3,117	6.0	959	1018	1111	866
FIRST_NAME;LAST_NAME	20,393	22,655	6,117	16.6	770	812	859	723

FIRST_NAME	5,992	6,487	3,017	31.9	604	625	675	554

We encoded the records considering four different BF lengths, 1000, 500, 200, and 100 bits. Table A.7 shows the number of hash functions for each attribute combination.

Table A.7: Characteristics of the q-gram sets for each attribute combinations

Encoded attribute	**bf len**	shared information	k	$\lvert Q_B^{seg}\rvert$	k_{min}^{seg}	k_{avr}^{seg}	k_{med}^{seg}	k_{max}^{seg}
	1000	1%	23	207	1	1.09	1	4
	1000	5%	23	720	1	1.64	1	6
	1000	10%	23	936	1	2.52	2	7
	1000	25%	23	1,018	1	5.82	6	14
	500	1%	11	104	1	1.05	1	2
	500	5%	11	434	1	1.25	1	4
	500	10%	11	683	1	1.55	1	5
FULL_NAME;BIRTH_CITY	**500**	25%	11	982	1	2.80	3	8
	200	1%	5	57	1	1.05	1	2
	200	5%	5	220	1	1.11	1	3
	200	10%	5	422	1	1.21	1	3
	200	25%	5	794	1	1.62	1	5
	100	1%	2	24	1	1.00	1	1
	100	5%	2	82	1	1.02	1	2
	100	10%	2	158	1	1.06	1	2
	100	25%	2	426	1	1.15	1	2
	1000	1%	56	332	1	1.29	1	4
	1000	5%	56	771	1	2.92	3	9
	1000	10%	56	811	1	5.51	5	13
	1000	25%	56	812	5	13.80	14	24
	500	1%	28	183	1	1.15	1	3
	500	5%	28	617	1	1.76	2	7
	500	10%	28	783	1	2.83	3	9
FIRST_NAME;LAST_NAME								

Continued on next page

Table A.7: Characteristics of the q-gram sets for each attribute combinations

Encoded attribute	bf len	shared information	k	$\lvert Q_B^{seg} \rvert$	k_{min}^{seg}	k_{avr}^{seg}	k_{med}^{seg}	k_{max}^{seg}
	500	25%	28	812	1	6.82	7	14
	200	1%	11	83	1	1.06	1	2
	200	5%	11	345	1	1.22	1	4
	200	10%	11	562	1	1.60	1	6
	200	25%	11	773	1	2.88	3	8
	100	1%	6	47	1	1.00	1	1
	100	5%	6	202	1	1.10	1	3
	100	10%	6	357	1	1.25	1	4
	100	25%	6	651	1	1.79	2	5
FIRST_NAME	**1000**	1%	119	421	1	1.64	1	5
	1000	5%	119	623	1	5.58	5	12
	1000	10%	119	625	3	11.21	11	22
	1000	25%	119	625	14	27.96	28	41
	500	1%	60	266	1	1.24	1	3
	500	5%	60	590	1	2.90	3	10
	500	10%	60	623	1	5.53	5	13
	500	25%	60	625	5	14.18	14	23
	200	1%	24	125	1	1.07	1	2
	200	5%	24	440	1	1.57	1	5
	200	10%	24	569	1	2.47	2	8
	200	25%	24	624	1	5.62	6	12
	100	1%	12	69	1	1.00	1	1
	100	5%	12	276	1	1.22	1	3
	100	10%	12	436	1	1.55	1	5
	100	25%	12	595	1	2.99	3	8

We employ the pprlSegmentAtomAttack over each scenario presented in Table A.7. We also expose the re-identification results considering the same output as Christen et al. (1-to-1-correct, 1-to-1-wrong, 1-to-many-correct, 1-to-many-wrong, and no matches). The re-identification results

are shown in Table A.8.

Notice that the re-identification rate is similar to the results shown in Section A.2.3. To assess the privacy limitation of BC-PPRL over a different dataset, we show the privacy metrics calculated over the victim's dataset in Table A.9.

Table A.9: Privacy metrics

Encoded attribute	bf len	shared information	k	PU(P,Q)	$KL(P\|\|Q)$	$JS(P\|\|Q)$
FULL_NAME;BIRTH_CITY	**1000**	25%	23	99%	0.000	0.003
	1000	10%	23	99%	0.000	0.004
	1000	5%	23	99%	0.000	0.013
	1000	1%	23	3%	0.744	0.487
	500	25%	11	99%	0.000	0.003
	500	10%	11	99%	0.001	0.014
	500	5%	11	91%	0.055	0.123
	500	1%	11	0%	0.668	0.474
	200	25%	5	99%	0.000	0.008
	200	10%	5	65%	0.297	0.282
	200	5%	5	3%	1.037	0.554
	200	1%	5	0%	0.120	0.209
	100	25%	2	90%	0.055	0.127
	100	10%	2	2%	0.991	0.573
	100	5%	2	0%	0.438	0.395
	100	1%	2	0%	0.056	0.143
FIRST_NAME;LAST_NAME	**1000**	25%	56	99%	0.000	0.009
	1000	10%	56	99%	0.001	0.014
	1000	5%	56	99%	0.003	0.029
	1000	1%	56	4%	0.460	0.399
	500	25%	28	99%	0.000	0.009
	500	10%	28	99%	0.003	0.029
	500	5%	28	90%	0.060	0.128

Continued on next page

Table A.9: Privacy metrics

Encoded attribute	bf len	shared information	k	PU(P,Q)	$KL(P\|\|Q)$	$JS(P\|\|Q)$
	500	1%	28	0%	0.189	0.258
	200	25%	11	99%	0.002	0.025
	200	10%	11	78%	0.127	0.193
	200	5%	11	4%	0.648	0.462
	200	1%	11	0%	0.108	0.199
	100	25%	6	91%	0.048	0.118
	100	10%	6	4%	0.586	0.445
	100	5%	6	0%	0.177	0.251
	100	1%	6	0%	0.005	0.043
FIRST_NAME	**1000**	25%	119	99%	0.001	0.015
	1000	10%	119	99%	0.001	0.016
	1000	5%	119	98%	0.008	0.049
	1000	1%	119	14%	0.414	0.379
	500	25%	60	99%	0.001	0.016
	500	10%	60	98%	0.005	0.040
	500	5%	60	94%	0.028	0.093
	500	1%	60	0%	0.134	0.222
	200	25%	24	98%	0.004	0.037
	200	10%	24	85%	0.082	0.156
	200	5%	24	14%	0.449	0.388
	200	1%	24	0%	0.012	0.068
	100	25%	12	93%	0.033	0.101
	100	10%	12	14%	0.382	0.363
	100	5%	12	0%	0.153	0.236
	100	1%	12	0%	0.000	0.000

In order to better illustrate the impact of Indistinguishability and Uncertainty on the re-identification rate of the attack, we plotted Table A.9 and A.8 result in Figure A.4. The elements of Figure A.4 are the same as the Figure A.7.

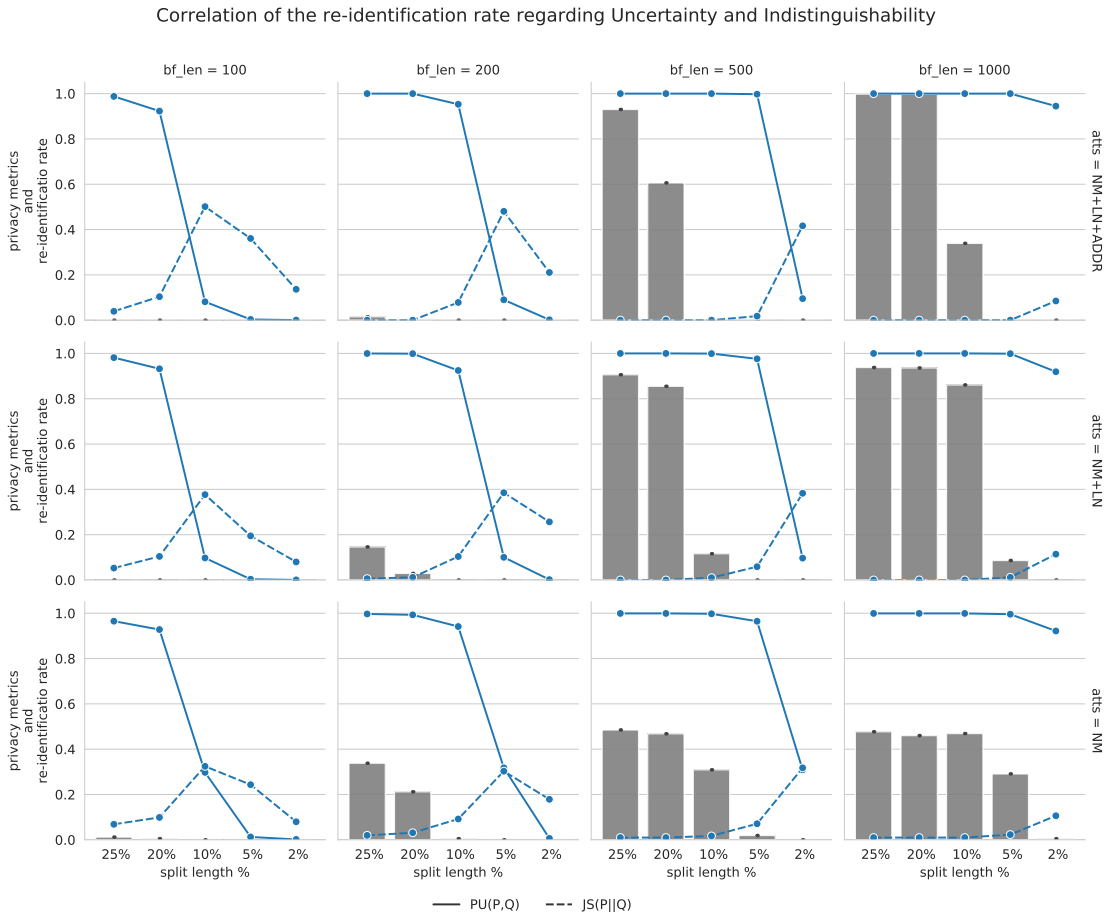


Figure A.7: Re-identification results and privacy metrics for different attribute combinations, BF length, and shared information.

Table A.8: Re-identification results.

attr	bf len	shared information	k	1-to-1 correct	1-to-many correct	1-to-1 wrong	1-to-many wrong	No matches
	1000	1%	23	2	0	18,536	10,472	0
	1000	5%	23	41	34	19,955	8,980	0
	1000	10%	23	4,828	2,452	14,772	6,958	0
	1000	25%	23	28,574	324	88	24	0
	500	1%	11	1	0	12,441	16,568	0
	500	5%	11	8	4	18,991	10,007	0
	500	10%	11	86	61	19,820	9043	0
FULL_NAME;BIRTH_CITY	**500**	25%	11	15,929	3793	6,731	2,557	0
	200	1%	5	1	1	16,042	12,966	0
	200	5%	5	1	2	22,848	6159	0
	200	10%	5	2	6	25,583	3,419	0
	200	25%	5	213	156	19,506	9,135	0
	100	1%	2	1	0	29,009	0	0
	100	5%	2	1	1	24,096	4,912	0
	100	10%	2	1	2	24,642	4,365	0
	100	25%	2	4	3	25,232	3,771	0
	1000	1%	56	6	13	11,835	10,801	0
	1000	5%	56	5,977	5,658	5,609	5,411	0
	1000	10%	56	20,214	1,965	307	169	0
	1000	25%	56	21,300	1,349	3	3	0
	500	1%	28	4	3	13,776	8,872	0
	500	5%	28	254	661	9,903	11,837	0
	500	10%	28	6,105	5,650	5,572	5,328	0
FIRST_NAME;LAST_NAME	**500**	25%	28	21,025	1,510	81	39	0
	200	1%	11	0	5	0	22,650	0
	200	5%	11	4	31	8,649	13,971	0
	200	10%	11	108	363	9,116	13,068	0
	200	25%	11	7,128	6,120	5,088	4,319	0
	100	1%	6	1	3	12,484	10,167	0
	100	5%	6	2	8	7,874	14,771	0
	100	10%	6	10	25	10,618	12,002	0
	100	25%	6	345	819	9,748	11,743	0
	1000	1%	119	14	45	2,656	3,772	0
	1000	5%	119	3,376	2,399	427	285	0
	1000	10%	119	4,695	1,781	10	1	0
	1000	25%	119	4,747	1,740	0	0	0
	500	1%	60	1	5	2,375	4,106	0
	500	5%	60	543	1,556	1,835	2,553	0
	500	10%	60	3,488	2,369	397	233	0
	500	25%	60	4,740	1,745	1	1	0
FIRST_NAME	**200**	1%	24	0	4	3,546	2,937	0
	200	5%	24	7	18	3,927	2,535	0
	200	10%	24	116	453	2,538	3,380	0
	200	25%	24	3,690	2,365	289	143	0
	100	1%	12	0	0	6,487	0	0
	100	5%	12	2	5	3,859	2,621	0
	100	10%	12	12	31	2,984	3,460	0
	100	25%	12	473	1,417	2,083	2,514	0

Notice that the attack was unable to re-identify the encoded records when Indistinguishability and Uncertainty are below the values mentioned in Section A.2.3. These results provide an additional indication that Indistinguishability and Uncertainty could be employed to estimate the privacy limitation of the BC-PPRL.

A.2.4 Conclusion

In the aforementioned sections, we acknowledge the BC-PPRL limitation that was reported by Christen et al. Furthermore, we presented elements (Indistinguishability and Uncertainty) that can be used to predict the cases which the BB-PPRL fails to preserve the privacy of the encoded records. In other words, Indistinguishability and Uncertainty evidence the amount of information that has a reduced chance to be re-identified in an attack.

It is important to mention that several parameters could impact BC-PPRL. For instance, the number of the hash functions (amount of information) represents an important element to the re-identification rate of an attack. Further investigation should be executed to evaluate the trade-off between linkage quality and privacy regarding the number of hash functions and filter length. It is worthwhile to mention that there are studies regarding the occupation of the BF; however, we are not aware of works that correlate the length and number of the hash functions to the privacy of the information.

Regarding the privacy issues of the PPRL, we believe that it is a hard task to fulfill. Mainly due to the high re-identification of the existing attacks, a total PPRL that can fully preserve all records' privacy is still an open problem. The BB-PPRL was not proposed to solve that issue. It was designed to demonstrate that novel techniques can be employed to improve the PPRL process.

Moreover, considering the i) complex parametrization of BC-PPRL, ii) computation cost of BC-PPRL, and iii) the fact that attacks could be improved and combined, the BB-PPRL should be carefully used. Furthermore, to ensure high privacy capabilities, the BB-PPRL should be combined with other techniques. The PPRL parties could use the BB-PPRL as a preliminary stage of the comparison step to select records and reduce the information that will be sent to a third party. For instance, considering the experimental scenario shown in Section A.2.3, the parties could re-encode the 1,346 records and send 75% of the original BF to a third party. This reduces the number and the amount of shared information of the encoded records, hampering the chance of success in re-identification attacks.

Finally, we would like to thank the time and effort of all seven researchers involved in the critique of the BB-PPRL. We believe that the discussion brought by Christen et al. was highly relevant to the PPRL and might help other researchers as much as we.

A.3 SBF Splits Differences

A BF stores m elements with k hash functions into l-bits, such that $e^{\tau} = [b_0, \cdots, b_l]$. SBF divides the original BF into s splits with $\frac{l}{s}$ bits, such that $SBF(e^{\tau}, s) = [\phi^0, \cdots, \phi^{s-1}]$, and $\phi^i = \left[b_i, \cdots, b_{i+(\frac{l}{s}-1)}\right]$. In this context, consider that $\Pr(u) = p$ represents that a bit position stores '1'[1], and the probability of a bit position storing the value '0' can be estimated by $\Pr(z) = 1 - \Pr(u)$; thus, $\Pr(z) = 1 - p$.

The probability of a split (ϕ^i) storing a specific distribution of bits ($\Pr(\phi^i = x)$) can be estimated by the number of '0' and '1' bits stored in the split [17]. Assuming that the BF length (l) is big enough, and the BF parameters are correctly configured, it is possible to consider that the BF bits' values are independent events [89]. Therefore, we can use a Binomial distribution, with $\frac{l}{s}$ positions (or trials) to estimate the $\Pr(\phi^i = x)$, i.e., $\Pr(\phi^i = x) \sim B\left(\frac{l}{s}, p\right)$, as presented in Eq. (A.3)

$$B\left(\frac{l}{s}, p\right) = \binom{\frac{l}{s}}{x} p^x (1-p)^{\frac{l}{s}-x} \tag{A.3}$$

Assuming $\Pr(\phi^i = x)$ as a Binomial distribution, we are able to estimate the variation of the bits within the splits using the binomial distribution mean (μ), variance (σ^2) and standard deviation (σ). Consider that $\mathcal{X}$ is a binomially distributed random variable, such as $\mathcal{X} \sim B(n, p)$, we can estimate the mean number of '1' in each split using Eq. (A.4).

$$\mu = \frac{l}{s} \cdot p \tag{A.4}$$

Knowing that the variance of a binomial distribution can be calculated by $\sigma^2(\mathcal{X}) = n \cdot p \cdot (1-p)$, and the standard deviation is the square root of the variance, we can use σ^2 and σ to estimate the variation of the bits within the splits applying Eq. (A.6) and Eq. (A.5).

$$\sigma^2 = \frac{l}{s} \cdot p \cdot (1-p) \tag{A.5}$$

$$\sigma = \sqrt{\frac{l}{s} \cdot p \cdot (1-p)} \tag{A.6}$$

Moreover, we employ the coefficient of variation (CV), defined as standard deviation (σ) divided by the mean (μ) to describe the variability of '1' within a split relative to the mean number of '1' in other splits. It is worthwhile to mention that the CV represents the variability in percentage. Eq. (A.7) calculates the CV of the SBF splits.

[1]The $\Pr(u)$ and $\Pr(z)$ of a BF can be estimated using BF setup parameters, as presented in Section 2.3.3, such that $\Pr(z) = e^{-\frac{k \times n}{l}}$ and $\Pr(u) = 1 - e^{-\frac{k \times n}{l}}$.

$$CV = \frac{\mu}{\sigma}$$

$$\Rightarrow \frac{\frac{l}{s} \cdot p}{\sqrt{\frac{l}{s} \cdot p \cdot (1-p)}} \Rightarrow \frac{\frac{l}{s} \cdot p}{\sqrt{\frac{l}{s}} \cdot \sqrt{p \cdot (1-p)}}$$

, considering that $\frac{l}{s} = \sqrt{\frac{l}{s}} \cdot \sqrt{\frac{l}{s}}$

$$\Rightarrow \frac{\cancel{\sqrt{\frac{l}{s}}} \cdot \sqrt{\frac{l}{s}} \cdot p}{\cancel{\sqrt{\frac{l}{s}}} \cdot \sqrt{p \cdot (1-p)}}$$

, considering that $p = \sqrt{p} \cdot \sqrt{p}$, and $\sqrt{a \cdot b} \Rightarrow \sqrt{a} \cdot \sqrt{b}$,

such that $a \geq 0$ and $b \geq 0$.

$$\Rightarrow \frac{\sqrt{\frac{l}{s}} \cdot \cancel{\sqrt{p}} \cdot \sqrt{p}}{\cancel{\sqrt{p}} \cdot \sqrt{(1-p)}} \Rightarrow \frac{\sqrt{\frac{l}{s} \cdot p}}{\sqrt{(1-p)}} \Rightarrow \frac{\sqrt{l} \cdot \sqrt{p}}{\sqrt{s} \cdot \sqrt{(1-p)}}$$

$$\Rightarrow \frac{\sqrt{l \cdot p}}{\sqrt{s \cdot (1-p)}}$$

Eq. (A.7) represents the percentual variation of '1' in terms of l, s, and p. Thus, considering that in a PPRL context $p \approx 0.5$ and l is bigger than s, it is intuitive that Eq. (A.7) will produce $CV > 0$. For instance, if we consider l=1,024, s=10 and p=0.5, the expect variability of '1' is approximated 31%. In other words, it is possible to configure a SBF to present, with a high probability, a slightly variation in the number of '1'. Therefore, assuming that the BF and SBF parameters are properly configured, the variation in the number of '1' is reflected in the Jaccard similarity metric, such that $Jaccard(\phi_a^i, \phi_b^i) \neq Jaccard(\phi_a^{i+1}, \phi_b^{i+1})$, where $\phi_a^i \in SBF(e_a^\tau, s)$ and $\phi_b^i \in SBF(e_b^\tau, s)$.

Furthermore, the Jaccard similarity of two BFs can be expressed as the mean similarity of their splits, as expressed in Eq. (4.3), $Jaccard\_SBF(e_a^\tau, e_b^\tau, s) = \frac{1}{s} \sum_{i=0}^{s} \frac{|\phi_a^i \bigcap \phi_b^i|}{|\phi_a^i \bigcup \phi_b^i|}$. Thus, considering the typical parameters of BF employed in a PPRL context, it is unlikely that every split of a BF is identical. This statement is straightforward to demonstrate by using the BF and SBF parameters in Eq. (A.3), and can be estimated by $\sum_{i=1}^{s} \Pr(\phi^i = x)$. For instance, considering the previous example (l=1,024, p=0.5, and s=10), the probability of every split being identical is $\Pr(\phi^i = x) \approx 2.52 \times 10^{-133}$.

Finally, consider that the SBF can be configured to vary the number '1' in its splits, and the remote probability of every split presents the same bit distribution. We assume that there is a difference between Jaccard similarity of the original BF and the similarity of one SBF split $Jaccard(e_a^\tau, e_b^\tau) \neq Jaccard(\phi_a^i, \phi_b^i)$, such that $\phi_a^i \in SBF(e_a^\tau, s)$ and $\phi_b^i \in SBF(e_b^\tau, s)$. Moreover, we associate this difference to an error ϵ, and use Eq. (4.4) to represent this intuition.

$$Jaccard(e_a^\tau, e_b^\tau) = Jaccard(\phi_a^i, \phi_b^i) + \epsilon$$

, such that $0 \leq \epsilon \leq 1$.

In this appendix, we intend to acknowledge the existence of differences between the similarity of one split and the BF that originated it. Appendix A.4 demonstrates the relation between our experimental results (Section 4.3) and the content of this Appendix

A.4 Experimental Analysis of the Relation between ϵ-error and σ

In order to illustrate the alignment of the arguments presented in Appendix A.3 with the results described in Section 4.3, we selected five datasets (bike, beer, NCVR, MVR and yv-er) that were encoded with the same filter length ($l = 1,024 - bits$) to show that the ϵ error is proportional to the error presented in Eq. (A.6).

Therefore, knowing the BF setup for these datasets and considering $p \approx 0.53$, we calculated the standard deviation (SD) for every used split length in the experiments and compared the results against the mean measured error (ϵ). The results are shown in Figure A.8; notice that we convert the error as a percentage of the bit length.

Notice that the calculated SD (x marks) is proportional to and near the line that represents the measured mean error of the experiments. This result corroborates with the experimental results presented in Sections 4.3 and 4.3.2, illustrated in Figures 4.3, 4.4, and 4.5, showing that the smaller the length of the SBF splits, the bigger the error originated from Eq. (4.4).

We would also like to mention that, until this point, we can acknowledge the existence of ϵ, and this error is related to the error measured in our experiments. In future work, we intend to investigate the particular characteristics of ϵ.

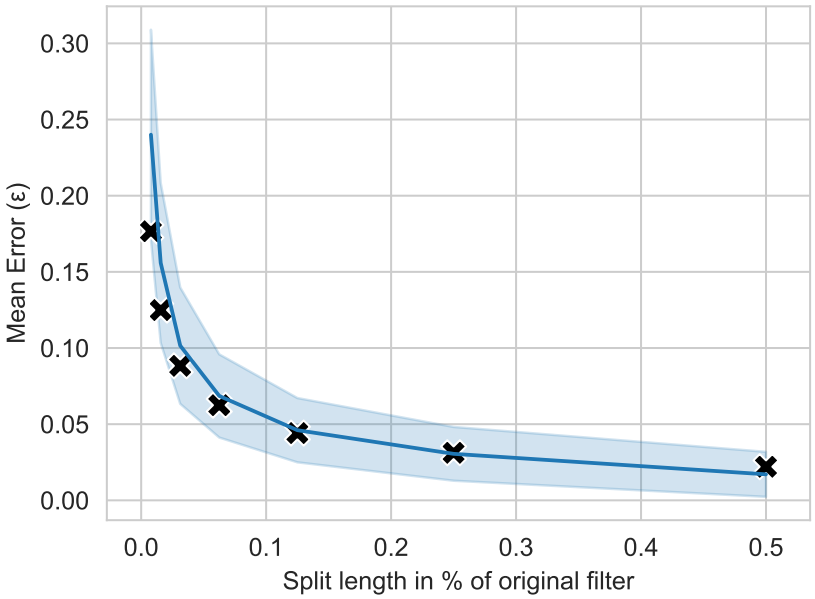


Figure A.8: Comparison of the measured and calculated errors associated with split length, considering a BBF.

A.5 Exploratory Data Analysis

Exploratory data analysis (EDA) investigates data sources and summarizes their main characteristics to define how to manipulate data sources to test hypotheses or check assumptions. Furthermore, EDA is used to understand data source variables and their relationships by summarizing their main characteristics, often employing data visualization methods.

In this research context, EDA represents an important tool to test our research hypotheses, especially when assessing and proposing classifiers to the PPRL process. Thus, to provide an overview of the 15 pairs of data sources employed in the evaluations presented in this document, we offer a brief EDA of the data source quality and how similar the duplicated entities (matching entities) are in each data source pair.

The EDA's main goal is to provide a better explanation of the insights and evaluations presented in this document. A summary of the data sources characteristics is shown in Table A.10.

context	DATASET DETAILS						
	name	ds-type	input-type	entities pair	matching pairs	attribute number	missing values
personal	census	synthetic	dirty	685,584	345	6	291
	mvr	real	clean-clean	2,250,000	150	9	1,772
	nvr	real	clean-clean	2,890,000	170	8	352
	tse	real	clean-clean	4,000,000	154	3	0
	yv-er	real	dirty	90,231,001	5,109	6	5,551
books	amazon-barnesnobel1	real	clean-clean	126,360	232	2	3
	goodreads-barnesnobel	real	clean-clean	90,662	89	4	925
	amazon-barnesnobel2	real	clean-clean	83,448	64	3	331
movies	imdb-roten	real	clean-clean	310,248	190	3	0
	imdb-tmd	real	clean-clean	127,086	356	2	0
restaurants	fodors-zagats	real	clean-clean	176,423	112	4	0
	yelp-yellowpages	real	clean-clean	76,874	116	6	0
	yelp-zomato	real	clean-clean	98,496	119	3	0
citacion	dplp-acm	real	clean-clean	6,001,104	2,224	4	14
products	abt-buy	real	dirty	1,157,776	1,076	1	0

Table A.10: Data source summary

Each dataset in Table A.10 presents different characteristics, such as the dataset type, input type, number of entity pairs, duplicated entities, missing values, the number and content of attributes, and the level of dirtiness of attributes (missing values). The input type indicates if all parties executed

a de-duplication process during the pre-processing step. This is relevant to the classification step because if a dirty input is employed to a PPRL, we could have '1:m' matching results; in other words, we could have one entity marked as similar to m other entities. Thus, making it harder to classifier identifies the matching entities.

In the following, each sub-section provides information about one specific data source. In each sub-section, we provide a brief description of the data source and report three characteristics of the data source:

- the missing attributes of each attribute;
- the number of unique values per attribute;
- the similarity of the duplicated entities.

It is worth mentioning that missing attribute values are an open research problem to the PPRL and RL and poses as a significant data source quality metric. In other words, data sources with missing values are considered hard to link [126] when compared to data sources with no missing values. Thus, to represent the missing attributes, we plot a figure where each bar describes one attribute and illustrates the missing values; we remove a piece of the bar.

The number of unique values per attribute may indicate a good blocking key or how each attribute is susceptible to privacy attacks. In other words, attributes with few unique values are easier to break the privacy; for example, the gender attribute usually has to value (male and female), making it more straightforward to reidentify than an attribute with more unique value like a phone number. Thus, to represent the attributes' unique values, we plotted a bar graph where each bar represents the number of unique values. We also plot two vertical axes, one representing the counting of the values and the other representing the percentual of unique values in the data source.

The duplicated entities' similarity intends to demonstrate how hard it is to select a classification threshold in the PPRL. This characteristic's primary goal is to illustrate that each data source presents different sub-optimal thresholds depending on entity values and the employed similarity metric employed. To illustrate this characteristic, we compare the data source's duplicated entities using different similarity metrics and plot de distribution (almost a histogram) of the similarity value in a figure.

In the following, we presented the characteristics mentioned above for the 15 data sources employed in this document.

A.5.1 Personal Data Sources

YV-ER

This data source comprised of all records having Italy as the victim's place of residence. The Dataset contains 9,499 records and 12,749 tagged record pairs. Thi data source is made public by the Yad Vashem foundation and Hewlett Packard Labs to be used for Entity Resolution research.

Name:	yv-er
Type:	real
Description:	Italian holocaust victim's
Original Application:	deduplication
url	https://github.com/tomersagi/yv-er
Collected:	21/10/2020

Considering that this data source was originally employed for de-duplication, in other words, all parties have the exact same data source, in the following Figures A.9, we presented the unique values and missing attributes.

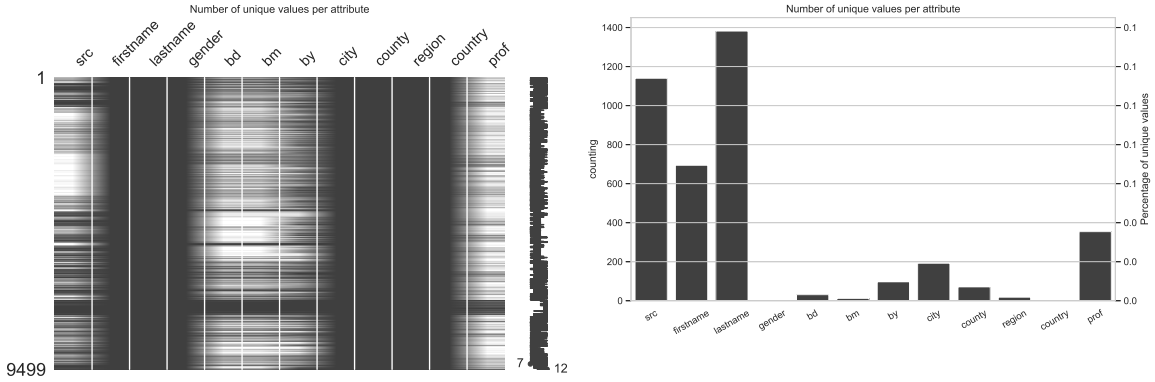


Figure A.9: Yv-er data source unique and missing values per attributes

Some of the attributes shown in Figure A.9 are not intuitive. Thus, Table A.11 provides a brief dictionary of data for a better understanding.

attribute	description
src	Name of submission and submission year for pages of testimony
firstname	Disambiguated first name
lastname	Disambiguated surname
gender	Male, Female
bd	birth day
bm	birth month
by	birth year
city	Disambiguated city of Permanent residence
county	Disambiguated county of War-time residence
region	Disambiguated region of death or camp type
prof	profession

Table A.11: Data dictionary

The similarity of the duplicated entities is plotted in Figure A.10

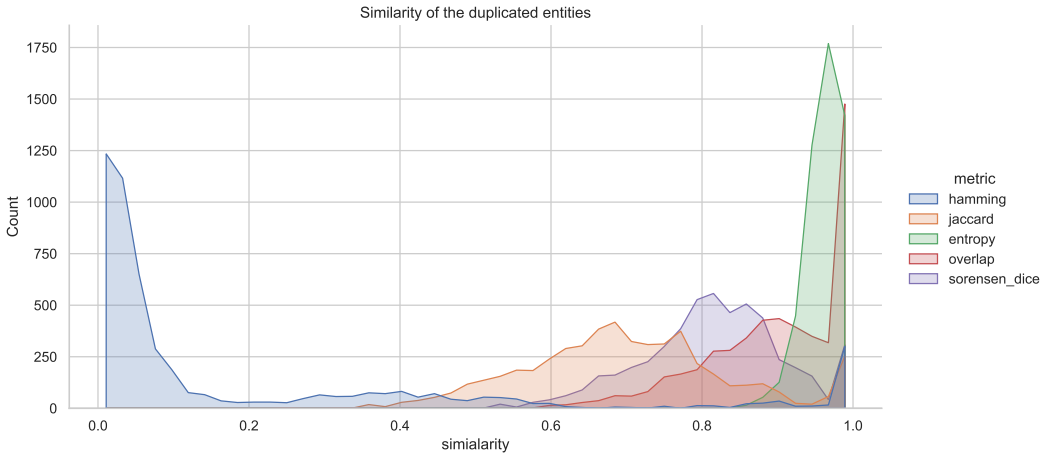


Figure A.10: Similarity of the duplicated entities.

A summary with further information about the attributes is shown in the following Table.

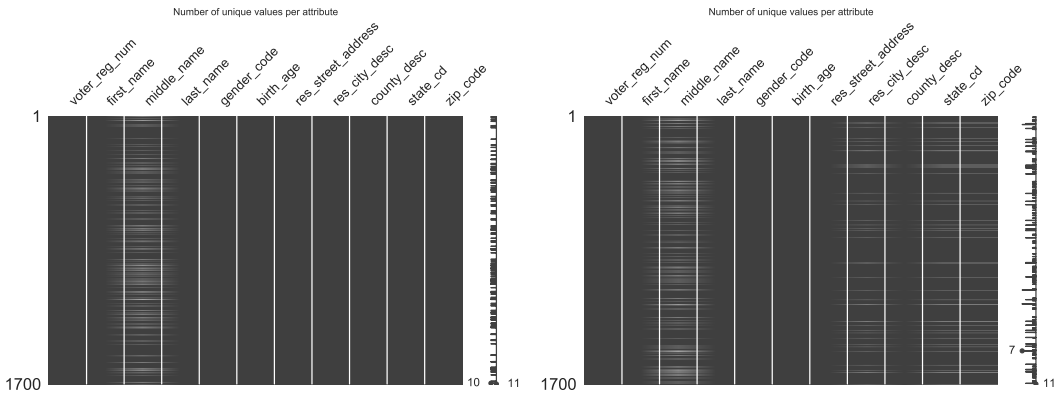
	metrica	id	src	firstname	lastname	gender	bd	bm	by	city	county	region	country	prof
0	count	9,499	9,499	9,499	9,499	9,499	3948	3979	6316	9,499	9,499	9,499	9,499	9,499
1	unique	9,499	1,141	694	1,382	2	31	12	97	192	71	18	2	355
2	top	7,080,822	NaN	Rakhel	Levi	Male	5	8	1900	Roma	Roma	Dodecanese Islands	Italy	NaN
3	freq	1	4,412	195	401	5,220	168	376	143	2,620	3,850	3,150	9,498	7,040
4	NaN	0	0	0	0	0	5,551	5,520	3,183	0	0	0	0	0

North Carolina Voter Registration (NCVR)

The NCVR comprises information about voter registration from North Carolina - USA. The methodology presented in [22,34,56] were employed to build the NCVR using data collected in October 2016 and June 2020.

Name:	ncvr
Type:	real
Description:	voter registration
Original Application:	linkage
url	https://www.ncsbe.gov/
Collected:	18/10/2016 and 31/05/2020

Considering that this data source was initially employed for linkage, we show the missing and unique value of the attributes separately. In Figure A.11 we expose the missing values in 2016 and 2020 NCVR data source, respectively.

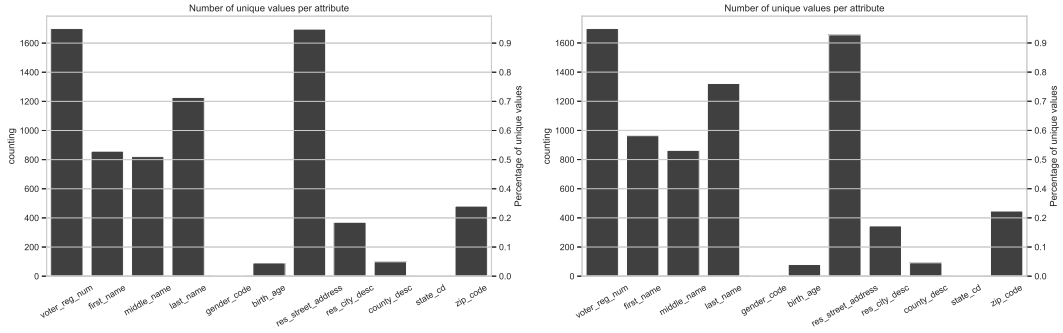


(a) 2016 NCVR data source (b) 2020 NCVR data source

Figure A.11: NCVR missisng attributes

In Figure **??** we presented the unique values of attributes form the 2016 and 2020 NCVR data source, respectively.

The similarity of the duplicated entities is plotted in Figure A.13.



(a) 2016 NCVR data source (b) 2020 NCVR data source

Figure A.12: NCVR missisng attributes

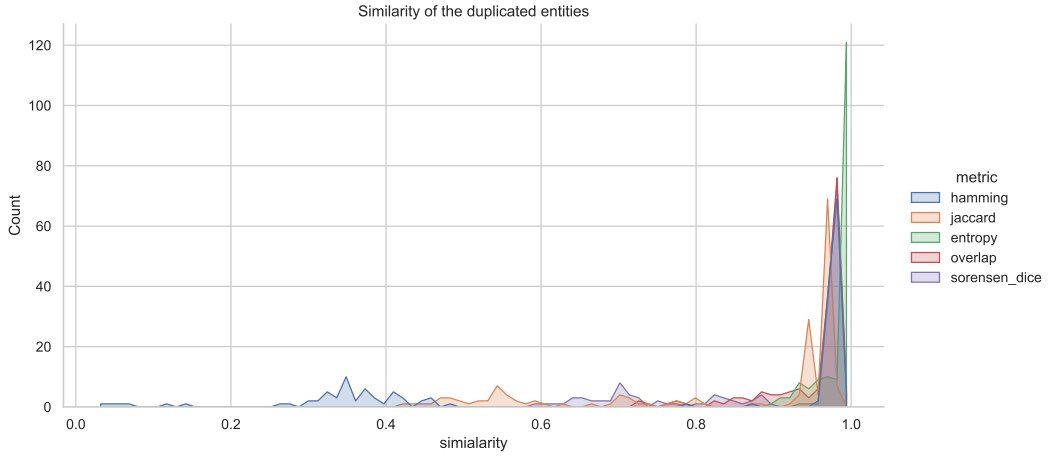


Figure A.13: Similarity of the duplicated entities.

A summary with further information about the attributes is shown in the following Table.

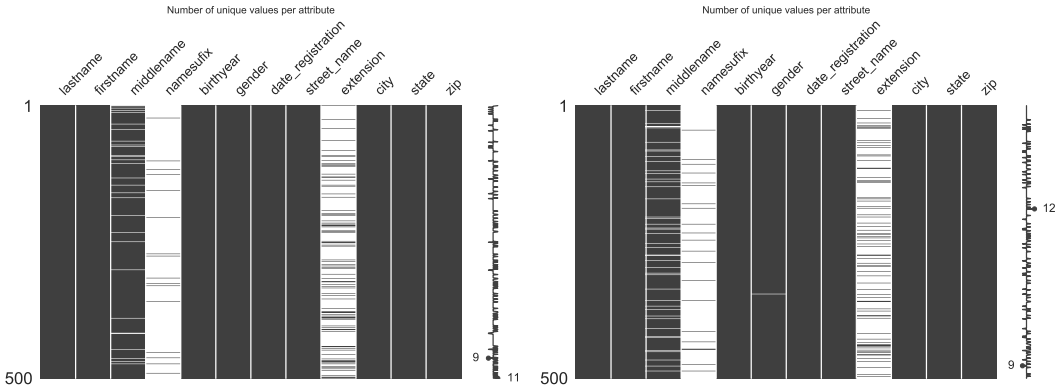
	metrica	ncid	first_name	middle_name	last_name	gender_code	res_street_address	res_city_desc	county_desc	state_cd
0	count	1,700	1,700	1,551	1,700	1,700	1,700	1,700	1,700	1,700
1	unique	1,700	858	821	1,227	3	1,696	370	99	1
2	top	CK13433	JAMES	LEE	WILLIAMS	F	1 DUKE UNIVERSITY WEST CAMPUS	CHARLOTTE	MECKLENBURG	NC
3	freq	1	24	41	17	922	2	133	149	1,700
4	NaN	0	0	149	0	0	0	0	0	0
0	count	1,700	1,700	1,546	1,700	1,700	1,700	1,658	1,700	1,658
1	unique	1,700	964	863	1,322	3	1656	345	92	1
2	top	AN211944	CHRISTOPHER	MARIE	MOORE	F	REMOVED	CHARLOTTE	MECKLENBURG	NC
3	freq	1	18	38	14	801	42	168	196	1,658
4	NaN	0	0	154	0	0	0	0	42	0

Michigan Voter Registration (MVR)

The MVR comprises information about voter registration from Michigan - USA. The methodology presented in [22,34,56] were employed to build the MVR using data collected in September 2014 and March 2017.

Name:	mvr
Type:	real
Description:	voter registration
Original Application:	linkage
url	https://mvic.sos.state.mi.us/
Collected:	05/09/2014 and 29/03/2017

Considering that this data source was initially employed for linkage, we show the missing and unique value of the attributes separately. In Figure A.14 we expose the missing values in 2016 and 2020 MVR data source, respectively.

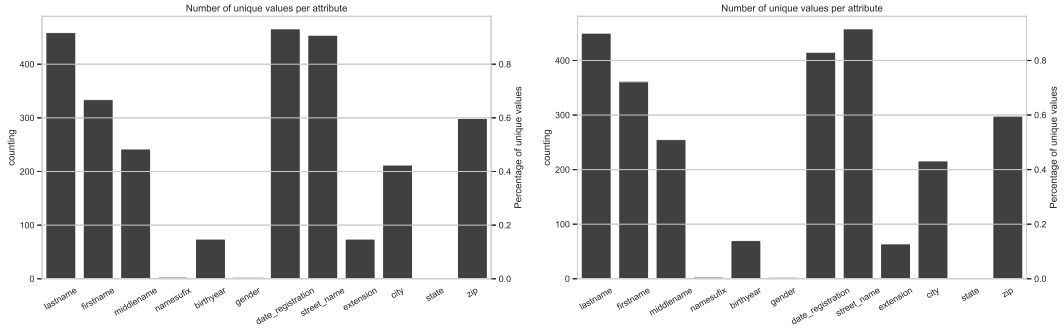


(a) 2014 MVR data source (b) 2017 MVR data source

Figure A.14: MVR missisng attributes

In Figure A.15 we presented the unique values of attributes form the 2014 and 2017 MVR data source, respectively.

The similarity of the duplicated entities is plotted in Figure A.16.



(a) 2014 MVR data source (b) 2017 MVR data source

Figure A.15: MVR missisng attributes

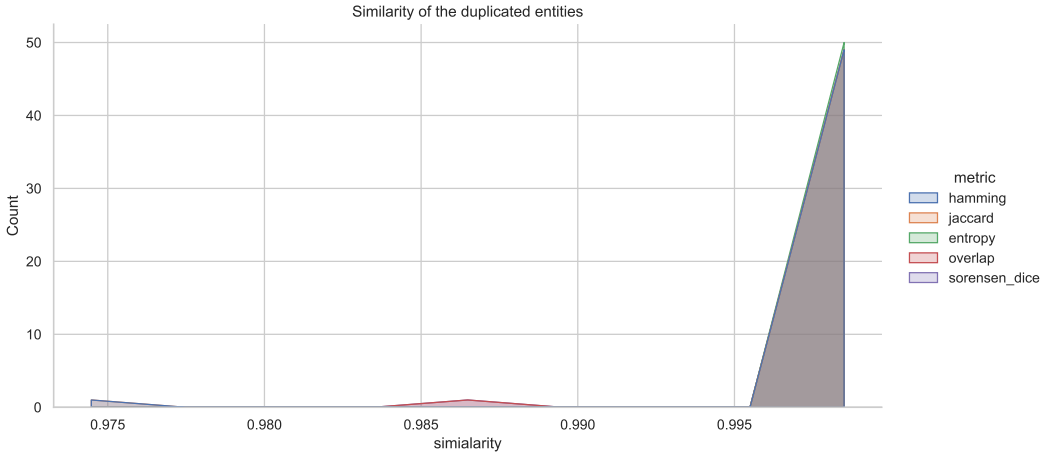


Figure A.16: Similarity of the duplicated entities.

A summary with further information about the attributes is shown in the following Table.

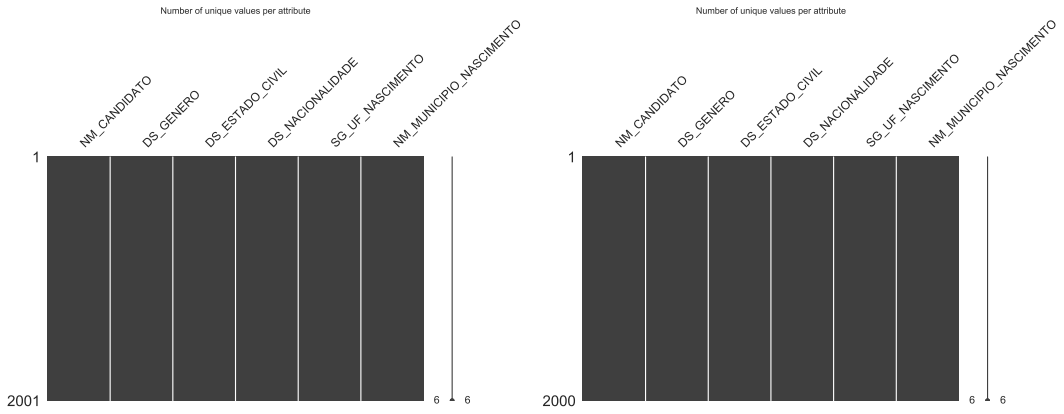
	metrica	lastname	firstname	middlename	namesufix	gender	street_name	extension	city	state
0	count	500	500	471	16	500	500	89	500	500
1	unique	459	334	242	3	2	454	74	212	1
2	top	SMITH	DAVID	MARIE	JR	F	OAK	APT 2	DETROIT	MI
3	freq	5	10	32	10	287	4	7	47	500
4	NaN	0	0	0	29	484	0	0	0	0
0	count	500	500	456	21	499	500	74	500	500
1	unique	450	361	255	3	2	458	64	216	1
2	top	WILSON	JAMES	MARIE	JR	F	FOREST	APT 1	DETROIT	MI
3	freq	6	7	17	14	267	3	5	33	500
4	NaN	0	0	0	44	479	0	1	0	0

Tribunal Superior Eleitoral (TSE)

The TSE comprises information about Brazilian politicians. The methodology presented in [22,34,56] were employed to build the TSE using data collected in September 2014 and March 2018.

Name:	tse
Type:	real
Description:	politicians registration
Original Application:	linkage
url	https://www.tse.jus.br/
Collected:	15/05/2019

Considering that this data source was initially employed for linkage, we show the missing and unique value of the attributes separately. In Figure A.17 we expose the missing values in 2014 and 2018 tse data source, respectively.

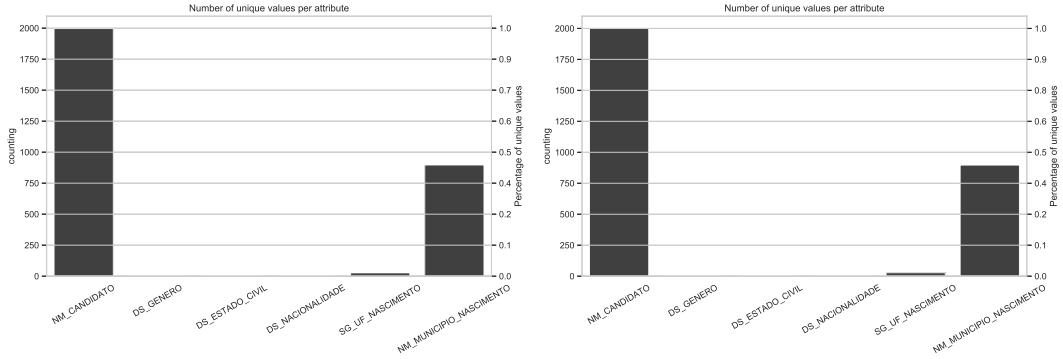


(a) 2016 TSE data source (b) 2018 TSE data source

Figure A.17: TSE missisng attributes

In Figure A.18 we presented the unique values of attributes form the 2014 and 2017 TSE data source, respectively.

The similarity of the duplicated entities is plotted in Figure A.19.



(a) 2014 TSE data source (b) 2017 TSE data source

Figure A.18: TSE missisng attributes

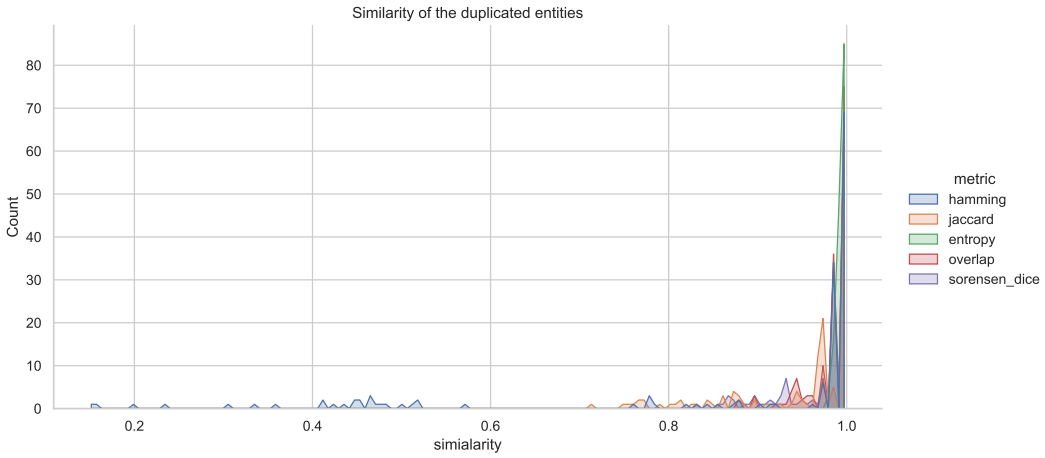


Figure A.19: Similarity of the duplicated entities.

A summary with further information about the attributes is shown in the following Table.

	metrica	NR_CPF_CANDIDATO	NM_CANDIDATO	DS_GENERO	DS_ESTADO_CIVIL	DS_COR_RACA	DS_OCUPACAO	DS_NACIONALIDADE	SG_UF	DT_NASCIMENTO	SG_UF_NASCIMENTO	NM_MUNICIPIO_NASCIMENTO	DS_GRAU_INSTRUCAO	NM_EMAIL
0	count	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001	2,001
1	unique	1999	1997	2	5	5	138	3	28	1,876	27	897	7	1,396
2	top	15705099487	FRANCISCO JOSÉ DA SILVA	MASCULINO	CASADO(A)	BRANCA	OUTROS	BRASILEIRA NATA	SP	11/03/1965	RJ	RIO DE JANEIRO	SUPERIOR COMPLETO	#NULO#
3	freq	2	2	1,337	1,052	1,096	371	1992	275	3	258	165	928	470
4	NaN	0	0	0	0	0	0	0	0	0	0	0	0	0
0	count	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000	2,000
1	unique	1999	1999	2	5	5	141	2	28	1872	28	897	7	1455
2	top	47555882604	ANTONIO AUGUSTO JUNHO ANASTASIA	MASCULINO	CASADO(A)	BRANCA	OUTROS	BRASILEIRA NATA	RJ	15/05/1967	RJ	RIO DE JANEIRO	SUPERIOR COMPLETO	#NULO#
3	freq	2	2	1,369	1,066	1,100	374	1994	294	3	281	168	899	396
4	NaN	0	0	0	0	0	0	0	0	0	0	0	0	0

Census

This data source is employed to simulate the de-duplication performed in a populational census.

Name:	census
Type:	synthetic
Description:	census data source
Original Application:	deduplication
url	https://hpi.de/naumann/projects/repeatability/datasets/
Collected:	09/06/2020

Considering that this data source was originally employed for de-duplication, in other words, all parties have the exact same data source, in the following Figure A.20, we presented the unique values and missing attributes.

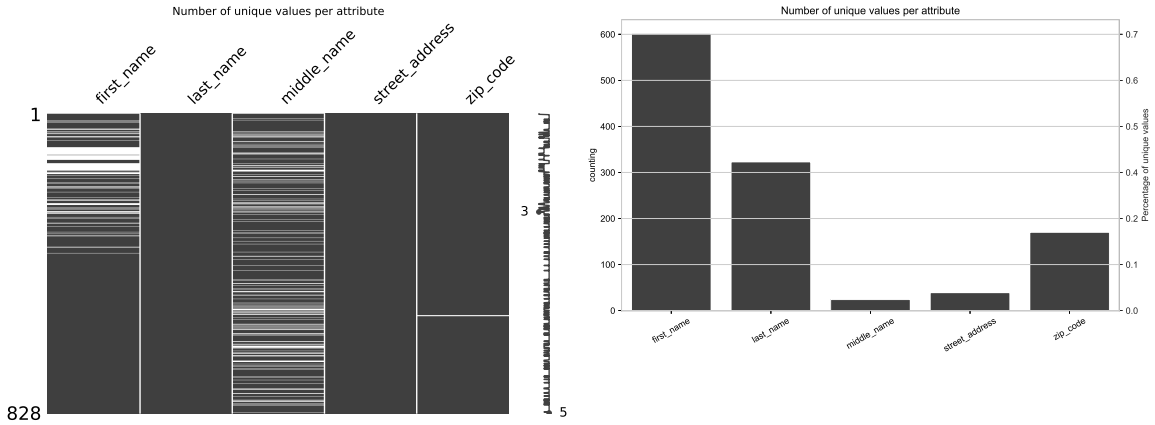


Figure A.20: Census data source unique and missing values per attributes

The similarity of the duplicated entities is plotted in Figure A.21

A summary with further information about the attributes is shown in the following Table.

	metrica	first_name	last_name	middle_name	street_address	zip_code
0	count	720	841	671	841	841
1	unique	606	323	24	39	171
2	top	JEFFREY	MOSQUERA	A	STARKEY	102
3	freq	9	7	92	98	26
4	NaN	121	0	170	0	3

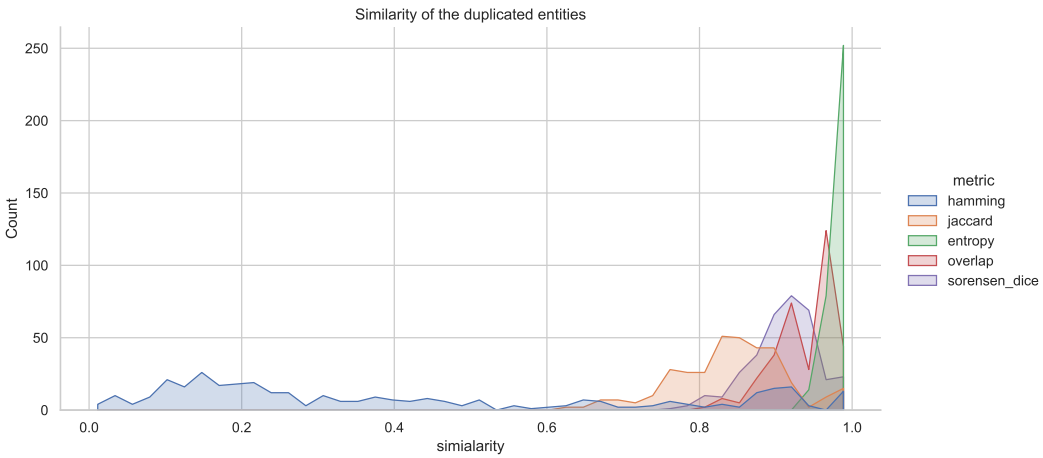


Figure A.21: Similarity of the duplicated entities.