Information Systems 140 (2026) 102726

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

The use of differential privacy for privacy-preserving record linkage: Protecting the bits but not the people

Sumayya Ziyad [a,*], Peter Christen [a], Rainer Schnell [b], Lucas Lange [c], Anushka Vidanage [a]

[a] Australian National University, Canberra, 2600, Australia
[b] University of Duisburg–Essen, Duisburg, 47057, Germany
[c] University of Leipzig, Leipzig, 04109, Germany

ARTICLE INFO

Dataset link: Code and appendix with additional experiments (Original data)

Keywords:
Private record linkage
Entity resolution
Data privacy
Bloom filter
Binary encoding
Bit flipping

ABSTRACT

Privacy-Preserving Record Linkage (PPRL) aims to identify records that refer to the same entity across databases held by different organisations without revealing sensitive information about the entities whose records are being linked. Research has shown that some popular PPRL techniques can be vulnerable to reidentification attacks. In response, the use of Differential Privacy (DP) has been investigated with the aim to provide formal privacy guarantees for PPRL. Multiple studies have explored the use of DP during the blocking stage, where similar records are grouped prior to comparison. Yet, since encodings of individual records must be shared for comparison and classification, the linkage process remains vulnerable to attacks despite being differentially private during the blocking stage, unless a computationally expensive secure multi-party protocol is used. Other studies have explored the use of DP during the encoding stage to guarantee that encoded records remain private even when exchanged between the parties involved in a PPRL protocol. While such approaches do employ established DP methods, we consider that their current application in the context of PPRL is nonsensical. The purpose of PPRL is to identify, with highest possible accuracy, specific records that refer to the same entity, while DP perturbs sensitive data to prevent possible reidentification of individuals within a data set. Therefore, this is a mismatch of paradigms. In its current use, DP for PPRL requires substantial perturbation to guarantee privacy, which in turn leads to a notable degradation of linkage quality. To support this argument, we survey and review the use of DP for PPRL, focusing on its effectiveness in protecting the real-world entities (generally people) whose records are being linked.

1. Introduction

Record linkage is the process of identifying records that refer to the same real-world entity, and is widely used to integrate data across different sources [1]. In many real-world applications, such as in the health domain, record linkage is performed on personal data [2]. In the absence of unique entity identifiers on which a direct database join can be conducted, linkage has to be based on personally identifiable information, such as names, addresses, and dates of birth [2]. Such attributes are referred to as quasi-identifiers (QIDs). The use of sensitive QIDs for linkage has led to a need for Privacy-Preserving Record Linkage (PPRL) techniques which ensure the privacy of the entities whose records are being linked [2,3].

Consider a research project investigating the impact of education on a population's employability. For such an analysis, a researcher does not need to know the actual identities of the individuals whose data is available, rather they only require that an individual's education history is accurately linked to their corresponding employment records. However, to create this integrated view, it is necessary to identify and link the records of all individuals that occur in both the employment and education databases. If we assume that the education and employment data are held by two different organisations (commonly known as the database owners), then such a linkage might not be possible on sensitive plaintext data due to privacy concerns and/or regulatory requirements [2]. Hence, PPRL must be employed on encoded versions of the sensitive QIDs of the entities for whom data is available. In practical PPRL applications, such databases may contain all citizens of a country, resulting in millions of records [4].

Over time, many different PPRL techniques have been developed and they can be broadly categorised into (1) secure multi-party computation-based techniques [5] and (2) perturbation-based techniques [2,6]. Secure multi-party computation-based techniques have been proven to provide strong and formal privacy guarantees, but they

* Corresponding author.
E-mail address: sumayya.ziyad@anu.edu.au (S. Ziyad).

https://doi.org/10.1016/j.is.2026.102726
Received 3 March 2026; Received in revised form 24 March 2026; Accepted 27 March 2026
Available online 28 March 2026

are typically resource intensive [7–9]. To link two data sets containing n and m records, respectively, Inan et al. [8] claim that most existing secure multi-party computation-based techniques require $O(n \cdot m)$ cryptographic operations. As such, even linking two relatively small data sets containing only 10,000 records each requires 10^8 operations. Lazrig et al. [7], in 2018, proposed a secure multi-party computation-based PPRL technique which required over eight hours to link two data sets containing 10,000 records each (with each database owner using a total of 21 nodes consisting of 2.4 GHz CPUs and 96 Gigabytes of memory). Due to the increasing volumes of data being handled in the real world, such resource limitations have, so far, prevented the use of secure multi-party computation-based techniques in practical PPRL applications, despite their privacy guarantees [2].

In contrast, existing perturbation-based PPRL techniques (that do not incorporate differential privacy) are known to provide a better trade-off between linkage quality, scalability, and privacy. Despite lacking formal privacy guarantees, these PPRL techniques are computationally efficient and scalable to large databases, thus preferred in real-world applications [2]. Bloom Filter (BF) based encoding [10], as we discuss in more detail in Section 3.2, is one such popular perturbation-based PPRL encoding technique where sensitive QID values are split into character q-grams and encoded into bit arrays using a set of independent hash functions [10].

However, recent studies have identified certain weaknesses of perturbation-based PPRL techniques, such as the frequencies and lengths of encodings [11]. Attacks have been designed to exploit such weaknesses with the aim to reidentify sensitive QID values, and the corresponding entities themselves [12]. Multiple studies have, therefore, attempted a combination of techniques to provide efficient linkage methods with provable privacy. One approach to bridging this gap has been through the use of differential privacy [8,13].

Differential Privacy (DP) is a formal privacy framework introduced to protect the individuals of a population whose data is being released through statistical summaries [14]. DP therefore aims to primarily prevent identity disclosure [15], as we discuss further in Section 2. DP is often achieved through the addition of random noise which masks the true underlying values. The amount of noise added is dependent on a privacy parameter, ϵ [14], which controls the trade-off between privacy and utility. Since the addition of random noise will result in a perturbed output [16,17], the applicability of DP to any task depends on the extent of utility one is willing to compromise.

While DP allows the release of *noisy* statistics about a database, PPRL expects the generation of the *accurate* final result of the linkage process (the complete set of record pairs classified as matches) without revealing any other sensitive information about the databases being linked [16]. This highlights a fundamental contradiction between the design goals of DP and PPRL.[1] Because although a privacy-utility trade-off is to be expected, the extent to which utility can be compromised is restricted in the context of PPRL. To emphasise, in PPRL, to prevent any true matches (two records that refer to the same real-world entity) to be missed (missed links) and no false matches (two records that refer to different entities) to occur (false links), it is required that all stages of the record linkage process remain as accurate as possible [1].

The consequences of a missed or false link will depend on how crucial it is to the decisions being made using the linked data set; if the presence (alternatively, the absence) of the said link can alter the decisions made [19]. For instance, consider a heavily imbalanced data set representing 95% males and only 5% females. If the noise introduced through the application of DP resulted in no classified matches for females (as well as a reduction in classified matches for males), then any decision made based on the outcome cannot be generalised to females. Importantly, the data consumer (the researcher analysing the linked data set) would not know if the lack of female matches were due to the absence of any true matches, due to linkage errors, or due to noise being added by DP. The implications of such situations can similarly be extended to domains such as health and criminology [19,20].

Linking databases is not just employed in research or data analytics contexts, but also in operational applications such as fraud prevention and national security [21,22]. A missed true match in such applications may not be a viable option.

Furthermore, given that the design goal of DP focuses on releasing statistical summaries, the extent to which it can be applied for PPRL is limited. As we discuss in Section 3, existing PPRL techniques that use DP can only do so in the stages of the record linkage process that require exchanging information about (1) the databases being linked (such as block sizes or token frequencies), or (2) the record pairs being compared. As such, differentially private PPRL (DP-PPRL) techniques vary specifically in the stage of the linkage process where DP is applied. Multiple techniques apply DP in the blocking stage [8,9,16], where similar records are grouped together based on a certain (set of) characteristic(s). Additionally, several PPRL techniques apply DP in the encoding stage to enforce formal privacy on the encoded records which are sent to the party that conducts linkage (generally known as the linkage unit). The most widely adopted technique in that stage employs bit flipping [23] to perturb potentially vulnerable patterns in BFs [13].

However, performing the final comparison and classification using DP is challenging, if not impossible, because it would require measuring the similarities between record pairs solely through statistical queries [8]. In any case, the resulting classification of record pairs is not differentially private unless the linkage outcome, the set of record pairs classified as matches, is perturbed. Such perturbation of the linkage outcome can result in missed true matches and in false matches, both of which can have severe consequences.

In this paper, to the best of our knowledge, we review all currently known techniques that use DP for PPRL. In particular, we argue that although DP mechanisms have formal privacy guarantees, their current application in the context of PPRL seems nonsensical. It must be noted that *we do not criticise DP*, we only discuss how the application of DP perturbs certain information in the linkage process, such as block sizes or bit patterns. However, the current use of DP in PPRL does not really protect the individuals in the databases being linked, as would be expected by the design goal of DP.

Hence, we consider the different stages of the PPRL process, and identify those stages that generate an output that can be exploited for reidentification when data is exchanged between the parties involved in a PPRL protocol [24]. We discuss the characteristics of PPRL techniques that enable a trade-off between linkage quality (utility), privacy, and scalability, and describe aspects of how to design PPRL techniques such that they address known privacy concerns while not affecting linkage quality.

The remainder of this paper is organised as follows. We next discuss relevant background on DP and PPRL. Then, in Section 3, we review existing DP-PPRL techniques categorised according to the stage of PPRL in which DP is applied. We then analyse the application and implementation aspects of these existing DP-PPRL techniques in Section 4. In Section 5, we experimentally evaluate the impact of DP in PPRL, with a particular focus on bit flipping and its influence on linkage quality and privacy. In Section 6, we discuss our insights into why we see the existing use of DP in PPRL to be nonsensical, and provide recommendations as to how existing PPRL techniques (not limited to DP-PPRL) should be employed in practical projects to achieve improved privacy without compromising on utility or scalability. Finally, in Section 7, we

[1] DP can be considered as a paradigm in the strict sense of Kuhn [18], who defined paradigms as "universally recognised scientific achievements that for a time provide model problems and solutions to a community of practitioners". The DP paradigm is concerned with statistical summaries, but PPRL is concerned specifically with uniquely identified records, not with summaries. Therefore, applying DP to PPRL can be seen as category error. That does not say anything about DP, just about the set of problems DP is designed for, where PPRL is not an element of this problem set.

conclude with a summary of our findings. In Appendix A, we provide key quotes from PPRL work that use DP to confirm our claims, while extended experimental results, as well as code and sample data sets, can be found on GitHub.[2]

2. Background

We now present the concepts relevant to our work, where we first define differential privacy (DP). Then, in Section 2.2, we define privacy-preserving record linkage (PPRL) and briefly describe the PPRL process. In Section 2.3 we outline the privacy objectives of PPRL, and how existing PPRL techniques aim to fulfil these objectives. Finally, in Section 2.4, we introduce Bloom filter encodings, the popular PPRL encoding technique for which differentially private bit flipping was introduced.

2.1. Differential privacy

The concept of DP was first introduced by Dwork [14] to assess the privacy risk incurred by individuals whose data is included in a database from which output is to be generated. For DP, it is assumed that the access to the database is limited to statistical queries. DP formalises the privacy guarantees as follows [14].

Definition 1 (*Differential Privacy*)**.** A randomised function $\mathcal{K}$ gives ϵ-differential privacy if for all data sets $\mathbf{D}_1$ and $\mathbf{D}_2$ differing in at most one element, and all query outcomes $S \subseteq \text{Range}(\mathcal{K})$,

$$\Pr[\mathcal{K}(\mathbf{D}_1) \in S] \leq e^{\epsilon} \cdot \Pr[\mathcal{K}(\mathbf{D}_2) \in S] \quad (1)$$

Accordingly, a randomised function $\mathcal{K}$ is ϵ-differentially private if any subset of the outcomes of $\mathcal{K}$ does not vary by more than a factor of ϵ. This implies that the presence or absence of a single record in a database should not substantially affect the outcome of any statistical query operation. If any query outcome was distinguishable as such, it is expected that an adversary can combine their knowledge of the queries and query outcomes to reidentify individuals [14,25]. The aim of DP, therefore, is to prevent identity disclosure. However, when applied in the context of PPRL, as we discuss in Section 2.2, DP primarily results in the prevention of group and attribute disclosure [15]. Note that if D_1 and D_2 differ in a single record then they are known as *adjacent databases* [26].

This privacy risk is controlled by ϵ, the privacy parameter that calculates the loss in privacy for an individual if their record was stored in a database. Based on Definition 1, a larger ϵ value implies a greater loss in privacy since it allows a greater variation in query results, and vice versa. The amount of noise to be added to perturb the underlying data is hence determined by ϵ, where such noise is often drawn randomly from a Laplace distribution [14]. While ϵ is recommended to be within the range of 0.001 and 1 [25], in practical applications ϵ generally varies up to a value of 16. An average value of 4.39 has been reported based on a comprehensive collection of real-world applications[3] of DP [27].

The application of DP can broadly be categorised into two [28]: (1) global (or central) DP and (2) local DP. Global DP is achieved by adding noise to aggregated data collected for multiple (or all) records of a database, such as aggregate summaries, and is often performed by a third-party curator. Local DP focuses on individual records in a database, such as perturbing the encoding of an individual record before it being shared [29].

[2] https://github.com/SumayyaZiyad/dp-for-pprl

[3] https://desfontain.es/blog/real-world-differential-privacy.html

2.2. Privacy-preserving record linkage

Record linkage aims to identify records that refer to the same real-world entity either within a single database, known as duplicate detection [30], or across two or more databases [1,31]. The latter is used to integrate data from different sources, and has seen many practical applications since the early 1940s [32,33]. While linking databases across different sources presents various challenges [34], PPRL focuses on addressing the privacy concerns associated with handling sensitive (often personal) data used for linkage. Hence, without loss of generality, we define PPRL for two databases as follows [2].

Definition 2 (*PPRL*)**.** Assume $\mathbf{D}_A$ and $\mathbf{D}_B$ are two databases, with their encoded versions denoted with $\mathbf{E}_A$ and $\mathbf{E}_B$. The PPRL process aims to determine which of the encodings $e^i_A \in \mathbf{E}_A, e^j_B \in \mathbf{E}_B$ refer to the same real-world entity, as determined by a decision model $C(e^i_A, e^j_B)$. In a perfect classification scenario, C would correctly classify all true matching pairs (e^i_A, e^j_B) into the set $\mathcal{M}$ (where their corresponding records, r^i_A and r^j_B, refer to the same real-world entity, $r^i_A \equiv r^j_B$), and all true non-matching pairs into the set $\mathcal{N}$ (where their corresponding records, r^i_A and r^j_B, refer to different entities, $r^i_A \not\equiv r^j_B$). At the end of the PPRL process the database owners only learn which records they have in common in $\mathcal{M}$, according to C, without any actual values of any records in $\mathbf{D}_A$ and $\mathbf{D}_B$ being revealed to any party within or outside the linkage protocol.

The primary objective of linkage (whether traditional record linkage or PPRL) is to accurately identify the sets $\mathcal{M}$ and $\mathcal{N}$. As discussed in Section 1, the consequences of misclassifying true matches (missed links) and true non-matches (false links) can be severe depending on the linkage application. Therefore, irrespective of the linkage technique and any privacy mechanisms employed, achieving high linkage quality must remain the highest priority.

The PPRL process is similar to that of traditional record linkage [2]. However, to ensure the privacy of the sensitive data handled during the different stages, linkage is either performed in a secure setting or any sensitive information exchanged between the parties is encoded to protect the actual sensitive values.

The PPRL process generally consists of six main stages [2], as outlined below and illustrated in Fig. 1.

1. *Parameter agreement*: The database owners first need to agree upon the required parameters. These include, but are not limited to, which QID attributes to use for blocking, which QID attributes to use for comparing records, and any other parameters specific to the data preprocessing and comparison techniques employed [1]. Additionally, the database owners have to agree on the secret keys and any parameters specific to the encoding and/or private blocking techniques employed.
2. *Data pre-processing*: The database owners then individually pre-process the QIDs used for blocking and comparison of records in their own database based on the requirements specified during the first stage. Examples of data pre-processing include case normalisation and tokenisation of strings, and standardisation of dates [1].
3. *Blocking (Indexing)*: Each record is evaluated using an agreed blocking mechanism, based on which records are assigned into blocks, each represented by a unique blocking key [35,36]. Depending on the blocking technique employed, records can either be assigned into a single block, or multiple blocking keys are generated per record if a technique such as locality sensitive hashing is employed [37,38]. In the comparison stage, only records that are in the same block will then be compared in detail with each other. Blocking substantially reduces the otherwise quadratic computational cost of comparing every possible record pair across the databases [35].

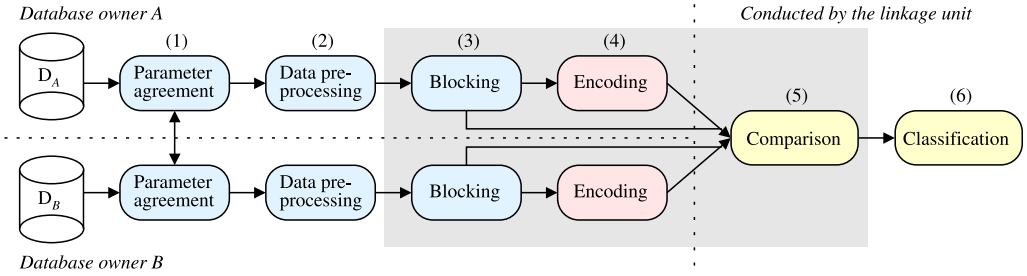

Fig. 1. An overview of the PPRL process as described in Section 2.2. In comparison to traditional record linkage, the encoding stage (4) is specific to the PPRL process, while stages (3) to (5) will be executed in a privacy-preserving manner (shown in the shaded area).

If a private blocking technique is employed, the blocks and/or the block metadata will be encoded or perturbed to prevent the leakage of sensitive information from the blocking stage. Perturbing block sizes, as we discuss in Section 3.1, aims to prevent group disclosure [15] (such as a very small block referring to individuals of a minority ethnicity [39]). However, perturbing blocks is generally neither preventing identity nor attribute disclosure [11], where the former is the main objective of DP, as we discussed in Section 2.1.

4. *Encoding:* Each record is transformed using an encoding technique to mask the actual sensitive QID values on which linkage is to be performed [2]. Therefore, encoding aims to prevent attribute disclosure [15], because this can lead to identity disclosure [11]. Each encoding is assigned an anonymised record identifier, which we assume does not reveal any information about the entity represented by that record. The encoding method employed can vary based on the PPRL technique used. They include homomorphic encryption for secure multi-party computation-based setups [7,9,40], hashed match-keys [41,42], BF encoding [10], tabulation based hashing [43], two-step hash encoding [44], and reference set-based encoding [45].
5. *Comparison:* Similarities are calculated on the generated encodings, where the similarity measure used depends on the type of encodings generated. For example, the similarity between bit arrays (as used for BF based encoding) can be calculated using a set-based similarity function such as the Dice coefficient [2]. Generally, as shown in Fig. 1, the comparison of encoded record pairs is performed by a linkage unit.
6. *Classification:* The classification of record pairs into matches, $\mathcal{M}$, non-matches, $\mathcal{N}$ (and for some classification techniques potential matches that require manual clerical review [1]), depends on the overall similarity scores calculated in the comparison stage. Techniques such as threshold-based classification are commonly employed [1], with recent advancements exploring machine learning and especially deep learning based techniques [46–48]. As per Definition 2, the database owners will only receive the anonymised identifiers of the encodings that were classified as matches (are in the set $\mathcal{M}$), and they should learn nothing else about the other databases or the outcome from the linkage unit [16].

For the set of classified matches, the database owners will then transfer the de-identified payload data along with their corresponding anonymised record identifiers to the data merger [24]. From these, the data merger will generate the linked data set by attaching the corresponding payload data of records to be used by the data consumers. Such linked data sets can be used in research studies (such as the medical or educational details of the individuals whose records were linked) and in operational systems (such as to identify individuals suspected of fraud or crimes).

2.3. Privacy objectives of PPRL

As per Definition 2 of PPRL, throughout the different stages of the linkage process, it is crucial that no participating party learns anything more than what is necessary to obtain an accurate linkage outcome, that is to correctly classify all true matches across the databases being linked. In common with most existing PPRL techniques [49], we assume that all participating parties are semi-honest; commonly referred to as the Honest-But-Curious threat model [5]. This model assumes that all parties involved in the linkage process follow the linkage protocol, but they can attempt to infer sensitive information based on their own data, the data received from other parties, and any external knowledge [2].

Accordingly, in addition to the linkage objective discussed in Section 2.2, we outline three privacy objectives for PPRL, focusing on the different parties involved: (1) the database owners, (2) the linkage unit, and (3) the data merger [24].

2.3.1. Privacy objectives at the database owner

A database owner must not learn any sensitive information about the data owned by the other participating database owners. Rather, a database owner can only learn which of their records are included in the set of matches, $\mathcal{M}$. Hence, any information exchanged in the blocking stage and the encodings generated must be anonymised to protect the privacy of the individuals (in the database of the database owner) whose data is being linked.

Formal privacy mechanisms such as DP have been adopted to provide privacy guarantees for the sensitive information being exchanged between the database owners, as we review in Section 3. All these techniques aim to satisfy this privacy objective, ensuring that the participating database owners do not learn any sensitive information about the data held by the other database owners.

2.3.2. Privacy objectives at the linkage unit

The linkage unit must not be able to infer any sensitive information about the entities (or their sensitive QID values) even when analysing the encoded databases received from the database owners for comparison and classification. Unless a secure multi-party computation-based PPRL technique is employed for the comparison stage, the privacy guarantees offered by techniques satisfying the privacy objectives at the database owner do not extend to the complete PPRL process. Rather, they only protect the information exchanged amongst the database owners at that specific stage.

Hence, the PPRL process can be rendered vulnerable if the linkage unit can exploit the encodings it receives from the database owners to infer sensitive information about the entities whose records are being linked [11]. Alternatively, if the encodings are guaranteed to be secure, exchanging them for comparison (within the expectations of the PPRL protocol) will not be a concern given it is assured that such encodings cannot be exploited [12].

One-way hashing is an example of how such secure encodings can be generated [41]. However, the drawback of using such hashing techniques is that they only allow for exact matching because the

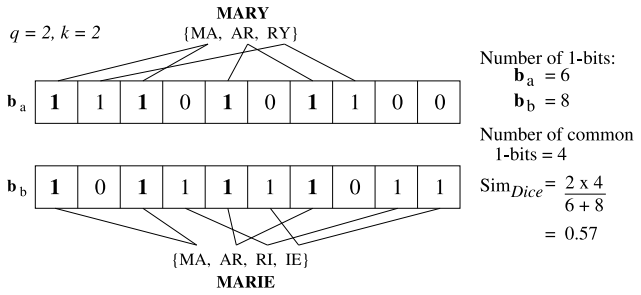

Fig. 2. An example BF encoding for the two first names *Mary* and *Marie*. Each name has been split into q-grams of length $q = 2$ (bigrams), and encoded into a bit array of length $l_b = 10$, using $k = 2$ independent hash functions.

underlying plaintext values have to be exactly the same to generate the same hash value [41]. Since practical linkage applications are likely deployed on databases that can contain variations and errors in the QID attributes being compared, employing exact matching techniques on such data will potentially result in many missed links [19]. Furthermore, such one-way hashing techniques are also known to be vulnerable to frequency and dictionary attacks [2,50].

Despite lacking formal privacy proofs, (non-DP) perturbation-based PPRL techniques are popular because they allow approximate comparison of QID values that might contain variations and errors, offering a better trade-off between privacy and linkage quality (utility). But to allow for variations means that the employed encodings, unlike one-way hashing techniques [41], will preserve certain similarity characteristics of the actual QID values which have shown to be vulnerable to attacks [12]. Recently developed perturbation-based PPRL techniques have integrated controls to prevent the leakage of sensitive information [44,45]. However, these techniques, still, do not provide formal privacy guarantees.

Alternatively, as explored through bit flipping [13] (as we discuss in Section 3.2), the generated encodings can be modified to satisfy some type of formal privacy requirement. These modifications, commonly known as *hardening* [13,51,52], are typically applied independent of the encoded sensitive QID values. Unfortunately, such hardening can lead to a distortion of the similarity characteristics between encodings and therefore a possible degradation of linkage quality [53]. The fundamental incompatibility of hardening with the objective of record linkage may raise concerns as to the extent to which such privacy mechanisms, applied as a post-encoding step, make sense for PPRL. From a record linkage perspective, it is crucial that any form of privacy mechanism employed must not impact the accurate classification of matches and non-matches [20].

2.3.3. Privacy objectives at the data merger

The data merger, the party which based on the matched record pairs generates the payload data set (the linked data set) [24], must not be able to infer sensitive information from the anonymised identifiers of the classified matches it received from the linkage unit, and the corresponding payload data received from the database owners.

From a PPRL perspective, it is expected that this privacy objective will be met due to the inherent design of the PPRL workflow. This is because the data merger and data consumer would only have access to anonymised identifiers and the payload data (such as health or education information), where we assume neither of which will reveal information that can be exploited to reidentify individuals. Furthermore, an important assumption in PPRL is that the database owners are willing to share payload data [49], and therefore their exchange would not be a threat to privacy unless any of the participating parties collude [24].

2.4. Bloom filter encoding for PPRL

A Bloom Filter (BF), b, is a probabilistic data structure containing l_b bit positions, $b[i]$, where $0 \leq i < l_b$, with all bits initially set to 0. In the context of PPRL, sensitive string values, such as personal names and addresses, are split into substrings of q characters (referred to as q-grams) and encoded into BFs using k independent hash functions [10]. These hash functions output a value between 0 and $l_b - 1$, and the bit positions corresponding to each hash outcome are set to 1. A set-based similarity measure, such as the Dice or Jaccard coefficients [2], can be used to calculate the similarity between two BF encodings. We illustrate an example in Fig. 2, where two first name values are encoded into two BFs. Despite the name values being different, the BFs share four common 1-bits due to their common q-grams, thereby enabling approximate matching. Similar BF-based techniques have also been proposed to encode numerical values [54–56], geographical locations [57], and hierarchical coding systems [58].

BFs are popular for practical PPRL applications due to the balance in linkage quality, scalability, and privacy they offer [2]. In practical applications, BF encodings are often considered as sufficiently private as the mapping of q-grams to their bit positions is dependent on the hash functions and secret key used, which are only known to the database owners. Additionally, the possibility of hash collisions [10] makes it challenging for an adversary to reidentify the q-gram set encoded in a BF (thereby aiming to prevent attribute disclosure [15]). However, recent studies have shown that BF encodings can be susceptible to frequency and pattern mining based attacks without requiring the adversary to have prior knowledge on the encoding parameters [11].

3. The use of differential privacy in PPRL

In this section, we review existing PPRL techniques that employ DP to provide formal privacy guarantees. Our review is based on studies identified through systematic searches of the PPRL literature and a comprehensive analysis of all citations to articles, reports, and books which discuss the use of DP for PPRL. In Section 3.1 we review techniques that enforce DP in the blocking stage, while in Section 3.2 we discuss techniques that enforce DP in the encoding stage. To support our claims, in Appendix A we provide key quotes from these PPRL works that employ DP.

3.1. Differential privacy for blocking

In the context of PPRL, the purpose of blocking is to improve the efficiency of the linkage process by reducing the number of candidate record pairs that need to be compared [35,59]. This is achieved by assigning each record one or more blocking key(s) based on the technique employed. The linkage unit receives the encoded records along with their blocking keys, from which it can generate the candidate record pairs to be compared.

Over the years, several studies have investigated the privacy aspects of blocking to ensure that no party involved in the PPRL process would be able to exploit the blocking information shared or the characteristics of the blocks themselves [9,16,60]. For example, if a particular block contains only a few records due to their unique QID values, then the block size itself can become sensitive information that could be exploited (via group disclosure). To mitigate such vulnerabilities, DP was adopted to enforce formal privacy guarantees for blocking in PPRL [8,16].

From a PPRL perspective, the use of DP during the blocking stage aims to mitigate the risk of an adversary being able to distinguish records amongst two adjacent blocks. Conceptually, two adjacent blocks are expected to differ in a single record. DP is incorporated through the addition of random noise, which can be achieved either through the addition of faked records or the suppression (removal) of actual database records.

Table 1
A chronological summary of differentially private blocking techniques for PPRL, characterising if faked records are added (Add) and/or if actual database records are suppressed (Sup), the structure of the blocks generated (disjoint versus overlapping), and the protocol setting based on the number of parties involved (where a three-party protocol involves an external linkage unit). All these techniques adopt global DP and add noise based on a Laplace distribution.

Technique	Block modifications	Blocking structure	Protocol setting
Inan et al. [8] (2010)	Add, Sup	Disjoint	Two-party
Kuzu et al. [40] (2013)	Add	Disjoint	Two-party
Cao et al. [9] (2015)	Add, Sup	Disjoint	Three-party
He et al. [16] (2017)	Add	*Not specified*	Two-party
Wu et al. [39] (2022)	Add	Disjoint	Three-party

In particular, there are three approaches to noise addition that can be employed: (1) naive addition of faked records which can result in the introduction of false matches; (2) systematic addition of faked records using attribute values outside of the database domain, thereby preventing false matches; and (3) suppression of database records which can result in a loss of true matches. Existing DP-based blocking techniques for PPRL use a combination of approaches (2) and (3), specifically to avoid the introduction of false matches.

In Table 1, we chronologically summarise DP-based blocking techniques for PPRL, and review each technique in detail next.

Inan et al. [8] (2010): This study proposed a two-party blocking protocol that uses aggregate range queries to query similar records and generate disjoint blocks. As such, this blocking technique is limited to numerical data (or categorical data by assigning each category a unique numerical value). The generated blocks are perturbed with random noise drawn from a Laplace distribution. Positive noise is added by creating faked records with attribute values outside of the domain, while negative noise is added by suppressing (removing) records. These noisy blocks, along with the query ranges used to generate the blocks (referred to as the block extent), are then transferred to the database owner serving as the linkage unit.

To enforce formal privacy during comparison and classification, the candidate record pairs are matched using a secure multi-party computation protocol. Since the database owners have no prior knowledge of which records result in matches [9], the suppressed records in one database are then compared against all records of the other database to avoid missed links. To reduce the number of secure multi-party computation-based operations performed despite blocking, a secure multi-party computation budget, agreed upon by the database owners, is used to limit the number of operations.

Kuzu et al. [40] (2013): This study introduces a DP-blocking technique that relies on an independent third-party to generate a blocking structure using a public reference database. This block structure, which includes the features of blocks, is shared with the database owners who assign each of their records to the block whose features are most similar to that of the record (resulting in disjoint blocks). The protocol proceeds on a two-party linkage model, requiring that at least one database owner shares its records with the other. To enforce privacy on the released blocks, records are encrypted using the Paillier cryptosystem [61], and the block sizes are perturbed using Laplacian noise. To avoid missed links, only positive noise is incorporated by adding faked records. To ensure that faked records do not result in false matches, they are assigned empty values for string attributes and negative values for numerical attributes.

Cao et al. [9] (2015): This study argues that the knowledge of both the noisy block sizes and the set of matches $\mathcal{M}$ (the final outcome of linkage) is sufficient to exploit non-matching record pairs; thereby violating DP guarantees. The aforementioned blocking techniques [8, 40] are hence deemed insecure, as they disclose the noisy block sizes to at least one participating database owner, while $\mathcal{M}$ in any case is known to all database owners. To address the above concerns, the authors proposed a three-party PPRL protocol, eliminating the need for database owners to exchange their blocks.

Similar to previous work, it is expected that the generated blocks are mutually exclusive and that the block sizes are perturbed using both positive and negative noise. The sensitive QID values are encoded using the Paillier cryptosystem [61] to ensure formal privacy guarantees. The protocol supports both string and numerical values, while categorical attributes are pre-processed by mapping each category to a unique numerical value. In continuing work, Rao et al. [62] proposed an efficient indexing scheme based on this blocking technique.

He et al. [16] (2017): This study argues that the use of negative noise can violate DP guarantees in multi-party linkage setups where the ratio of matches and non-matches is sensitive to the suppressed records (we refer the readers to the extensive mathematical proofs provided by the authors [16]). This limitation is inherent since at the point of suppression, the database owner will not know if a suppressed record contributes to a match or not. The authors also demonstrate that the knowledge on the number of pairwise comparisons made, even if they are compared and classified by a third-party linkage unit [9], can be exploited.

To overcome these weaknesses, the authors developed a two-party PPRL protocol that perturbs block counts only through positive noise (by adding faked records outside of the domains of attributes). To balance linkage quality and privacy, the authors propose *output constrained differential privacy*, aiming for weaker privacy guarantees than that achieved using secure multi-party computation protocols (resulting from $\epsilon > 0$). The protocol uses a greedy matching approach where the insights gained through the matching of records within a particular block are used to refine the subsequent blocks prior to comparison. This technique is reported to improve comparison efficiency by at least two magnitudes compared to that of a secure multi-party computation-based protocol. Yet, Rao et al. [62] (which uses the DP blocking method proposed by Cao et al. [9]) in their later work argue that this blocking technique also violates DP guarantees because all database owners learn about the block extents and the noisy block sizes.

Wu et al. [39] (2022): This study proposes a three-party DP-blocking protocol which uses positive Laplacian noise. Unlike prior work, it considers both cost and fairness when determining noise levels, motivated by the observation that despite using DP, blocking can still leak sensitive information about protected (minority) groups when based on protected attributes such as age or ethnicity. Cost constraints limit excess overhead from linking faked records. This protocol uses BF encodings, assigning each record to the block that contains all other records that share the same pattern for a selected subset of bit positions, which results in disjoint blocks.

This protocol uses *feature-level differentially private blocking*; where for each block, the number of faked records added is calculated separately for each protected group. The faked records are hardened versions of selected records within the blocks themselves. However, the authors do not describe how the noisy blocks are communicated or the extent to which block characteristics are known by other parties. Hence, formal privacy proofs will extend beyond the blocking phase only if a secure multi-party computation-based matching protocol is adopted, since BFs are known to be vulnerable [11].

3.2. Differential privacy for encoding

Existing work that employ DP in the encoding stage of the PPRL process can be classified into three categories based on the information they perturb: (1) techniques that perturb the metadata required by a PPRL technique, (2) techniques that perturb the record encodings generated, and (3) techniques that perturb the sensitive QID values being encoded. In Table 2 we summarise PPRL encoding techniques, and we review each category next.

Table 2
A chronological summary of differentially private encoding techniques for PPRL, characterising the type of information perturbed and the differentially private noise mechanism adopted.

Technique	Information perturbed	DP noise mechanism	Type of DP
Bonomi et al. [63] (2012)	Metadata	Laplace	Global
Schnell et al. [13] (2016)	Encodings	Bit flipping	Local
Sun et al. [56] (2018)	Encodings	Bit flipping	Local
Vaiwsri et al. [64] (2018)	Encodings	Bit flipping	Local
Ranbaduge et al. [48] (2023)	Encodings	Bit flipping	Local
Dorgbefu et al. [65] (2025)	QIDs	Laplace	Global

3.2.1. Techniques that perturb encoding metadata

Bonomi et al. [63] **(2012)**: This study introduces a PPRL encoding technique which relies on a reference base generated using the frequent q-grams extracted from the databases being linked. Given that such q-grams and their frequencies are sensitive information that can be exploited by an adversary [11], the protocol masks the original frequencies using Laplacian noise. The frequent q-grams and their perturbed frequencies are exchanged between the database owners, and one database owner is tasked with constructing and sharing the reference base generated using the frequent q-grams. The sensitive records are then encoded based on their similarities with the frequent q-gram embeddings in the reference base. The encoded vectors are subsequently transferred to a linkage unit for comparison and classification.

Although the linkage unit does not have access to the reference base from which the encodings were generated, the use of DP only provides a formal privacy guarantee for the frequent q-grams that were extracted, but not for any of the other data exchanged and used during the PPRL process.

3.2.2. Techniques that perturb encodings

Techniques that add differentially private noise to PPRL encodings have been relatively more popular than other methods, for two main reasons: (1) if the encodings themselves have formal privacy guarantees, it is not necessary to adopt a computationally expensive linkage setup (such as secure multi-party computation) to compare encoded records, and (2) all these techniques were proposed for BF-based encodings, which are widely adopted in real-world PPRL applications [2,4,66].

Schnell and Borgs [13] **(2016)**: This study introduced a BF hardening technique that relies on perturbing BF encodings through differentially private bit flipping, primarily aiming to prevent deterministic attacks [67,68] on the encodings. **Vaiwsri et al.** [64] **(2018)** extended on this work focusing on linkage quality aspects of the bit flipping process. A similar bit flipping mechanism was also proposed by **Sun et al.** [56] **(2018)** to encode numerical values for PPRL. All these bit flipping techniques were motivated by earlier work by Alaggan et al. [29] and Erlingsson et al. [23], who were among the first to explore DP for BFs through the BLoom and flIP (BLIP) and Randomised Aggregate Privacy-Preserving Ordinal Response (RAPPOR) techniques, respectively.

Bit flipping aims to add differentially private noise through the modification of randomly chosen bit positions, and is referred to as a *randomised response technique* [69] since each bit is flipped independently of any other bits. Considering that random noise is added to the individual BFs, bit flipping, in this context, provides *local* DP, in contrast to the *global* DP achieved by adding noise to data that has been aggregated across multiple or all records in a database [29,70].

As common to all DP mechanisms, privacy guarantees are controlled through the privacy parameter, ϵ. For BFs, Alaggan et al. [29] proposed to enforce this parameter using a flip probability, f_p, which specifies the amount of noise to be added to a BF. However, the flipping mechanisms used by the BLIP and RAPPOR techniques have slight variations in how modifications are made. In BLIP, using Eq. (2), the flipping function is guaranteed to output the inverse of the input bit for every position chosen to be flipped. Whereas in RAPPOR, Eq. (3), the output of the flipping function can either be 0 or 1 depending on the randomness adopted.

$$b_B[i] = \begin{cases} 1 - b[i], & \text{with probability } f_p \\ b[i], & \text{with probability } 1 - f_p \end{cases} \tag{2}$$

$$b_R[i] = \begin{cases} 1, & \text{with probability } \frac{1}{2} f_p \\ 0, & \text{with probability } \frac{1}{2} f_p \\ b[i], & \text{with probability } 1 - f_p \end{cases} \tag{3}$$

Depending on the technique used, the flip probability, f_p, required to achieve a particular ϵ-DP differs. Regardless of the mechanism used, a higher value for f_p leads to increased perturbation of the BFs, which in turn provides improved privacy. However, the downside of increased perturbation (as for any noise addition technique) is that it can degrade linkage quality.

The amount of privacy achieved through bit flipping depends on k, the number of hash functions used for BF encoding, and how the adjacency between two BFs is defined. This is because every item (generally a q-gram in PPRL) is hashed k times (with likely $k > 1$), as a result of which a single item can affect k bit positions. As such, two adjacent BFs that differ in one item can differ in up-to k bit positions at most. Interestingly, ϵ does not depend on the length of the BF, l_b, which is typically considered a crucial configuration of the BF encoding setup. Erlingsson et al. [23] claim that although l_b affects the number of hash collisions, the ambiguity provided by collisions is neither adequate nor necessary to guarantee ϵ-DP.

As such, for BLIP proposed by Alaggan et al. [29], the privacy loss (ϵ_B) can be calculated using Eq. (4). For the PPRL hardening technique proposed by Schnell and Borgs [13] (which uses the RAPPOR mechanism [23]), the loss in privacy (ϵ_R) can be calculated using Eq. (5).

$$\epsilon_B = k \cdot \ln\left(\frac{1 - f_p}{f_p}\right) \Rightarrow f_p = \frac{1}{1 + e^{\epsilon_B / k}} \tag{4}$$

$$\epsilon_R = 2 \cdot k \cdot \ln\left(\frac{1 - \frac{1}{2} f_p}{\frac{1}{2} f_p}\right) \Rightarrow f_p = \frac{2}{1 + e^{\epsilon_R / (2 \cdot k)}} \tag{5}$$

Ranbaduge et al. [48] **(2023)**: This study introduced a deep learning based PPRL protocol that uses BLIP-hardened BFs as input. The authors discuss that two BFs in a given encoded database can differ by at most $2 \cdot n_m \cdot k$ bit positions, where n_m is the largest number of q-grams that will be encoded into a BF in this database. Accordingly, they scale Eq. (4) to account for n_m and calculate the privacy loss, ϵ_{DL}, as per Eq. (6).

$$\epsilon_{DL} = 2 \cdot n_m \cdot k \cdot \ln\left(\frac{1 - f_p}{f_p}\right) \Rightarrow f_p = \frac{1}{1 + e^{\epsilon_{DL} / (2 \cdot n_m \cdot k)}} \tag{6}$$

It is important to note that this work by Ranbaduge et al. [48] does not introduce a new flipping mechanism, instead it introduces a different approach to quantify the privacy loss. However, since the value of n_m can be influenced by outliers, such as unusually long names or addresses [11], it is possible that the number of bits to be flipped to achieve a sufficient privacy level is overestimated. This may lead to an unnecessary drop in linkage quality with no improvements to privacy.

Wu et al. [70] **(2023)**: This study introduced a PPRL technique which also uses BLIP-hardened BFs as input. While the flipping mechanism is the same as that proposed by Alaggan et al. [29], the authors do not account for k when calculating the amount of privacy achieved for a particular f_p. The authors demonstrate local DP using adjacent BFs that differ in one bit position; an assumption that however holds only if $k = 1$. This is because each q-gram likely affects k bit positions in a BF encoding (or slightly less if there are collisions, as can happen if $k > 1$). Given that using a single hash function is highly unlikely in a practical PPRL application, we do not further consider this variation of privacy loss calculation in our review.

3.2.3. Techniques that perturb sensitive quasi-identifiers

Dorgbefu Jnr et al. [65] (2025): This study introduces a PPRL protocol where, to our understanding, DP is adopted to anonymise unique entity identifiers within a database (such as SSNs or voter identification numbers). Blocking and comparison are then performed in a secure multi-party computation-based setup using homomorphic encryption or additive secret sharing (where the noisy QID values are split into two components and held by different parties). However, the application of DP on such unique identifiers, which are specific to the individual databases being linked, is questionable since such identifiers might not be useful for linkage. Furthermore, while the use of both DP and secure multi-party computation will ensure formal privacy guarantees, it is expected that this protocol will be computationally expensive when linking large databases.

4. Discussion of existing DP-PPRL techniques

In this section, we provide a conceptual discussion of existing PPRL techniques that employ DP, as we reviewed in the previous section. We first discuss the implications of conducting differentially private blocking. Then, in Section 4.2, we discuss the privacy implications of differentially private encodings generated through bit flipping, with a particular focus on their implementation aspects.

4.1. Differentially private blocking techniques

As reviewed in Section 3.1, existing DP-blocking techniques for PPRL enforce privacy guarantees only on the blocks and associated information generated during the blocking stage. Hence, all such techniques require that the comparison and classification of records are also to be performed in a setting with formal privacy guarantees. The end-to-end linkage process is entirely private only if a technique such as secure multi-party computation or homomorphic encryption is adopted for the comparison stage [2].

Furthermore, all known DP-blocking techniques for PPRL aim to achieve privacy by perturbing the sizes of the blocks generated (by adding faked records and possibly suppressing database records). It is important to note that none of these DP-blocking techniques modify any database records or their encodings. However, despite perturbing the sizes of blocks, the database owners will learn which of their own records do not occur in the other database,[4] unless a PPRL protocol without data backflow is being used [24]. Furthermore, perturbing block sizes only prevents group disclosure because it masks the number of records assigned to a particular block. Therefore, DP-blocking techniques do not provide privacy against identity disclosure, as expected in DP.

PPRL protocols which incorporate secure multi-party computation techniques are commonly referred to as hybrid techniques [62,72], since they use some form of perturbation (noise addition for DP-blocking) to reduce the number of candidate pairs that have to be compared in a resource-intensive secure multi-party computation-based setup [8,40]. However, although blocking improves the efficiency of the linkage process to a certain extent [8], the computational costs of performing PPRL in such secure settings, when using secure multi-party computation-based techniques, remains high [40,73].

[4] This is similar to private set intersection, where even if a secure multi-party protocol is used, each party learns which of their own elements are not part of the intersection [71].

4.2. Differentially private encoding techniques

The impact of bit flipping, on both privacy and linkage quality, can vary drastically depending on its implementation. As we discussed in Section 3.2, the expectation is that for each bit position $b[i]$ in a BF, where $0 \leq i < l_b$, a random coin toss is simulated by drawing a random value, r, with $0 \leq r \leq 1$. If $r \leq f_p$, the particular bit position is flipped. The flipping process generally depends on a pseudo-random number generator [2], which randomly selects the bit positions to be flipped, while the output of the flipping function is determined by the flipping mechanism employed, as following Eq. (2) or (3). The randomness of this bit flipping process is controlled by a secret seed, which is generally agreed upon by the database owners. In the current application of bit flipping for PPRL, there are four distinct approaches to setting the random seed.

1. *Setting the same seed for all BFs:* This approach renders the bit flipping process deterministic, since, for each BF the pseudo-random number generator will sample the same subset of bit positions to be flipped. If the RAPPOR flipping mechanism (Eq. (3)) is to be used, each of these sampled bit positions will also be set to the same value. Hence, this approach is unlikely to provide any privacy because it deterministically modifies a fixed set of bit positions across an entire database.
2. *Setting no seed (fully random):* From an implementation perspective, every BF is allocated a completely random seed value which is often assigned by the system based on time. Hence, the set of bit positions flipped and the values assigned to these positions (for RAPPOR, Eq. (3)) are entirely random for each BF. Consequently, even exact matches might produce different BF encodings [53]. As such, linkage performed using this approach would not be reproducible. It must be noted that in the original proposal of BLIP for PPRL by Schnell and Borgs [13], a secret seed was not specified.
3. *Setting a seed once for the entire database:* In existing implementations of bit flipping for PPRL, the database owners agree on a shared secret seed that is used to harden their databases. This seed value is set at the beginning of the hardening process, once for the entire database to be hardened [74]. Hence, the bit positions flipped and the values they are assigned to is dependent on the ordering of records in the databases. Similar to the second approach, this approach will also affect linkage quality; however, the linkage process will remain reproducible (as long as all records, their number, and their order in the database remain unchanged).
4. *Using salt values as seeds:* The use of a record-specific salt value, instead of a random seed, is an alternative approach to control the bit flipping process. Similar to the salting mechanism used for hardening in PPRL [2], a stable, complete, and accurate QID attribute (such as year or place of birth) can be used as the salt which controls the flipping process. For records that represent the same entity, this salting attribute will, with high likelihood, have the same value. Hence, the bit flipping process will also be the same for such records, thereby preserving their similarities. However, a general weakness of salting is that it can only be adopted if the databases share a common suitable QID attribute of high quality.
   However, selecting a seed value based on the characteristics of the records themselves can be considered to be a deterministic mechanism. For example, if the year of birth is used as the salt, BFs corresponding to individuals born in the same year will all be flipped similarly. Hence, the extent to which this approach enforces DP guarantees must be further explored.

The Reference value based BLIP BF (RBBF) method proposed by Vaiwsri et al. [64] can be considered an alternative to salting. RBBF

incorporates a secret seed to the flipping process, but the choice of this seed depends on the record being hardened. The database owners extract a set of reference values from a public database, and for each record, the set of its $z \geq 1$ most similar reference values are identified. These reference values are then used as the random seed(s) that control(s) the flipping function. If $z > 1$, bit flipping is applied z times (each with a different random seed), and the resultant BFs are concatenated to generate the final hardened BF. If multiple records share the same z most similar reference values, their corresponding BFs will be hardened in the same way because the same bit positions will be flipped. By doing so, RBBF aims to preserve the similarities between the original BFs.

Irrespective of the approach used to set the secret seed, unless the same random seed is used for all BFs, it can be expected that bit flipping will add noise to the original BF encodings. The key concern, however, is if bit flipping can protect the entities (people) represented by their BFs, and if the resulting perturbation is sufficient to provide strong privacy guarantees.

Conceptually, bit flipping will guarantee ϵ-DP, provided that f_p is configured according to the specific flipping mechanism employed (Eqs. (4) to (6)). But its application in the context of PPRL does not achieve the intended goal of protecting the individuals whose records are being linked. This is because *bit flipping [29] was proposed to protect the individual items that are encoded into a BF*, and not the entity represented by the BF. In the work by Alaggan et al. [29], the assumption was that a BF corresponds to a user profile and the items encoded were the user's (potentially sensitive) interests. The objective of this work adopting bit flipping was to protect the user's interests that were encoded into the BF, and not the user themself. Therefore, bit flipping as employed by Alaggan et al. [29] was aimed at protecting against attribute disclosure (each interest of a user is an attribute of the user) but not identity disclosure [15].

In PPRL, these items translate to the q-grams generated from the sensitive QID values of the entities. However, solely masking certain q-grams would not be sufficient to protect the entity represented by a given BF. This is because bit flipping is a randomised response technique designed to provide local DP. As a result, a particular BF can still be distinguishable amongst other encodings as it can retain certain unique characteristics that make it vulnerable, and potentially result in identity disclosure. For instance, if a particular record is length-vulnerable [11] due to long QID values that result in a q-gram set that is the unique longest in the database, then the Hamming weight (number of 1-bits) of the corresponding BF can remain vulnerable even after bit flipping. Moreover, since the flipping technique relies on some form of randomness, there is no guarantee as to which bit positions are flipped. Hence, given that an individual q-gram is encoded into k different bit positions using k independent hash functions, even an individual q-gram can still be vulnerable even after some of its bit positions have been perturbed.

As an example, consider a scenario where sensitive QID values are encoded into a BF of length $l_b = 1000$ using $k = 20$ independent hash functions. If a long q-gram set with length, say, $l_s = 30$ was to be encoded, the output BF will have $k \cdot l_s = 30 \cdot 20 = 600$ 1-bits at most (assuming no collisions). If these encodings were to be hardened using BLIP (Eq. (4)) with the highest recommended flip probability of $f_p = 0.2$ (20%) [13], then on average 10% of the 1-bits (alongside 10% of the 0-bits) would be affected. Even if all of the chosen 1-bits were to be randomly set to 0, only $0.1 \cdot 600 = 60$ 1-bits will be lost. This means the encoding for this BF will still retain its 1-bit patterns across the remaining 540 1-bit positions. Similarly, from the perspective of an individual q-gram, each q-gram will likely contribute with $k = 20$ 1-bits to the encoded BF (or slightly less in case of collisions). Assuming 10% of a q-gram's 1-bits are flipped on average, this will result in the loss of only $0.1 \cdot 20 = 2$ of the 1-bits, thereby leaving 18 bit positions corresponding to the q-gram unchanged. The patterns of such 1-bit positions can potentially still be exploited in a pattern mining attack, as we experimentally evaluate in Section 5.2.

Table 3
A comparison of the loss in privacy across the three different bit flipping mechanisms (ϵ_B, ϵ_R, and ϵ_{DL}) for varying flip probabilities (f_p) and number of independent hash functions (k) used to encode BFs. We assume that $l_b = 1000$, while the value of $n_m = 50$ (based on the experimental setup used in [48]).

k	ϵ	$f_p = 0.05$	$f_p = 0.1$	$f_p = 0.2$	$f_p = 0.3$	$f_p = 0.5$
1	ϵ_B	2.95	2.20	1.39	0.85	0.00
	ϵ_R	7.33	5.89	4.39	3.47	2.20
	ϵ_{DL}	294.44	219.72	138.63	84.73	0.00
5	ϵ_B	14.72	10.99	6.93	4.24	0.00
	ϵ_R	36.63	29.44	21.97	17.35	10.99
	ϵ_{DL}	1472.22	1098.61	693.15	423.65	0.00
10	ϵ_B	29.44	21.97	13.86	8.47	0.00
	ϵ_R	73.27	58.89	43.94	34.69	21.97
	ϵ_{DL}	2944.44	2197.22	1386.29	847.30	0.00
20	ϵ_B	58.89	43.94	27.72	16.94	0.00
	ϵ_R	146.54	117.78	87.89	69.38	43.94
	ϵ_{DL}	5888.88	4394.45	2772.59	1694.60	0.00
40	ϵ_B	117.78	87.89	55.45	33.89	0.00
	ϵ_R	293.08	235.55	175.78	138.77	87.89
	ϵ_{DL}	11,777.76	8788.90	5545.18	3389.19	0.00

To achieve improved privacy means to increase the number of bits flipped (the amount of noise added), which can be controlled using f_p. In Table 3, we vary the values of f_p and k, and quantify the loss in privacy for the three different bit flipping mechanisms (Eqs. (4), (5), and (6)). As can be seen, most of these values are significantly larger than the theoretically recommended range of 0.001 and 1 for ϵ [25] and the real-world application average of 4.39, as we discussed in Section 2.1. Furthermore, for the same f_p, the flipping mechanism of BLIP [29] guarantees greater privacy than that of RAPPOR [23]. This can be attributed to the fact that BLIP guarantees to flip all randomly chosen bit positions using Eq. (2). In contrast, this guarantee does not hold for RAPPOR, where the outcome of flipping is independent of the input bit value, see Eq. (3). However, from a different perspective, one could note that adding randomness independently of the input bit values (as RAPPOR [23] does) provides a further layer of ambiguity on top of that achieved by flipping actual bit values, making it even more challenging for an adversary to reidentify the q-grams encoded in BFs.

When the maximum number of q-grams encoded in a BF, n_m, is factored into the calculation [48], following Eq. (6), the privacy loss is significantly higher for $f_p < 0.5$, as can be seen in Table 3. This is because the amount of noise added to the BF is based on perturbing $2 \cdot n_m \cdot k$ hash events. Although BLIP guarantees $\epsilon_B = 0.0$ and $\epsilon_{DL} = 0.0$ at the cost of flipping 50% of the bits in a BF, such perturbation, with high likelihood, will lead to a significant drop in linkage quality, as our results in Section 5.3 confirm.

On the other hand, with the RAPPOR technique proposed for PPRL [13], even flipping 50% ($f_p = 0.5$) of the bits in a BF yields moderate amounts of privacy only where $k = 1$. However, in practice, the number of hash functions used for BF-based encoding is commonly determined according to the concept of the *optimal number* of hash functions [2]. This value, referred to by k_{opt}, is data-dependent and optimised to limit the number of possible hash collisions. k_{opt} is calculated as $k_{opt} = \ln 2 \cdot l_b/n$, where n is the expected average number of q-grams that will be encoded into a BF [2].

In practical applications, where l_b is commonly set to $l_b = 1000$ [2], even if multiple attributes are encoded into one BF (resulting in higher values of n), the value of k_{opt} will, with high likelihood, be much larger than 1 [2]. Thus, although bit flipping will lead to a certain amount of perturbation of the encodings, the amount of noise required to achieve strong privacy guarantees is impractically high, and will substantially affect the quality of any obtained linkage results [75], as we experimentally evaluate next.

5. Experimental evaluation of bit flipping

We now evaluate the use of bit flipping for PPRL and present our findings. We first describe our setup in Section 5.1, and in Sections 5.2 and 5.3 we assess the impact of bit flipping on privacy and linkage quality, respectively. The source code for the experimental setup and detailed results (beyond what we present here) can be found on GitHub.[5]

5.1. Experimental setup

Our experiments were based on records sampled from the North Carolina Voter Registration (NCVR) database[6] [76], a real-world data set that contains personal information about registered voters. NCVR has previously been used to evaluate PPRL methods [2,39,44,45,48,53,70]. We used five sensitive quasi-identifier (QID) attributes: First Name (FN), Last Name (LN), Street Address (SA), Zip Code (ZC), and City (CI), and four attribute combinations: (1) Names Only (NO) with [FN, LN], (2) Names and Street Address (NS) with [FN, LN, SA], (3) Names, Street Address, and Zip Code (NSZ) with [FN, LN, SA, ZC], and (4) Names, Street Address, City, and Zip Code (NSCZ) with [FN, LN, SA, CI, ZC].

Based on the types of vulnerabilities of personal data identified in the context of PPRL [11], we sampled voter records based on characteristics that are known to increase their risk of reidentification, specifically the lengths and frequencies of QID values being encoded. We explored if bit flipping provides adequate privacy for such vulnerable records (as converted into q-gram sets), thereby preventing possible reidentifications. These characteristics are:

1. *Length of q-gram set*: Since every q-gram is encoded into k bit positions, the number of q-grams in a record q-gram set determines the number of 1-bits (Hamming weight) in the corresponding BF. Hence, BFs that encode longer QID values will have a higher Hamming weight, while those encoding shorter QID values will have a lower Hamming weight. The Hamming weights can therefore leak length information that can be exploited by an adversary, even without any knowledge of the encoding parameters [11].
2. *Frequency of individual q-grams*: Since a particular q-gram is hashed into the same k bit positions determined by the k independent hash functions, the frequencies of bit positions set to 1 in a BF database can retain frequency patterns of the q-grams encoded into them [77]. If an adversary has access to a database that is similar to the sensitive database from which the BFs were generated [12], he can exploit the corresponding frequency patterns to reidentify encoded q-grams [78].
3. *Average frequency of q-grams in record q-gram sets*: Here we assign a frequency score to each record q-gram set based on the total frequencies of the q-grams occurring in the record, averaged over the number of q-grams in a given set. This is a reflection of the q-gram combination in the record (whether the most frequent, rare, or a mix of q-grams occur together). If a particular record has a combination of rare q-grams, and thereby 1-bits in positions with very low Hamming weights, an adversary can infer that such records have unique characters in their sensitive QID values.

For each of these three characteristics, we selected 1000 unique q-gram sets from each extreme (such as the 1000 shortest and 1000 longest q-gram sets), as well as 1000 from around the average, resulting in nine subsets of 1000 q-gram sets each. In addition, we created three random samples of 1000 q-gram sets each (from where we then averaged results), leading to a total of ten subsets.

To avoid duplicates, we ensured that all q-gram sets in the three random samples were unique across all the subsets. However, the nine subsets based on the length and frequency characteristics may contain overlapping q-gram sets, as certain q-gram sets can exhibit more than one characteristic. For example, one of the longest q-gram sets can also contain the most frequent q-gram, in which case this q-gram will occur in two q-gram subsets. We provide further details about the characteristics of the sampled q-gram sets and provide the source code on GitHub.[7]

We then performed record linkage independently on each of the generated subsets. For every subset, we linked its 1000 q-gram sets to a second data set (named the *base data set*) that consists of the same 1000 q-gram sets concatenated with a sample of 1 million randomly selected other unique q-gram sets (where none of these q-gram sets occurred in any of the twelve subsets). Each of the linkages therefore linked 1000 q-gram sets with 1,001,000 q-gram sets.

We encoded the q-gram sets generated from QID values into BFs of length $l_b = 1000$ using the optimal number of hash functions [2], k_{opt}, as well as half and twice as many hash functions, $k_{opt}/2$ and $2 \cdot k_{opt}$. We implemented both bit flipping mechanisms used in PPRL: BLIP [39,48,70], Eq. (2), and RAPPOR [13,64], Eq. (3). To evaluate the impact of the amount of noise added, we varied the bit flipping probability as $f_p = [0.05, 0.1, 0.2]$, following values used in existing literature [13,48]. Since these flipping probabilities yielded high ϵ values, as can be seen in Table 4, we also set $f_p = 0.5$ to further restrict the loss in privacy. We employed the setup without a fixed random seed (as we described in Section 4.2), resulting in a different set of bit positions being selected for flipping in each BF. We used the linkage performed on the original BF encodings (without any bit flipping) as the baseline. For this setup, we expect that each of the BFs will correctly match with its corresponding BF in the larger data set resulting in 1000 exact matches for each subset.

Our evaluation focused on two dimensions: (1) *Privacy:* The extent to which bit flipping improves the privacy of the entity represented by an encoding, the objective of adopting DP in PPRL. (2) *Linkage quality:* The extent to which bit flipping affects the ability to correctly identify true matches, the objective of record linkage.

5.2. Privacy

We first calculated the loss in privacy, ϵ, for each of the different experimental setups, and summarise these values for varying k and f_p in Table 4. As can be seen from this table, even a flip probability of $f_p = 0.2$ (flipping 20% of the bits) leads to high ϵ values for both the BLIP and RAPPOR techniques. For BLIP, the highest level of privacy, $\epsilon_B = 0$, is achieved when the flip probability is set to $f_p = 0.5$. However, as we discuss in Section 5.3 below, flipping 50% of the bits of a BF encoding significantly degrades linkage quality (with zero correct links identified), thereby contradicting the objectives of performing record linkage.

To evaluate the extent of privacy offered by DP-based bit flipping, we conducted a basic frequency-based cryptanalysis attack [78] on the NCVR snapshot from which the subsets were sampled (because the subsets themselves are too small to mount a meaningful attack). This attack exploits the 1- and 0-bit patterns to first assign possible q-grams to bit positions, followed by the generation of feasible QID values from these possible q-grams. We expect that the success of such a fundamental attack suggests that more advanced attacks [77,79] will, likely, also succeed. For this attack, we limit the flip probability f_p to 0.1 and 0.2, since the privacy loss at $f_p = 0.05$ is very high (not much privacy is achieved) as can be seen in Table 4, while the utility is very low for $f_p = 0.5$ as we discuss in Section 5.3 below.

[5] See: https://github.com/SumayyaZiyad/dp-for-pprl.

[6] See: https://dl.ncsbe.gov/?prefix=data/.

[7] See: https://github.com/SumayyaZiyad/dp-for-pprl.

Table 4
The privacy loss (ϵ_B/ϵ_R), calculated for different attribute combinations and flip probabilities, f_p. For each setup, we show three rows with $k_{opt}/2$, k_{opt} (the optimal number of hash functions, shown in italics font), and $2 \cdot k_{opt}$. These values are consistent for all ten subsets since the privacy loss is dependent only on f_p and k.

	k	ϵ_B (BLIP [29])/ϵ_R (RAPPOR [23])			
		$f_p = 0.05$	$f_p = 0.1$	$f_p = 0.2$	$f_p = 0.5$
NO	32	94.2 / 234.5	70.3 / 188.4	44.4 / 140.6	0 / 70.3
	65	191.4 / 476.3	142.8 / 382.8	90.1 / 285.6	0 / 142.8
	130	382.8 / 952.5	285.6 / 765.6	180.2 / 571.3	0 / 285.6
NS	14	41.2 / 102.6	30.8 / 82.4	19.4 / 61.5	0 / 30.8
	29	85.4 / 212.5	63.7 / 170.8	40.2 / 127.4	0 / 63.7
	58	170.8 / 425.0	127.4 / 341.6	80.4 / 254.9	0 / 127.4
NSZ	12	35.3 / 87.9	26.4 / 70.7	16.6 / 52.7	0 / 26.4
	25	73.6 / 183.2	54.9 / 147.2	34.7 / 109.9	0 / 54.9
	50	147.2 / 366.4	109.9 / 294.4	69.3 / 219.7	0 / 109.9
NSZC	10	29.4 / 73.3	22.0 / 58.9	13.9 / 43.9	0 / 22.0
	20	58.9 / 146.5	43.9 / 117.8	27.7 / 87.9	0 / 43.9
	40	117.8 / 293.1	87.9 / 235.6	55.5 / 175.8	0 / 87.9

Table 5
Results of a frequency-based attack [78] aiming to reidentify the twenty most frequent QID values. These results are shown as the number of 1-1-correct/1-many-correct/wrong/no guesses. QID values were encoded into BFs of length $l_b = 1000$ using k_{opt} hash functions.

	No bit	BLIP		RAPPOR	
QIDs	flipping	$f_p = 0.1$	$f_p = 0.2$	$f_p = 0.1$	$f_p = 0.2$
FN	20/0/0/0	20/0/0/0	20/0/0/ 0	20/0/0/0	20/0/0/0
LN	20/0/0/0	20/0/0/0	20/0/0/ 0	20/0/0/0	20/0/0/0
NO	1/4/15/0	6/0/0/14	6/0/0/14	6/0/0/14	6/0/0/14
NS	9/7/0/4	3/4/0/13	2/4/0/14	9/7/0/4	9/7/0/4
NSZ	0/0/0/0	0/0/0/0	0/0/0/0	0/0/0/0	0/0/0/0
NSZC	0/0/0/0	0/0/0/0	0/0/0/0	0/0/0/0	0/0/0/0

We targeted the twenty most frequent QID values, with the results of this attack shown in Table 5. As can be seen, bit flipping offers no privacy improvement for when only a single QID attribute (first name or last name) is encoded into BFs. Interestingly, when we encode the NO attribute combination, the perturbed bit patterns resulted in an increase in the number of QID values reidentified correctly. This is likely due to bit flipping resulting in more unique BF bit patterns which in turn makes reidentification less challenging. These results reinforce our argument that applying DP on BFs through bit flipping is not protecting q-gram sets (and therefore BFs and individuals represented by these BFs) unless a much larger amount of noise is added (beyond what is acceptable from a record linkage quality perspective).

For the NS attribute combination, RAPPOR offers no privacy protection, while BLIP shows a decrease in the number of QID values correctly reidentified (the only case where bit flipping improves the privacy of QID values). On the other hand, for the NSZ and NSZC attribute combinations (where four or even five QID attributes are encoded) reidentification becomes impossible, with similar attack outcomes noticed even when no bit flipping is applied. This is expected, as encoding a larger number of QID values leads to q-grams from different attributes being encoded into the same BF which makes it challenging to reidentify the individual QID values [78].

5.3. Linkage quality (utility)

To evaluate the impact of bit flipping on linkage quality (which corresponds to utility of the linked data set), we analysed the linkage results for each of the ten subsets of 1000 q-gram sets. Particularly, we assessed if each q-gram set in a subset shared the highest Dice similarity [2] with its own q-gram set in the larger base data set of 1,001,000 q-gram sets, as described above. The outcome is considered to be a 1-1-correct match if a q-gram set in the subset has been matched with a single q-gram set in the base data set and it is the true match (without bit flipping, this is expected, as these will be exact-matching BFs). The use of an exact-match scenario was intended to demonstrate that if bit flipping degrades linkage quality even in such conditions, it is highly likely that the linkage quality would further degrade when linking approximate true matches (true matches with variations or errors).

In Fig. 3, we summarise the 1-1-correct matches for each subset across the range of flip probabilities (detailed results are provided in our GitHub repository). For both flipping mechanisms (BLIP and RAPPOR) the linkage quality shows similar trends across all subsets. As can be seen, linkage quality is still very high for both setups even for a flip probability of $f_p = 0.2$ (with an average of over 84% correct 1-1 matches for BLIP and over 99% correct 1-1 matches for RAPPOR). The only exceptions are the subsets with the shortest q-gram sets and those with the lowest average q-gram set frequency for the NO and NS attribute combinations, where for BLIP linkage quality degrades even at $f_p = 0.2$.

As expected, a further increase in f_p to 0.5 (flipping 50% of the bits in a BF) results in a drastic drop in linkage quality. For BLIP, no correct links are identified at $f_p = 0.5$, a flip probability which ensures the highest level of privacy of $\epsilon_B = 0$, as can be seen in Table 4. As we discussed in Section 4.2, this highlights the limitation of bit flipping for PPRL: a reasonable privacy guarantee cannot be achieved without significantly degrading of linkage quality which compromises the objective of conducting linkage.

For RAPPOR, despite the significant loss in utility for high values of f_p, the privacy loss also remains remarkably high, as Table 4 shows. This again highlights another limitation of bit flipping for PPRL: a reasonable privacy guarantee might not be achieved despite flipping a substantial number of bits in the BF encodings.

6. Discussion and recommendations

In this section, we summarise our main findings and outline the desired characteristics of PPRL techniques, with a particular focus on

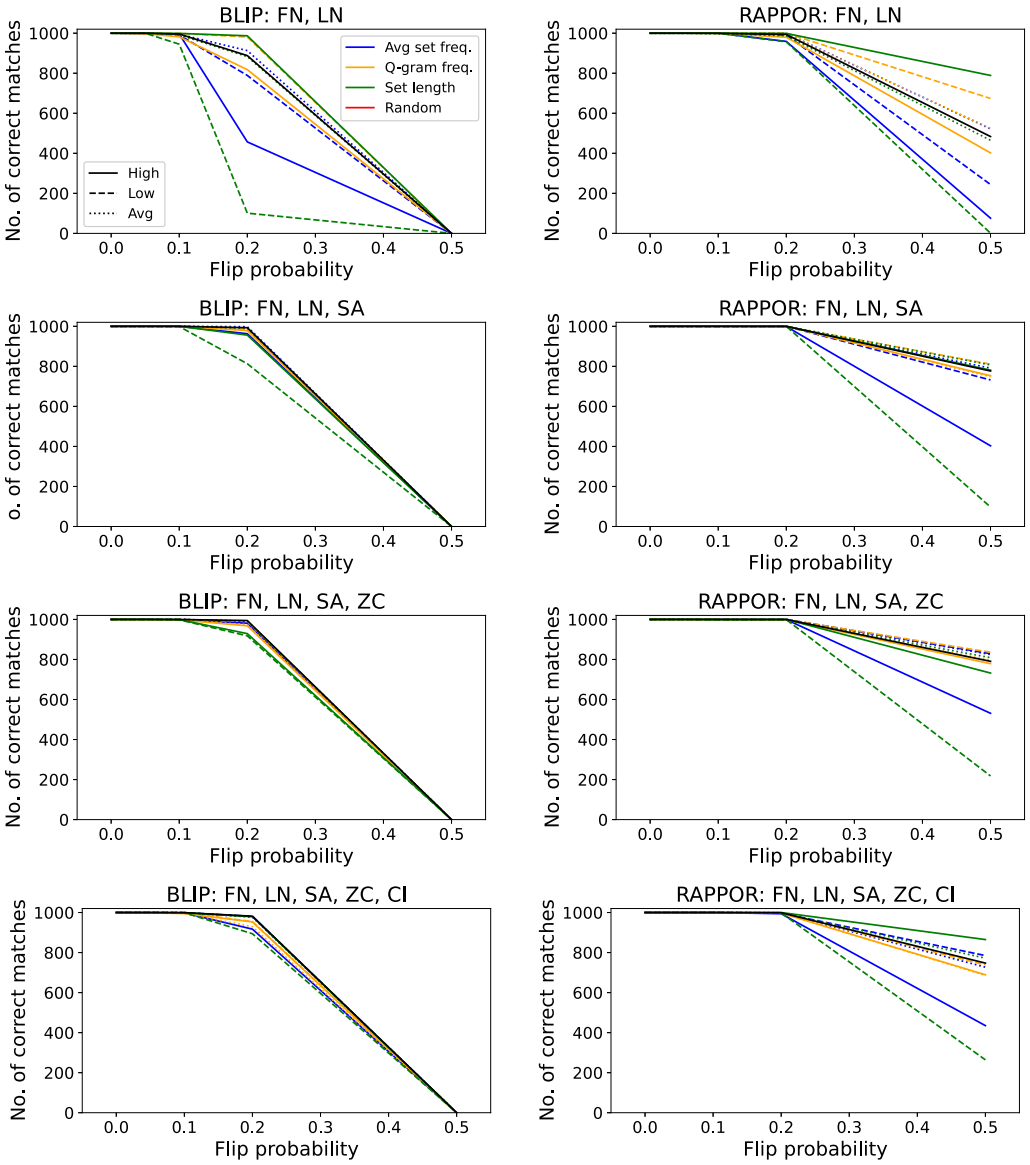

Fig. 3. Summary of 1-1-correct matches identified for varying flip probabilities for BLIP (first column, Eq. (2)) and RAPPOR (second column, Eq. (3)), with different QID combinations as discussed in Section 5.1. The line colours correspond to the different characteristics determining the samples, while the line style indicates the type of extreme considered (high, low, average).

the three dimensions of linkage quality (utility), privacy, and scalability. We then provide a set of recommendations to address the known limitations of PPRL techniques to achieve 'true' PPRL.

Research in PPRL has explored the use of DP due to the lack of formal privacy guarantees in (non-DP) perturbation-based PPRL techniques, such as popular BF encoding [2,10], and the vulnerabilities [11] of these techniques which have been successfully exploited [12]. However, as our review has shown, there are several aspects that demonstrate the fundamental incompatibility of the application of DP in PPRL. To summarise:

1. The objective of record linkage, and thus PPRL, is to accurately identify all true matches across the databases being linked, while not incorrectly linking any non-matching records. In some use cases of linked data, even a single wrong match can lead to major negative outcomes [19,20]. On the other hand, DP aims to enforce formal privacy on the individuals whose data are being processed (or linked, in a PPRL context), to protect against identity disclosure. This is typically achieved through the addition of random noise when data are being published [14].
2. To meet the objective of PPRL means that any form of noise addition, as is done through DP, must not affect the final linkage outcome. Specifically, any noise added must neither lead to the misclassification of true matching record pairs as non-matches, nor to the misclassification of true non-matching record pairs as matches.
3. While certain DP-PPRL blocking techniques enforce privacy without affecting the classification of true matches (by adding faked records to blocks), the privacy guarantees achieved by these techniques hold only during the blocking stage and they do not extend throughout the full PPRL process. Furthermore, the current application of DP for blocking primarily prevents group disclosure, in contrast to the objective of DP which is to prevent identity disclosure.
4. When applied on the encodings generated by a PPRL technique, DP intends to protect these encodings by making them resilient to reidentification attacks [12]. However, in its current application, DP only protects the individual items encoded (such as q-grams) and not the actual entities represented by these

encodings. As such, DP applied on encodings aims to prevent attribute disclosure, while identity disclosure can still be possible because some encodings might retain certain vulnerable characteristics (such as a unique large number of 1-bits in the context of BFs [11]).

As a result, although DP has formal privacy guarantees that can be met through the addition of controlled noise, its current application for PPRL is nonsensical in that it does not protect the individual entities whose records are being linked, and it also likely leads to a reduction in linkage quality. The current application of DP, such as on sensitive block characteristics or bit patterns in BFs, only protects the information exchanged at that particular stage of the linkage process. Furthermore, adding substantial noise (which might be necessary in order to achieve strong privacy, as our experiments in Section 5 have shown) contradicts the objective of performing record linkage, which is to identify all, and only, true matches across the databases being linked.

Therefore, future research in PPRL must focus on developing techniques with improved privacy guarantees without compromising on the utility (linkage quality) of the final linked data set. It is, however, important to note that the trade-off in PPRL is not limited to utility and privacy; it must also account for the efficiency (scalability) of the linkage process for PPRL to be suitable for practical applications, where real-world databases can contain many millions of records.

Fundamentally, linkage quality is dependent on the quality of the data sets being linked, specifically on the accuracy and completeness of the sensitive QID attributes being available to perform linkage [80]. Given that the utility of a linked data set is dependent on linkage quality, PPRL techniques must strive to achieve linkage quality equal to that of performing linkage on plaintext data. The privacy expectations of PPRL are generally governed by legal and regulatory requirements which must be enforced by a specific PPRL technique and protocol employed. In contrast, scalability is constrained by the computational (algorithmic) complexity and resources available, where resources can likely be improved more easily compared to changing privacy laws or trying to obtain data of better quality.

To balance the trade-offs between these three dimensions, there are certain parameters and settings of PPRL techniques that can be adjusted during the blocking and encoding stages (where outputs are generated that are communicated between parties [24]).

1. *Blocking*: Private blocking techniques aim to mask the sizes of blocks, and this is generally achieved by adding faked records or suppressing (removing) actual database records [8,9,16,39,40]. Any addition of faked records results in an increase in the number of records to be encoded and communicated, and the number of encodings to be compared. If actual database records are removed then these need to be compared in an additional comparison step [8,9]. Any such modification of the databases will not result in a reduction of linkage quality, while there is a trade-off between privacy (in the form of reduced risks of group disclosure) and scalability (increased computational requirements).
   Certain existing blocking techniques have parameters that can be tuned to balance this trade-off by controlling the block sizes [2, 81], as a result of which the potential impact of increasing the number of faked records can be limited. As discussed above, the level of faked records to be added will likely also be dictated by the privacy regulations that govern a PPRL project.
2. *Encoding*: The encoding process of QID values in PPRL is typically controlled by a set of parameters for which optimal values are recommended by the specific technique. Such recommendations aim to provide an optimal balance across dimensions, generally focusing on linkage quality and privacy. For example, the number of hash functions for BF-based encoding [10] and the number of reference sets for reference-set based encoding [45] provide a trade-off between linkage quality and privacy (and to a lesser degree scalability). It is therefore recommended to follow such optimal parameter settings, within the context of any privacy regulations that must be fulfilled.

As such, we propose the following recommendations to be considered when implementing PPRL protocols in real-world applications to enhance privacy without compromising on linkage quality:

1. Select a PPRL blocking technique that is not vulnerable to group disclosure while remaining within the desired computational limits of a PPRL system. One way to minimise group disclosure is to employ a blocking technique that is based on encodings [39, 82] rather than the sensitive plaintext QID values themselves. In such a case, block sizes are determined by the frequencies of encodings rather than QID values.
2. Select a PPRL encoding technique that is known to address existing vulnerabilities [11], without relying on post-processing methods (such as hardening [51–53]) that only provide a fix for vulnerable PPRL encoding techniques (often at a cost to linkage quality).
   For example, Ziyad et al. [45] proposed a reference set-based encoding technique that inherently mitigates all known vulnerabilities except the similarity vulnerability. The similarity vulnerability can be exploited by a LU who has access to the encodings and the similarities calculated between pairs of encodings (as is generally required to in PPRL to conduct linkage). Future research should explore alternative protocol designs that minimise the similarity-related information that a LU has access to.

7. Conclusion

In this paper, we have reviewed the application of differential privacy (DP) in the context of privacy-preserving record linkage (PPRL). We highlighted a fundamental contradiction between the current application of DP in PPRL and the objective of PPRL, making it challenging to reconcile the two concepts. Our objective with this work was not to criticise DP or the differentially private mechanisms used, but rather to investigate how the current application of DP in PPRL only enforces privacy on certain pieces of information as exchanged between parties during the PPRL process. We have specifically shown that while DP aims to prevent identity disclosure, its application in the context of PPRL primarily prevents group and attribute disclosure, while (as our experiments have shown) identity disclosure can still occur. We conclude that in its current use in PPRL, DP does not protect the privacy of the entities whose records are being linked.

Future work in PPRL, therefore, should explore alternate methods to effectively employ DP in PPRL in a sensible way, with considerations such as the use of global DP to perturb encodings. Even better, future work should develop methods that integrate strong privacy guarantees directly into the design of PPRL protocols and encoding methods, rather than enforcing them as a subsequent fix to protect certain pieces of vulnerable information.

CRediT authorship contribution statement

Sumayya Ziyad: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Peter Christen:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Rainer Schnell:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization. **Lucas Lange:** Writing – review & editing, Validation, Methodology, Funding acquisition. **Anushka Vidanage:** Writing – review & editing, Validation, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Sumayya Ziyad, Peter Christen, Anushka Vidanage reports financial support was provided by Universities Australia. Rainer Schnell, Lucas Lange reports financial support was provided by German Academic Exchange Service. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially funded by Universities Australia and the German Academic Exchange Service (DAAD) grants 57559735 and 57701258. Sumayya Ziyad gratefully acknowledges the Australian Government Research Training Program.

Appendix A. Selected quotes on the use of differential privacy for record linkage

In the following we show selected texts from publications which discuss the use of differential privacy in the context of privacy preserving record linkage. Note that the reference numbers shown here refer to our list of references rather than the numbers given in the original corresponding publications.

Private record matching using differential privacy (2010) [8]: "The realization of the new model (*referring to the differentially private blocking technique proposed*) presents difficult challenges, such as the evaluation of distance-based matching conditions with the help of only a statistical queries interface."

Efficient privacy-aware record integration (2013) [40]: "Differential privacy is designed to address the case of statistical databases where users are allowed to ask only aggregate queries".

Composing differential privacy and secure computation: A case study on scaling private record linkage (2017) [16]: "Differential privacy has arisen as a gold standard for privacy in situations where it is ok to reveal statistical properties of datasets but not reveal properties of individuals". ... "This is because of a fundamental disconnect between the privacy guarantees in the two stages of these algorithms *(referring to record linkage)*. DP does not allow learning any fact about the input datasets with certainty, while IND-S2PC *(referring to **S**ecure **2**-**P**arty **C**omputation)* (and PRL protocols that satisfy this definition) like secure multi-party computation-based techniques can reveal the output of the function f truthfully. On the other hand, while DP can reveal aggregate properties of the input datasets with low error, protocols that satisfy IND-S2PC are not allowed to leak any information beyond the output of f. Hence, DP and IND-S2PC do not naturally compose". ... "However, there is a fundamental incompatibility between the two definitions. IND-S2PC reveals the output of a function truthfully; whereas, nothing truthful can be revealed under differential privacy. On the other hand, DP reveals noisy yet accurate (to within an approximation factor) aggregate statistics about all the records in the dataset; but, nothing other than the output of a pre-specified function can be revealed under IND-S2PC". ... "It is clear that the privacy guarantees given by DP and IND-S2PC are different. To ensure scalable record linkage with formal privacy guarantees, we need the best of both worlds: the ability to reveal records that appear in the match truthfully, the ability to reveal statistics about non-matching records, and yet not reveal the presence or absence of individual non-matching records in the dataset".

Hybrid framework of differential privacy and secure multi-party computation for privacy-preserving entity resolution (2025) [65]:
"While differential privacy provides provable guarantees at the cost of accuracy degradation (Dwork and Roth, 2014 [26]; Lindell and Pinkas, 2009 [5]), secure multi-party computation ensures confidentiality but suffers computational inefficiency".

A multi-party privacy-preserving record linkage method based on secondary encoding (2024) [83]: "Schnell and Borgs [13] also proposed a scheme involving random bit flipping, where bit values are flipped at certain positions in the Bloom Filter according to differential privacy mechanisms to enhance security. However, due to the noise mechanism of differential privacy, data quality is compromised, leading to a decrease in linkage quality".

Modern privacy-preserving record linkage techniques: An overview (2021) [3]: "Differential privacy has been used in PPRL to perturb data by adding noise such that every individual in the dataset is indistinguishable. However, noise addition incurs significant utility loss and volume increase".

Private record matching using differential privacy (2010) [8]: "When the input datasets T and V are large, even after considerable amount of reduction in comparison space, costs of applying our solutions *(referring to their use of secure multi-party computation-based operations)* might still be higher than the amount anticipated by the participants". ... "Devising a differentially private blocking step is a challenging task, due to the following reasons:

- The matching operation requires evaluations of distance based conditions, whereas differential privacy allows statistical queries only.
- The amount of noise added by the statistical database to protect data is dependent on the sensitivity characteristic of the answered queries. The matching protocols must be carefully designed such that the sensitivity of the combination of queries that are answered is kept low.
- Existing data-partitioning index structures, which are necessary to improve blocking effectiveness, disclose a lot of information about data distribution, and are not compliant with differential privacy. Specialized variations of such structures must be carefully designed to ensure privacy".

A hybrid private record linkage scheme: Separating differentially private synopses from matching records (2015) [9]: "At the end of the private linkage *(referring to the protocol proposed by Inan et al. [72])*, each party obtains both the matching records and the differentially private sizes of the subsets. When these two sources of information are combined together, differential privacy for the non-matching records does not hold any more".

Hybrid private record linkage: Separating differentially private synopses from matching records (2019) [62]: "The two-party record linkage protocol proposed by He et al. (2017) [16] suffers from the disadvantage that each data owner learns the noisy number of non-matching records in each partition of the other party after the linkage process, which may be undesirable".

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.is.2026.102726.

Data availability

Research Link Provided

Code and appendix with additional experiments (Original data) (GitHub)

References

[1] P. Christen, Data Matching – Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection, Springer, Heidelberg, 2012.

[2] P. Christen, T. Ranbaduge, R. Schnell, Linking Sensitive Data, Springer, Heidelberg, 2020.

[3] A. Gkoulalas-Divanis, D. Vatsalan, D. Karapiperis, M. Kantarcioglu, Modern privacy-preserving record linkage techniques: An overview, Trans. Inf. Forensics Secur. 16 (2021) 4966–4987.

[4] J.H. Boyd, S.M. Randall, A.M. Ferrante, Application of privacy-preserving techniques in operational record linkage centres, in: Medical Data Privacy Handbook, Springer, Cham, 2015, pp. 267–287.

[5] Y. Lindell, B. Pinkas, Secure multiparty computation for privacy-preserving data mining, J. Priv. Confidentiality 1 (1) (2009) 5.

[6] D. Vatsalan, Z. Sehili, P. Christen, E. Rahm, Privacy-preserving record linkage for Big Data: Current approaches and research challenges, in: Handbook of Big Data Technologies, Springer, 2017, pp. 851–895.

[7] I. Lazrig, T. Ong, I. Ray, I. Ray, X. Jiang, J. Vaidya, Privacy preserving probabilistic record linkage without trusted third party, in: Privacy, Security and Trust, IEEE, Belfast, 2018, pp. 1–10.

[8] A. Inan, M. Kantarcioglu, G. Ghinita, E. Bertino, Private record matching using differential privacy, in: Conference on Extending Database Technology, ACM, Lausanne, 2010, pp. 123–134.

[9] J. Cao, F.-Y. Rao, E. Bertino, M. Kantarcioglu, A hybrid private record linkage scheme: Separating differentially private synopses from matching records, in: International Conference on Data Engineering, IEEE, Seoul, 2015, pp. 1011–1022.

[10] R. Schnell, T. Bachteler, J. Reiher, Privacy-preserving record linkage using Bloom filters, Med. Informatics Decis. Mak. 9 (1) (2009).

[11] A. Vidanage, P. Christen, T. Ranbaduge, R. Schnell, A vulnerability assessment framework for privacy-preserving record linkage, Trans. Priv. Secur. 26 (3) (2023) 1–31.

[12] A. Vidanage, T. Ranbaduge, P. Christen, R. Schnell, Taxonomy of attacks on privacy-preserving record linkage, J. Priv. Confidentiality 12 (1) (2022).

[13] R. Schnell, C. Borgs, Randomized response and balanced Bloom filters for privacy preserving record linkage, in: Workshop on Data Integration and Applications, IEEE, Barcelona, 2016, pp. 218–224.

[14] C. Dwork, F. McSherry, K. Nissim, A. Smith, Calibrating noise to sensitivity in private data analysis, in: Theory of Cryptography, Springer, New York, 2006, pp. 265–284.

[15] S. Matwin, J. Nin, M. Sehatkar, T. Szapiro, A review of attribute disclosure control, in: G. Navarro-Arribas, V. Torra (Eds.), Advanced Research in Data Privacy, Springer, 2015, pp. 41–61.

[16] X. He, A. Machanavajjhala, C. Flynn, D. Srivastava, Composing differential privacy and secure computation: A case study on scaling private record linkage, in: Conference on Computer and Communications Security, ACM, Dallas, 2017, pp. 1389–1406.

[17] D.L. Oberski, F. Kreuter, Differential Privacy and Social Science: An Urgent Puzzle, Harv. Data Sci. Rev. 2 (1) (2020).

[18] T.S. Kuhn, The Structure of Scientific Revolutions, third ed., University of Chicago Press, Chicago, 1996.

[19] G. Kamat, Mismatches and missed matches: Why mislinked data can mislead public health studies, Significance 23 (2026) 14–16.

[20] S. Tahamont, Z. Jelveh, A. Chalfin, S. Yan, B. Hansen, Dude, where's my treatment effect? Errors in administrative data linking and the destruction of statistical power in randomized experiments, J. Quant. Criminol. 37 (3) (2021) 715–749.

[21] J. Lane, Towards an approach for evaluating the impact of AI standards the use case of entity resolution (2506.13839), 2025, arXiv Preprint.

[22] M.D. Larsen, Record linkage, nondisclosure, counterterrorism, and statistics, in: Survey Methods Section, Canadian Statistical Society, London, 2006.

[23] Ú. Erlingsson, V. Pihur, A. Korolova, RAPPOR: Randomized aggregatable privacy-preserving ordinal response, in: Security, Audit and Control, ACM, Scottsdale, Arizona, 2014, pp. 1054–1067.

[24] P. Christen, R. Schnell, A. Vidanage, Information leakage in the practical linking of sensitive data: Parties, protocols, and adversaries, Inf. Syst. (2026).

[25] K. Nissim, T. Steinke, A. Wood, M. Altman, A. Bembenek, M. Bun, M. Gaboardi, D.R. O'Brien, S. Vadhan, Differential privacy: A primer for a non-technical audience, in: Privacy Law Scholars Conference, Berkeley, 2017.

[26] C. Dwork, A. Roth, et al., The algorithmic foundations of differential privacy, Found. Trends Theor. Comput. Sci. 9 (3–4) (2014) 211–407.

[27] R. Tjhin, M.S. Akbar, C. Canonne, R. Bashir, Analysis, design, and implementation of a user-friendly differential privacy application, Sensors (ISSN: 1424-8220) 25 (5) (2025).

[28] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, A. Thakurta, Amplification by shuffling: from local to central differential privacy via anonymity, in: ACM-SIAM Symposium on Discrete Algorithms, San Diego, 2019, pp. 2468–2479.

[29] M. Alaggan, S. Gambs, A. Kermarrec, BLIP: Non-interactive differentially-private similarity computation on bloom filters, in: Stabilization, Safety, and Security of Distributed Systems, Springer, Toronto, 2012, pp. 202–216.

[30] F. Naumann, M. Herschel, An introduction to duplicate detection, in: Synthesis Lectures on Data Management, Morgan and Claypool Publishers, California, 2010.

[31] T.N. Herzog, F. Scheuren, W.E. Winkler, Data Quality and Record Linkage Techniques, Springer, New York, 2007.

[32] H.L. Dunn, Record linkage, Am. J. Public Health 36 (12) (1946) 1412.

[33] I.P. Fellegi, A.B. Sunter, A theory for record linkage, J. Amer. Statist. Assoc. 64 (328) (1969) 1183–1210.

[34] O. Binette, R.C. Steorts, (Almost) all of entity resolution, Sci. Adv. 8 (12) (2022) eabi8021.

[35] P. Christen, A survey of indexing techniques for scalable record linkage and deduplication, Trans. Knowl. Data Eng. 24 (9) (2012) 1537–1555.

[36] G. Papadakis, J. Svirsky, A. Gal, T. Palpanas, Comparative analysis of approximate blocking techniques for entity resolution, VLDB Endow. 9 (9) (2016) 684–695.

[37] P. Indyk, R. Motwani, Approximate nearest neighbors: towards removing the curse of dimensionality, in: Symposium on Theory of Computing, ACM, Dallas, 1998, pp. 604–613.

[38] C. Nanayakkara, P. Christen, Locality sensitive hashing with temporal and spatial constraints for efficient population record linkage, in: Conference on Information & Knowledge Management, ACM, Atlanta, 2022, pp. 4354–4358.

[39] N. Wu, D. Vatsalan, S. Verma, M.A. Kaafar, Fairness and Cost Constrained Privacy-Aware Record Linkage, IEEE Trans. Inf. Forensics Secur. 17 (2022) 2644–2656.

[40] M. Kuzu, M. Kantarcioglu, A. Inan, E. Bertino, E. Durham, B. Malin, Efficient privacy-aware record integration, in: Conference on Extending Database Technology, ACM, Genoa, 2013, pp. 167–178.

[41] S.M. Randall, A.P. Brown, A.M. Ferrante, J.H. Boyd, Privacy preserving linkage using multiple dynamic match keys, Int. J. Popul. Data Sci. 4 (1) (2019).

[42] M. Kantarcioglu, W. Howe, B. Liu, V. Petkov, E. Casas-Silva, D. Velasquez-Kolnik, B.A. Malin, L. Penberthy, A novel analysis methodology for assessment of re-identification risks for the national cancer institute cancer registry privacy preserving record linkage technique, J. Am. Med. Informat. Assoc. (2025) ocaf172.

[43] D. Smith, Secure pseudonymisation for privacy-preserving probabilistic record linkage, J. Inf. Secur. Appl. 34 (2017) 271–279.

[44] T. Ranbaduge, P. Christen, R. Schnell, Secure and accurate two-step hash encoding for privacy-preserving record linkage, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Singapore, 2020, pp. 139–151.

[45] S. Ziyad, P. Christen, A. Vidanage, C. Nanayakkara, R. Schnell, Privacy-preserving record linkage using reference set based encoding: A single parameter method, Inf. Syst. 113 (2025) 102569.

[46] N. Barlaug, J.A. Gulla, Neural networks for entity matching: A survey, Trans. Knowl. Discov. from Data 15 (3) (2021).

[47] P. Konda, S. Das, P. Suganthan G.C., A. Doan, et al., Magellan: Toward building entity matching management systems, VLDB Endow. 9 (12) (2016) 1197–1208.

[48] T. Ranbaduge, D. Vatsalan, M. Ding, Privacy-preserving deep learning based record linkage, Trans. Knowl. Data Eng. 36 (11) (2023) 6839–6850.

[49] D. Vatsalan, P. Christen, V.S. Verykios, A taxonomy of privacy-preserving record linkage techniques, Inf. Syst. 38 (6) (2013) 946–969.

[50] A. Vidanage, T. Ranbaduge, P. Christen, S. Randall, A privacy attack on multiple dynamic match-key based privacy-preserving record linkage, Int. J. Popul. Data Sci. 5 (1) (2020).

[51] T. Ranbaduge, R. Schnell, Securing bloom filters for privacy-preserving record linkage, in: Conference on Information and Knowledge Management, ACM, Galway, 2020, pp. 2185–2188.

[52] M. Franke, Z. Sehili, F. Rohde, E. Rahm, Evaluation of hardening techniques for privacy-preserving record linkage., in: EDBT, 2021, pp. 289–300.

[53] S. Ziyad, P. Christen, A. Vidanage, C. Nanayakkara, R. Schnell, Vulnerability-aware hardening for secure privacy-preserving record linkage, in: Conference on Information and Knowledge Management, ACM, Seoul, 2025, pp. 4582—-4591.

[54] D. Vatsalan, P. Christen, Privacy-preserving matching of similar patients, J. Biomed. Informat. 59 (2016) 285–298.

[55] D. Karapiperis, A. Gkoulalas-Divanis, V.S. Verykios, FEDERAL: A framework for distance-aware privacy-preserving record linkage, Trans. Knowl. Data Eng. 30 (2) (2017) 292–304.

[56] L. Sun, L. Zhang, X. Ye, Randomized bit vector: Privacy-preserving encoding mechanism, in: Conference on Information and Knowledge Management, ACM, Turin, 2018, pp. 1263–1272.

[57] R. Schnell, J. Klingwort, J.M. Farrow, Locational privacy-preserving distance computations with intersecting sets of randomly labeled grid points, Int. J. Health Geogr. 20 (2021) 14.

[58] R. Schnell, C. Borgs, Encoding hierarchical classification codes for privacy-preserving record linkage using bloom filters, in: Workshop on Data Integration and Applications, Held At ECML/PKDD, Springer, Würzburg, 2019, pp. 142–156.

[59] G. Papadakis, D. Skoutas, E. Thanos, T. Palpanas, Blocking and filtering techniques for entity resolution: A survey, Comput. Surv. 53 (2) (2020) 1–42.

[60] T. Ranbaduge, D. Vatsalan, P. Christen, Scalable block scheduling for efficient multi-database record linkage, in: International Conference on Data Mining, IEEE, Barcelona, 2016, pp. 1161–1166.
[61] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: Theory and Application of Cryptographic Techniques, Springer, Prague, 1999, pp. 223–238.
[62] F.-Y. Rao, J. Cao, E. Bertino, M. Kantarcioglu, Hybrid private record linkage: Separating differentially private synopses from matching records, Trans. Priv. Secur. 22 (3) (2019) 1–36.
[63] L. Bonomi, L. Xiong, R. Chen, B.C. Fung, Frequent grams based embedding for privacy preserving record linkage, in: Conference on Information and Knowledge Management, ACM, Maui, 2012, pp. 1597–1601.
[64] S. Vaiwsri, T. Ranbaduge, P. Christen, Reference values based hardening for Bloom filters based privacy-preserving record linkage, in: Australasian Data Mining Conference, Springer, Bathurst, 2018, pp. 189–202.
[65] M. Dorgbefu, Y.M. Missah, N. Ussiph, G. Abdul-Salaam, O. Kornyo, J.M. Mensah, Hybrid framework of differential privacy and secure multi-party computation for privacy-preserving entity resolution, Comput. Secur. 157 (2025) 104603.
[66] S.M. Randall, A.P. Brown, A.M. Ferrante, J.H. Boyd, S. Robinson, Implementing privacy preserving record linkage: Insights from Australian use cases, Int. J. Med. Informatics 191 (2024).
[67] M. Kuzu, M. Kantarcioglu, E. Durham, B. Malin, A constraint satisfaction cryptanalysis of Bloom filters in private record linkage, in: Privacy Enhancing Technologies Symposium, Waterloo, Canada, 2011, pp. 226–245.
[68] F. Niedermeyer, S. Steinmetzer, M. Kroll, R. Schnell, Cryptanalysis of basic Bloom filters used for privacy preserving record linkage, J. Priv. Confidentiality 6 (2) (2014) 59–79.
[69] S.L. Warner, Randomized response: A survey technique for eliminating evasive answer bias, J. Amer. Statist. Assoc. 60 (309) (1965) 63–69.
[70] N. Wu, D. Vatsalan, M. Kaafar, S. Ramesh, Privacy-preserving record linkage for cardinality counting, in: Asia Conference on Computer and Communications Security, ACM, Melbourne, 2023, pp. 53–64.
[71] D. Morales, I. Agudo, J. Lopez, Private set intersection: A systematic literature review, Comput. Sci. Rev. (ISSN: 1574-0137) 49 (2023) 100567.
[72] A. Inan, M. Kantarcioglu, E. Bertino, M. Scannapieco, A hybrid approach to private record linkage, in: International Conference on Data Engineering, IEEE, Cancun, 2008, pp. 496–505.
[73] A. Karakasidis, G. Koloniari, V.S. Verykios, Scalable blocking for privacy preserving record linkage, in: Conference on Knowledge Discovery and Data Mining, ACM, Sydney, 2015, pp. 527–536.
[74] R. Schnell, D. Rukasz, PPRL: Privacy preserving record linkage, 2022, http://dx.doi.org/10.32614/CRAN.package.PPRL, R package version 0.3.8.
[75] S. Patel, R. Dewri, Private record linkage with linkage maps, Secur. Priv. 5 (6) (2022) e265.
[76] P. Christen, Preparation of a real temporal voter data set for record linkage and duplicate detection research, Tech. rep., Australian National University, 2014.
[77] A. Vidanage, T. Ranbaduge, P. Christen, R. Schnell, Efficient pattern mining based cryptanalysis for privacy-preserving record linkage, in: IEEE International Conference on Data Engineering, Macau, 2019, pp. 1698–1701.
[78] P. Christen, R. Schnell, D. Vatsalan, T. Ranbaduge, Efficient cryptanalysis of Bloom filters for privacy-preserving record linkage, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Jeju, Korea, 2017, pp. 628–640.
[79] A. Vidanage, P. Christen, T. Ranbaduge, R. Schnell, A graph matching attack on privacy-preserving record linkage, in: ACM Conference on Information and Knowledge Management, Galway, 2020, pp. 1485–1494.
[80] P. Christen, R. Schnell, Thirty-three myths and misconceptions about population data: from data capture and processing to linkage, Int. J. Popul. Data Sci. 8 (1) (2023).
[81] J. Fisher, P. Christen, Q. Wang, E. Rahm, A clustering-based framework to control block sizes for entity resolution, in: ACM Conference on Knowledge Discovery and Data Mining, Sydney, 2015, pp. 279–288.
[82] T. Ranbaduge, D. Vatsalan, P. Christen, V.S. Verykios, Hashing-based distributed multi-party blocking for privacy-preserving record linkage, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Auckland, 2016, pp. 415–427.
[83] S. Han, Y. Wang, D. Shen, C. Wang, A multi-party privacy-preserving record linkage method based on secondary encoding, Mathematics 12 (12) (2024) 1800.