Privacy-Preserving Record Linkage for Cardinality Counting

Nan Wu
Macquarie University, CSIRO's Data61
Sydney, Australia
nan.wu5@hdr.mq.edu.au

Dinusha Vatsalan
Macquarie University
Sydney, Australia
dinusha.vatsalan@mq.edu.au

Mohamed Ali Kaafar
Macquarie University
Sydney, Australia
dali.kaafar@mq.edu.au

Sanath Kumar Ramesh
OpenTreatments Foundation, CuresDev LLC
Seattle, San Jose, United States
sanath@gpx4.org

ABSTRACT

Several applications require counting the number of distinct items in the data, which is known as the cardinality counting problem. Example applications include health applications such as rare disease patients counting for adequate awareness and funding, and counting the number of cases of a new disease for outbreak detection, marketing applications such as counting the visibility reached for a new product, and cybersecurity applications such as tracking the number of unique views of social media posts. The data needed for the counting is however often personal and sensitive, and need to be processed using privacy-preserving techniques. The quality of data in different databases, for example typos, errors and variations, poses additional challenges for accurate cardinality estimation. While privacy-preserving cardinality counting has gained much attention in the recent times and a few privacy-preserving algorithms have been developed for cardinality estimation, no work has so far been done on privacy-preserving cardinality counting using record linkage techniques with fuzzy matching and provable privacy guarantees. We propose a novel privacy-preserving record linkage algorithm using unsupervised clustering techniques to link and count the cardinality of individuals in multiple datasets without compromising their privacy or identity. In addition, existing Elbow methods to find the optimal number of clusters as the cardinality are far from accurate as they do not take into account the purity and completeness of generated clusters. We propose a novel method to find the optimal number of clusters in unsupervised learning. Our experimental results on real and synthetic datasets are highly promising in terms of significantly smaller error rate of less than 0.1 with a privacy budget $\epsilon = 1.0$ compared to the state-of-the-art fuzzy matching and clustering method.

CCS CONCEPTS

• **Security and privacy** → **Data anonymization and sanitization**; **Management and querying of encrypted data**; **Privacy protections**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Probabilistic counting, distinct-counting, fuzzy matching, Bloom filters, unsupervised learning, differential privacy

ACM Reference Format:
Nan Wu, Dinusha Vatsalan, Mohamed Ali Kaafar, and Sanath Kumar Ramesh. 2023. Privacy-Preserving Record Linkage for Cardinality Counting. In *ACM ASIA Conference on Computer and Communications Security (ASIA CCS '23), July 10–14, 2023, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3579856.3590338

1 INTRODUCTION

The cardinality counting problem has become of tremendous interest in many different applications to enable a variety of analytics of dispersed data. However, the privacy concerns of sharing or revealing individuals' data containing personal information for analytic purposes require privacy-preserving processing of counting. In most cases, the records of the same individual in different databases do not contain a unique identifier, and the quasi identifiers in the records, such as names, addresses, and ages, are often prone to data errors, inconsistencies and variations. Accurately estimating the cardinality of individuals or items represented by records in multiple different databases without compromising the privacy of the individuals is hence a challenging research problem.

Gaining insight into the number of unique records from multiple data sources is crucial in many applications. A promising real-world application is rare disease patients counting. Rare diseases in general do not receive sufficient funding for treatment [38] and this disparity is created in part because funders measure the impact of their investments based on the size of patient population affected by a given disease. Unfortunately, for a majority of rare diseases, this data is at best a wild guess and at worst non-existent. Another example in the health domain is disease outbreak detection where the number of unique cases of a new disease needs to be continuously monitored from multiple different hospitals and clinics to predict the likelihood of an outbreak and to make preventive measures.

Similarly, national security or cybersecurity applications monitor the number of views of videos or posts in online or social media in order to predict the potential threats of any video/post that becomes

ASIA CCS '23, July 10–14, 2023, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0098-9/23/07…$15.00
https://doi.org/10.1145/3579856.3590338

viral within a short time period (for example, fake news with phishing links [40]) and make any timely decision. Online businesses need to monitor the number of unique views by customers of a new product in order to make decisions on the marketing strategies to manage the marketing costs. Web search log analysis may require calculating the number of distinct queries in a list of queries from many users (e.g., the number of distinct queries made to a search engine over a week) to improve the performance of the search engine in terms of estimating the selectivity of queries and designing good strategies for executing a query [3, 31, 54]. Social game industry and e-commerce applications use count distinct metrics, such as the daily active users (DAU) and monthly active users (MAU) metrics [51], to estimate the workload for those online applications. In all these example applications, the data needed to derive such insights is personal and sensitive, and must be processed private.

While there have been several methods proposed for the cardinality counting problem in general [3, 6, 18, 21, 25, 30], privacy aspects of cardinality counting have only recently received attention in the research literature. Some recent works developed privacy-preserving algorithms for cardinality counting using different probabilistic data structures (KMV, FM-Sketch, or HyperLogLog) [41, 42, 50]. A recent study has shown that probabilistic cardinality estimators like HyperLogLog do not preserve privacy as achieving accurate and private cardinality estimation is impossible, and therefore they can be sensitive as raw data [12]. In addition, they are not robust or tolerant to errors and variations in data. Privacy-preserving record linkage is hence required to link or deduplicate records corresponding to the same individual based on fuzzy matching of personal identifying information (PII) contained in the quasi-identifiers (e.g. names and addresses) to count the cardinality of individuals.

In this work, we propose a novel privacy-preserving record linkage algorithm for linking and counting unique individuals or items from multiple databases using a combination of Bloom filter encoding, local differential privacy, and machine learning techniques. Specifically, the database owners locally encode and perturb the PII in their records using Bloom filters and local differential privacy. The encoded and perturbed records from all the databases are then input to a clustering algorithm that aims to link and group records corresponding to the same individual/item into the same cluster and different individuals/item into different clusters.

The optimal number of clusters is then computed to calculate the cardinality of records. Since ground-truth data is not available in real applications and is not trivial to manually label data due to privacy and confidentiality concerns, finding the optimal number of clusters for such unsupervised machine learning tasks is highly challenging. Existing Elbow methods based on metrics like silhouette coefficient and Calinski-Harabasz score measure the inter and intra cluster distances to find the optimal number of clusters [5, 13, 36, 52]. However, they are not accurate and optimal, especially for linking or deduplicating records, and thereby counting the correct cardinality. The main limitation is that they do not account for the purity and completeness of clusters which are necessary for accurate record linkage. Hence, we calculate the optimal number of clusters or cardinality by proposing a novel method to measure the purity and completeness of generated clusters. While we propose an algorithm for the distinct-counting problem, our proposed method for finding the optimal number of clusters can be used with any unsupervised clustering techniques that do not have labelled data for fine-tuning and/or evaluation.

The main contributions of this paper are:

(1) We study the problem of privacy-preserving cardinality counting of individuals or items from multiple different databases in the presence of data errors and variations.
(2) We introduce a novel privacy-preserving record linkage algorithm for cardinality counting with provable privacy guarantees. Our algorithm uses Bloom filter encoding and local differential privacy for data encoding and unsupervised clustering on the encoded data to estimate the cardinality.
(3) We propose a novel clustering algorithm to find the optimal number of clusters in the absence of labelled data to predict the accurate cardinality. We develop two variations of our clustering method and evaluate their accuracies for cardinality estimation.
(4) We provide formal proof of privacy guarantees of our proposed method.
(5) We conduct experimental evaluation on real and synthetic North Carolina voter registration (NCVR) datasets to validate the accuracy of our method. Since existing cardinality estimators do not allow fuzzy matching for cardinality counting, we compare only with a state-of-the-art baseline method for fuzzy matching and clustering [36] to provide a fair comparison. The experimental results show that our methods can achieve a very small error rate closer to 0.0 with a small privacy budget of $\epsilon = 1.0$ or $\epsilon = 2.0$ even on highly corrupted datasets, and significantly outperform the existing methods.

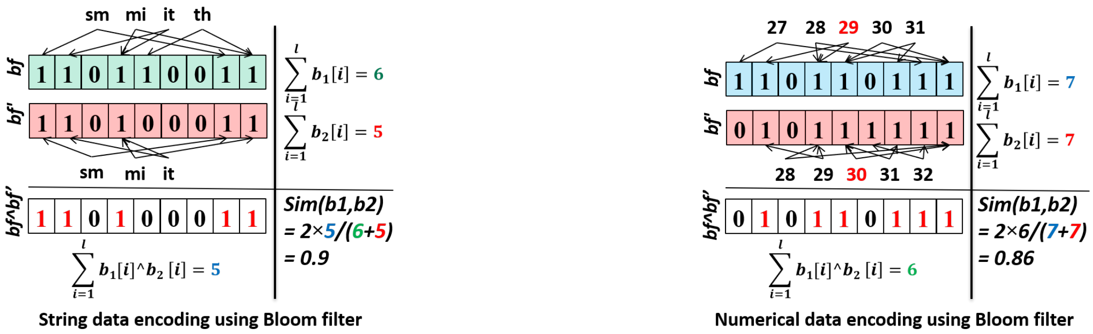
Outline: We provide preliminaries in the following section and describe our methodology in Section 3. In Section 4 we present the results of our experimental study and in Section 5 we review the literature of privacy-preserving counting techniques. Finally we summarise, discuss limitations, and provide directions to future research in Section 6.

2 BACKGROUND

In this section, we describe the preliminaries of this work. We first provide preliminaries for the Bloom filter encoding in Section 2.1. We then describe the system architecture and threat model of the research problem we address in this paper in Section 2.2, and finally we describe differential privacy in Section 2.3.

2.1 Bloom filter encoding

Bloom filters are probabilistic data structures that are highly efficient for storing, processing, and computation. Essentially, Bloom filters are bit vectors that initially contain 0 in all the bit positions. k independent hash functions $h_i(\cdot)$ (with $1 \leq i \leq k$) are used to hash-map an element x by setting the corresponding bit positions in the Bloom filter b to 1 (i.e. $\forall_i\ b[h_i(x)] = 1$). A Bloom filter allows a tunable false positive rate fpr so that a query returns either "definitely not" (with no error), or "probably yes" (with probability fpr of being wrong). The lower fpr is, the better utility is as there will be a smaller number of false matches identified by matching the Bloom filter encoded data, but the more space the filter requires. The false positive probability for encoding n elements into a Bloom



String data encoding using Bloom filter

Numerical data encoding using Bloom filter

Figure 1: Bloom filter encoding of string values (left) and numerical values (right) [39, 45], and fuzzy matching using Dice-coefficient similarity function, as described in Section 2.1.

filter of length ℓ bits using k hash functions is $fpr = (1 - e^{-kn/\ell})^k$, which is controllable by tuning the parameters k and ℓ.

The main feature of Bloom filter encoding that makes it applicable to efficient fuzzy matching of encoded records is that it preserves the similarity/distance between records in the Bloom filter space (with a negligible utility loss) [39, 45]. For example, with string values the q-grams (sub-strings of length q) of string values can be hash-mapped into the Bloom filter bf using k independent hash functions [39], while for numerical values, the neighbouring values (within a certain interval to allow fuzzy matching) of values can be hash-mapped into the Bloom filter [45]. Fig. 1 illustrates an example of fuzzy matching of string and numerical values using Bloom filters [39, 45].

The matching of Bloom filters can be determined by calculating the similarity value using a token-based similarity function, such as Jaccard, Dice, or Hamming [48]. For example, Dice-coefficient similarity metric is calculated for the example pairs of Bloom filter encoded strings and integers in Fig. 1 as $2 \times \frac{\sum(bf_1 \cap bf_2)}{\sum(bf_1)+\sum(bf_2)}$, where bf_1 and bf_2 are the two Bloom filters. Collision of different elements being mapped to the same bit position can occur during the hash-mapping (depending on the parameter setting), resulting in false positives with matching Bloom filter encoded records. However, with appropriate parameter settings, Bloom filters have shown to be successful in providing high matching results while being highly efficient [35, 39, 45].

The computation complexity of encoding $N = |D|$ records, where D contains all the records and $|\cdot|$ denotes the size of a given set, with an average of n elements in each record (e.g. q-grams) using k hash functions into l-length Bloom filters is $O(n \cdot k \cdot N)$, the computation complexity of Bloom filter encoded data matching is $O(N \cdot (N-1))$ and the communication complexity is $O(N \cdot l)$. A variety of indexing/blocking techniques has been proposed in the literature to reduce the quadratic computation complexity of linkage and thereby improve the scalability of linkage [49]. The focus of this work is not on the scalability aspect, but on the utility aspect of linkage in the absence of labelled data.

2.2 System architecture and threat model

The system architecture of our proposed method for privacy-preserving cardinality counting is illustrated in Fig. 2. At each data owner side, the PII in records are encoded into Bloom filters first and then perturbed using local differential Privacy. The encoded and perturbed records from multiple different data owners are sent to a linkage unit that applies our proposed clustering algorithm on the Bloom filters such that similar Bloom filters corresponding to the same individual/patient are grouped into one cluster. The optimal number of clusters is estimated as the cardinality of records from multiple databases and reported.

Only the Bloom filters are shared with the linkage unit. Bloom filter encoding does provide some inherent privacy guarantees due to the collision of different elements being hash-mapped to same bits in the Bloom filters, providing uncertainty in decoding. However, as shown in the recent research, Bloom filters can be vulnerable to cryptanalysis attacks that map bits to q-grams or elements based on the frequency of bits or bit patterns [9, 10]. As another layer of privacy to provide provable privacy guarantees, we combine Bloom filter encoding with local differential privacy. Differential privacy noise is added to the Bloom filters using the randomized response technique, as will be described in detail in the following sub-section, in order to make the bits in the Bloom filters differentially private so that bits cannot be distinguished based on their presence or frequency information. Further, the linkage unit is assumed to follow the steps of the system without deviating or modifying the Bloom filters (i.e. integrity of Bloom filters is preserved).

2.3 Differential Privacy

Differential privacy [14–16] guarantees for each individual in a dataset that the probability of any information that could be discovered about an individual with their data in the dataset is similar to the probability that could be discovered without their data in the dataset. That is, the output of any query f performed on dataset x will be indistinguishable from the output of the same query f performed on dataset y, where y differs from x by at most one record (the record of any individual). Moreover, it promises that any supplementary data an adversary might have about the individual is irrelevant; the adversary is unable to identify any additional information about an individual from the data regardless of the auxiliary knowledge about the individual with a high probability.

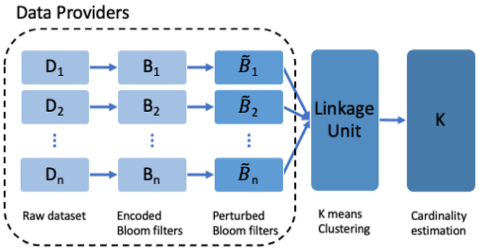


Figure 2: An outline of our system model for privacy-preserving cardinality estimation.

Definition 2.1 (Differential Privacy [14]). A randomized function $\mathcal{A}$ (i.e. a function with a randomized component) is ϵ-differentially private if for all measurable subset of outputs $y \subseteq Range(\mathcal{A})$ and for all data $x, x' \in \mathcal{D}^n$ such that $||x - x'||_1 \leq 1$:

$$Pr(\mathcal{A}(x) \in y) \leq e^{\epsilon} \times Pr(\mathcal{A}(x') \in y). \tag{1}$$

Local differential Privacy (LDP) is a differential privacy model developed specifically to provide guarantees such that even if an adversary has access to the personal responses of an individual in the dataset, the adversary is still unable to learn additional information about the individual from the personal data with high probability [19]. It ensures differential privacy guarantees for each individual's inputs by processing (perturbing) the data locally on-device rather than requiring a trusted data curator/central server [32]. LDP has become the de-facto privacy standard around the world in recent years, with the technology companies Google and Apple implementing LDP in their latest operating systems and applications [1, 17, 26, 34].

Definition 2.2 (Local Differential Privacy [14]). Let $\mathcal{A} : \mathcal{D} \to \mathcal{Y}$ be a randomized algorithm mapping a data entry in $\mathcal{D}$ to $\mathcal{Y}$. The algorithm $\mathcal{A}$ is ϵ-local differentially private if for data entry $x, \neg x \in \mathcal{D}$ and all outputs $y \in \mathcal{Y}$,

$$Pr(\mathcal{A}(x) \in y) \leq e^{\epsilon} \times Pr(\mathcal{A}(\neg x) \in y). \tag{2}$$

A widely used mechanism specifically for designing LDP algorithms is the randomized response technique [53]. The primary idea is that the data owners respond to binary questions (e.g. 0 or 1) in a randomized manner. Since Bloom filters contain binary information (1 or 0), we use randomized response technique to add noise to Bloom filters such that the Bloom filters are differentially private and robust against cryptanalysis attacks that exploit the presence or frequency of bits in the Bloom filters, as will be described in Section 3.1.

2.4 K-means Clustering

k-means clustering is an unsupervised machine learning algorithm that groups the unlabelled dataset into different clusters, such that similar data points or records are grouped into the same cluster and dissimilar records are grouped into different clusters. It is an iterative algorithm where the clusters are refined over the iterations by minimizing the sum of distances between the data points and their corresponding clusters.

Algorithm 1 *Privacy-preserving Bloom filter encoding with ϵ-local differential Privacy*

```
Inputs:
    Raw dataset from one data provider: $D$,
    Privacy budget: $\epsilon$
Outputs:
    Perturbed Bloom filters: $B'$
Initialize:
    $B' \leftarrow \Phi$
for $i = 1, \ldots, n_{|D|}$ do                         ▷ Do for each record in raw dataset
    $bf_i = \text{encode}(\text{record}_i), \text{record}_i \in D$
    for $j = 1, \ldots, \ell$ do                        ▷ For each bit in the Bloom filter
        $bf'_i \leftarrow \emptyset$
        $\eta = \frac{1}{1+e^{\epsilon}}$
        $p = random[0, 1]$
        if $p \leq \eta$ then                           ▷ flip $b_j$ with probability $\eta$
            $b'_j = b_j$
        else
            $b'_j = 1 - b_j$
        end if
        $bf'_i = bf'_i \cup b'_j$
    end for
    $B' = B' \cup bf'_i$
end for
```

The value for the number of clusters (k) needs to be pre-determined. Choosing the optimal number of clusters is a challenging task. The Elbow method is one of the common ways to find the optimal number of clusters, which uses the inter/intra cluster distances that can be measured, for example, using Silhouette Coefficient score and Calinski-Harabasz score to evaluate and fine-tune the parameter k [5, 13, 36, 52].

The Silhouette Coefficient score is calculated for each data point d as $\frac{(y-x)}{max(x,y)}$, where x is the mean intra-cluster distance of which d corresponds to, and y is the mean nearest-cluster distance.

The elbow method executes the k-means clustering on a given dataset for different k values, and for each value, it calculates the inter/intra cluster distances, for example, using the Silhouette score. Plotting a curve between the Silhouette score and k provides a sharp point of bend (a point of the plot looks like an elbow), which is considered as the best value for k.

3 METHODOLOGY

In this section, we describe our proposed method for privacy-preserving distinct-counting of individuals/entities from multiple different databases. Our method consists of two main modules: 1) data encoding and 2) linkage and clustering. The former is conducted at the local data owner side, and the latter is conducted by the central linkage unit.

3.1 Data encoding

Data providers encode their datasets into Bloom filters, one Bloom filter per record. The PII in each of the records are hash-mapped into one record-level Bloom filter per record, as described in Section 2.1. Then, the Bloom filters are perturbed using the randomized response method to make the Bloom filters differentially private. Randomized response method is utilized to provide ϵ-local differential privacy (LDP) guarantees by flipping each bit in the Bloom filter of encoded records locally by the data providers with probability

$\eta = \frac{1}{1+e^{\epsilon}}$. Then, the perturbed Bloom filters are sent to the linkage unit for linking and clustering.

Definition 3.1 (Adjacent Bloom filters). Adjacent Bloom filters are two Bloom filters bf and bf' of length ℓ bits that differ by only one bit position, i.e. $\forall_{i, 1\leq i \leq \ell \text{ and } i \neq j} b_i = b'_i$ *and* $b_j \neq b'_j$.

Lemma 3.2 (ϵ-LDP for Bloom filters). *Flipping the bits in Bloom filters with $\frac{1}{1+e^{\epsilon}}$ probability makes the bits in the Bloom filters ϵ-local differentially private.*

Proof. Let us assume two adjacent Bloom filters bf and bf' of two records, each containing ℓ bits that differ in only one bit position j. Let $\mathcal{A} : \{0,1\}^{\ell} \rightarrow \{0,1\}^{\ell}$ be a random noise function such that $\mathcal{A}(i) = i$ with probability $\frac{e^{\epsilon}}{1+e^{\epsilon}}$, and $\mathcal{A}(i) = 1 - i$ with probability $\frac{1}{1+e^{\epsilon}}$, where $i \in \{0,1\}$.

This gives us the expression:

$$\frac{Pr[\mathcal{A}(bf,\epsilon) = \tilde{v}]}{Pr[\mathcal{A}(bf',\epsilon) = \tilde{v}]} = \prod_{i=1}^{\ell} \frac{Pr[\mathcal{A}(b_i) = \tilde{v_i}]}{Pr[\mathcal{A}(b'_i) = \tilde{v_i}]} \tag{3}$$

Note that any two adjacent Bloom filters $bf, bf' \in \{0,1\}^{\ell}$ can only differ in one bit position. Without loss of generality, let us assume that the differing bit position is the first bit position ($j = 1$) in the two Bloom filters, i.e. $b_1 \neq b'_1$ and $b_i = b'_i$ with $2 \leq i \leq \ell$.

This simplifies the ratio in (3) by considering only the first bit position.

$$\frac{Pr[\mathcal{A}(bf,\epsilon) = \tilde{v}]}{Pr[\mathcal{A}(bf',\epsilon) = \tilde{v}]} = \frac{Pr[\mathcal{A}(b_j) = \tilde{v_j}]}{Pr[\mathcal{A}(b'_j) = \tilde{v_j}]}, \tag{4}$$

where $j = 1$. This ratio is maximized when j^{th} bit position is flipped in only one of the two Bloom filters (maximum ratio).

$$e^{-\epsilon} \leq \frac{Pr[\mathcal{A}(bf,\epsilon) = \tilde{v}]}{Pr[\mathcal{A}(bf',\epsilon) = \tilde{v}]} \leq \frac{\frac{e^{\epsilon}}{1+e^{\epsilon}}}{\frac{1}{1+e^{\epsilon}}} \leq e^{\epsilon} \tag{5}$$

Bounding the above ratio, we get

$$-\epsilon \leq \ln\left(\frac{Pr[\mathcal{A}(bf,\epsilon) = \tilde{v}]}{Pr[\mathcal{A}(bf',\epsilon) = \tilde{v}]}\right) \leq \epsilon \tag{6}$$

□

By making the bits in the Bloom filters differentially private, we make them robust against cryptanalysis attacks based on sensitive bits [9]. The Bloom filter encoding function with local differential privacy is outlined in Algorithm 1. The local differentially private Bloom filters of records are then sent to the linkage unit for clustering and calculating the unique individual counts based on the number of clusters. At the linkage unit, clustering algorithm is used, as will be described in detail in the following sub-section, to link and group records which are likely to correspond to the same entity into the same cluster. The number of clusters is the number of unique individuals across multiple datasets from different data providers.

The perturbed Bloom filters are generated by randomly flipping each bit in the Bloom filters with the probability $\eta = \frac{1}{1+e^{\epsilon}}$. The Euclidean distance between an original Bloom filter bf and its perturbed Bloom filter bf' is:

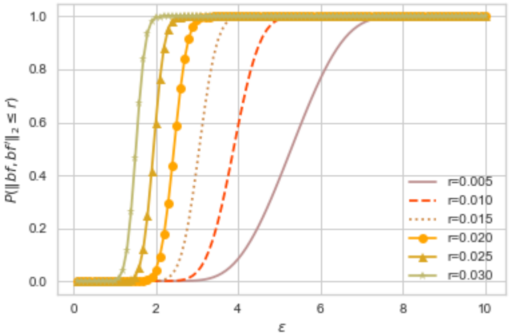


Figure 3: Probability of bf and bf' being grouped into the same cluster versus privacy budget ϵ in Equation (8), with cluster size $r \in [0.005, 0.010, 0.015, 0.020, 0.025, 0.030]$, and $\ell = 200$.

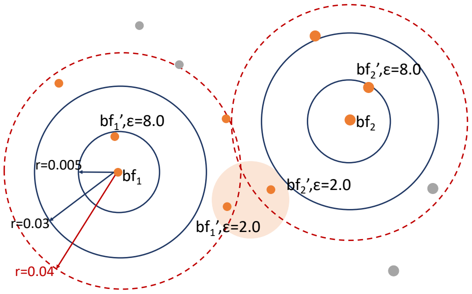


Figure 4: Bloom filters belonging to the same entity being grouped into the same cluster versus privacy budget ϵ, with cluster size $r \in [0.005, 0.030, 0.040]$, and $\ell = 200$.

$$\|bf, bf'\|_2 = \sqrt{\sum_{i=1}^{\ell}(b_i - b'_i)^2}, \tag{7}$$

where ℓ is the length of a Bloom filter, b_i is the i^{th} bit in original Bloom filter bf and b'_i is the i^{th} bit in its perturbed Bloom filter bf'. The value of $\|bf, bf'\|_2^2$ follows Binomial Distribution, which can be approximated to the Normal Distribution with a large number of ℓ and a probability η, $\|bf, bf'\|_2^2 \sim N(\mu, \sigma^2)$, $\mu = \ell\eta$, $\sigma = \sqrt{\ell\eta(1-\eta)}$. Assume if the Euclidean distance $\|bf, bf'\|_2$ is less than a constant integer value (threshold) $r \in [0, \ell]$, then the original Bloom filter and the perturbed Bloom filter are grouped into same cluster. The probability of bf and bf' being classified as the same entity is:

$$\begin{aligned} P(\|bf, bf'\|_2 \leq r) &= P\left(\sqrt{\sum_{i=1}^{\ell}(b_i - b'_i)^2} \leq r\right) \\ &= \frac{1}{2} + \frac{1}{2}\,\text{erf}\left(\frac{r^2 - \ell\eta}{\sqrt{2\ell\eta(1-\eta)}}\right) \end{aligned} \tag{8}$$

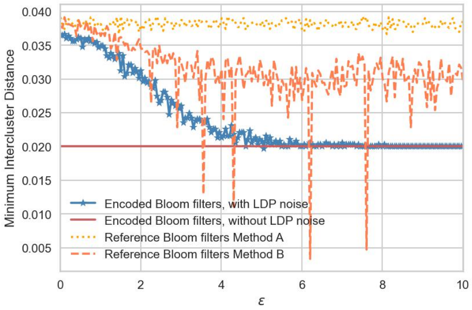


Figure 5: Minimum inter-cluster distance of Bloom filters versus privacy budget ϵ, $\ell = 200$.

As shown in Fig. 3, with the increasing privacy budget ϵ, the probability to flip bits η decreases (less noise) and thus the probability of bf and bf' being grouped into the same cluster increases. With a larger value of threshold r, a smaller value of privacy budget is required to keep bf and bf' in the same cluster. There is a trade-off between privacy budget ϵ and distance between Bloom filters bf_1 and bf_2 from two unique persons as illustrated in Fig. 4. If ϵ is small, for example $\epsilon < 2$, bf_1' and bf_2' are grouped into same cluster. If r is too large, the outlayers of bf_1 and bf_2 are overlapped. Therefore, it is a challenge to group Bloom filters from the same entity into unique clusters while to ensure Bloom filters from different entities are grouped into different clusters.

3.2 Unsupervised clustering

With the noisy encoded Bloom filters from different data providers as input to the linkage unit, a clustering algorithm is used to group similar Bloom filters into clusters. Due to privacy constraints, it is not trivial to generate labelled data to train a supervised machine learning algorithm for cardinality estimation in the privacy-preserving context. Hence an unsupervised clustering algorithm is used to do the clustering without training labels, such as k-means clustering or Hierarchical clustering. In addition, Elbow methods with inter and/or intra-cluster distance metrics like Silhouette Score and Calinski-Harabasz score are used in the literature to find the optimal number of clusters k in k-means clustering [5, 13, 36, 52]. Ideally, the optimal value k^* for number of clusters needs to be determined based on the quality of clusters generated, such as purity and completeness of clusters.

Definition 3.3 (Purity of clusters). A cluster c is pure if all the records $r \in c$ are similar to each other, i.e. belong to the same cluster.

Definition 3.4 (Completeness of clusters). A cluster c is complete if all records in the dataset D that are similar to the cluster center of c are grouped into c.

However, the traditional Elbow methods are far from accurate due to the limitation of data distribution, impurity, incompleteness and uncertainty of generated clusters, as will be validated by our experimental results in Section 4. Therefore, we propose a new algorithm to find the optimal k value in the unsupervised k-means clustering.

Algorithm 2 *Linking and clustering records for cardinality counting*

```
Inputs:
    Encoded noisy datasets from $\mathcal{N}$ multiple data providers: $B_i'$,
    $i \in [1, \mathcal{N}]$,
    Number of reference Bloom filters: $n_{\text{ref}}$,
    Number of dummy Bloom filters for each reference Bloom filter:
    $n_{\text{dum}}$,
    Flipping probability for generating dummy records for reference
    Bloom filters: $p_{\text{flip}}$
Outputs:
    $k^*$
for $i = 1, \ldots, n_{\text{ref}}$ do
    Create a reference Bloom filter $bf_{\text{ref,i}}$   ▷ obtained from either Method A or B
    $B_{\text{ref}} = B_{\text{ref}} \cup bf_{\text{ref,i}}$
    for $j = 1, \ldots, n_{\text{dum}}$ do
        Create dummy record $bf_{\text{ref,dum,j}}$
        $B_{\text{ref,dum}} = B_{\text{ref,dum}} \cup bf_{\text{ref,dum,j}}$
    end for
end for
$X = B_{\text{ref}} + B_{\text{ref,dum}} + \sum_1^{\mathcal{N}} B_i'$   ▷ X is the training dataset
for $k = 1, \ldots, n_{|D|}$ do
    $k, X \longrightarrow$ k-means and train
    Obtain $purity_i$, $\forall i \in [1, \ldots, n_{\text{ref}}]$ by Equation (9)
    $purity_k = \sum_{i=1}^{n_{\text{ref}}} purity_i$
end for
$k^* = \arg\max_k purity_k$
```

A set of reference Bloom filters with known training labels is generated to evaluate the clustering performance. Our proposed clustering algorithm first generates random Bloom filters or randomly selects a subset of Bloom filters as reference Bloom filters, and then generates corresponding dummy Bloom filters for each reference Bloom filter. The reference Bloom filters can be generated in two methods:

- Method A: randomly creates a number of fake Bloom filters following a uniform distribution.
- Method B: selects a subset of all Bloom filters from the different data providers.

As shown in Fig. 5, the distribution of reference Bloom filters generated by Method B follows a similar distribution as the original encoded Bloom filters with LDP noise. It is noted that when the privacy budget is too small $\epsilon < 1.0$, the data distribution of reference Bloom filters with pure noise generated by Method A is similar to the distribution of the encoded Bloom filters with LDP noise.

For each reference Bloom filter bf_{ref}, a number of dummy Bloom filters $b_{\text{ref,dum}_i}$, $i \in [1, n_{\text{dum}}]$ are then generated by randomly flipping each bit in reference Bloom filter bf_{ref} with the flipping probability p_{flip}. A larger p_{flip} means generating dummy Bloom filters that are far from the reference Bloom filters, while a too small p_{flip} might mean generating not a diverse set of Bloom filters to fine-tune the model. It is important to note that the reference Bloom filters from which the linkage unit generates these dummy Bloom filters are either already differentially private or fake Bloom filters (methods B and A, respectively), and hence they would not reveal any information about the original Bloom filters.

The aim of these reference and dummy Bloom filters is to provide some form of labelled data to fine-tune and evaluate unsupervised clustering techniques. In real applications, like rare disease patient records linking, ground-truth data is not available and/or accessible due to privacy constraints. Current methods to evaluate and choose the optimal number of clusters k in unsupervised clustering techniques, like k-means clustering, are based on the inter and intra similarities/distances between generated clusters, which do not provide an accurate method to evaluate how pure and complete are the generated clusters.

Our proposed method checks whether the reference Bloom filters are grouped with their corresponding dummy Bloom filters to evaluate the optimal k in the k-means clustering algorithm. Each Bloom filter is classified into a cluster with a label $c \in [0, k-1]$. In order to evaluate the clustering quality and find the best optimal k, we introduce a new *purity* function for each reference Bloom filter i at cluster c:

$$purity_i = \frac{n_{i,\text{dum},c}}{n_{i,\text{dum}} + n_c - 1 - n_{i,\text{dum},c}}, \tag{9}$$

where $n_{i,\text{dum},c}$ is the number of dummy records for i^{th} reference Bloom filter that are grouped in the same cluster with label c, n_c is the number of Bloom filters that are grouped into the cluster with label c, $n_{i,\text{dum}}$ is the total number of dummy records for i^{th} reference Bloom filter. This *purity* function measures how accurate the clustering is in terms of grouping all the dummy Bloom filters of each reference Bloom filter into the same cluster as the reference Bloom filter.

Based on this *purity* function, we calculate the purity of all reference Bloom filters with different values of k, and then find the optimal k using the Elbow method with the purity score. Current Elbow methods use silhouette score [36] or Calinski-Harabasz [5]. Our proposed clustering method is outlined in Algorithm 2.

The optimal clustering outcome is subject to:

$$\|bf_i, bf_i'\|_2 \leq r, \forall i \in [1, \ldots, n_{|D|}], \tag{10}$$

$$\|bf_i, bf_j\|_2 > r, \forall i, j \in [1, \ldots, n_{|D|}], i \neq j. \tag{11}$$

where $n_{|D|}$ is the size of all input datasets, bf_i and bf_j belong to any two unique entities in the dataset, and bf_i and bf_i' belong to the same individual in the dataset. In the ideal case, the Bloom filters corresponding to the same individual are grouped into the same cluster, while the Bloom filters corresponding to two different individuals are grouped into different clusters.

4 EXPERIMENTAL EVALUATION

In this section we present and discuss the results of experimental study of our proposed method.

Datasets: We used three sets of datasets extracted from the North Carolina Voter Registration (NCVR) database [1]. This database contains records of voters in the North Carolina State, USA. Ground-truth is available based on the voter registration identifiers to evaluate the accuracy of our proposed cardinality estimator in our experiments. We used given name (string), surname (string), suburb (string), postcode (string), and gender (categorical) attributes as PII for the linkage. The ground-truth cardinality is 171 in all three sets of datasets, i.e. the datasets contain records corresponding to 171 unique voters.

The first set contains duplicate records of the same person with no modified or corrupted PII values. The second set contains duplicate records with modified or corrupted PII values (20% of records) to reflect real-world data errors and variations, while the third set contains highly corrupted PII values (40% of records) to evaluate how real data errors impact the accuracy of cardinality estimation. We used the GeCo tool [44] to generate the synthetically corrupted/modified duplicate records. We applied various corruption functions from the GeCo tool [44] on randomly selected attribute values, including character edit operations (insertions, deletions, substitutions, and transpositions), and optical character recognition and phonetic modifications based on look-up tables and corruption rules [44]. We implemented the prototype of our proposed algorithm in Python 3.5.2, and ran all experiments on a server with four 2-core 64-bit Intel Core I7 2.6 GHz CPUs, 8 GBytes of memory and running Ubuntu 16.04. The programs and test datasets are available from the authors.

Baseline method: We compare our methods with the baseline Elbow method that uses the silhouette coefficient metric [36] to find the optimal number of clusters. We do not compare with other existing cardinality estimators as they do not allow fuzzy matching for counting the cardinality and hence do not provide a fair comparison. The accuracy of count estimation is measured using the estimation error and error rate, i.e. the difference between true count and estimated count and rate of error. Privacy is measured using the privacy budget ϵ.

Parameter setting: Default parameter setting for the Bloom filter encoding is $q = 2$ for strings, length of Bloom filters is $\ell = 200$, and the number of hash functions is 20. Privacy budgets used are $\epsilon = [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. It is important to note that, unlike with central differential privacy, with local differential privacy achieving a high utility with a very small privacy budget (≤ 1) is non-trivial. When $\epsilon = 0.1$, almost 50% of the bits in the Bloom filters need to be flipped, which makes the Bloom filters completely non-informative. Other local differential privacy algorithms proposed, for example by Google for RAPPOR statistics and Apple for mobile usage statistics [1, 17], also use ϵ in the range of $\epsilon \in [1.0, 2.0, 4.0, 6.0, 8.0]$. $\epsilon = 10.0$ is used as a baseline with no privacy guarantees. For the clustering algorithm, the default reference Bloom filters pick ratio used is 0.1, the default dummy/noisy Bloom filter ratio is set to 0.1, and the flipping probability for the dummy/noisy Bloom filters is used in the range $[0.10 - 0.30]$, with a step of 0.01. We vary the flipping probabilities in the dummy Bloom filters and evaluate the k value and error rate as it impacts the quality of clustering depending on the data quality and privacy budget. We ran each of the experiments 50 times and reported the average results.

Discussion: We first compare the optimal k value (k^*) provided by our algorithm with the ground-truth cardinality and evaluate the error in estimation with different flipping probabilities used in the clustering algorithm on the records encoded with different privacy budgets. The results of our methods (Method A and Method B) compared with the baseline method on the clean dataset are shown in Fig. 6. As shown, the ground-truth cardinality is 171. When $\epsilon = 1.0$, 26% of bits are flipped in the Bloom filters, which means 52 out of 200 bits are flipped making most of the Bloom filters unique. Hence, the estimated cardinality by Method A is 200, leading to an error of 29. Even when the flipping probability

[1] Available from http://dl.ncsbe.gov/data/

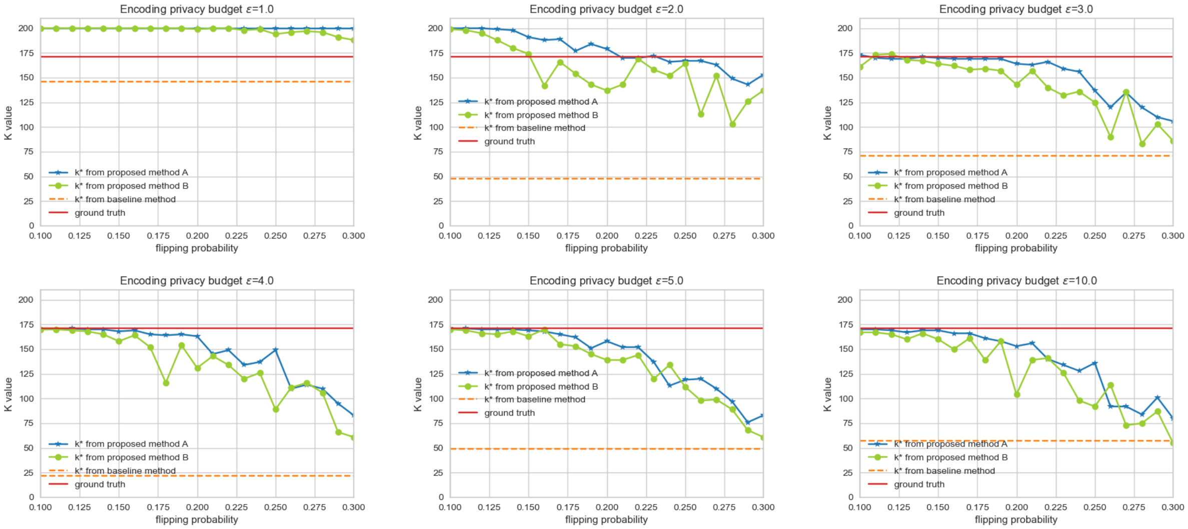


Figure 6: Estimated cardinality (k value) of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the clean datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

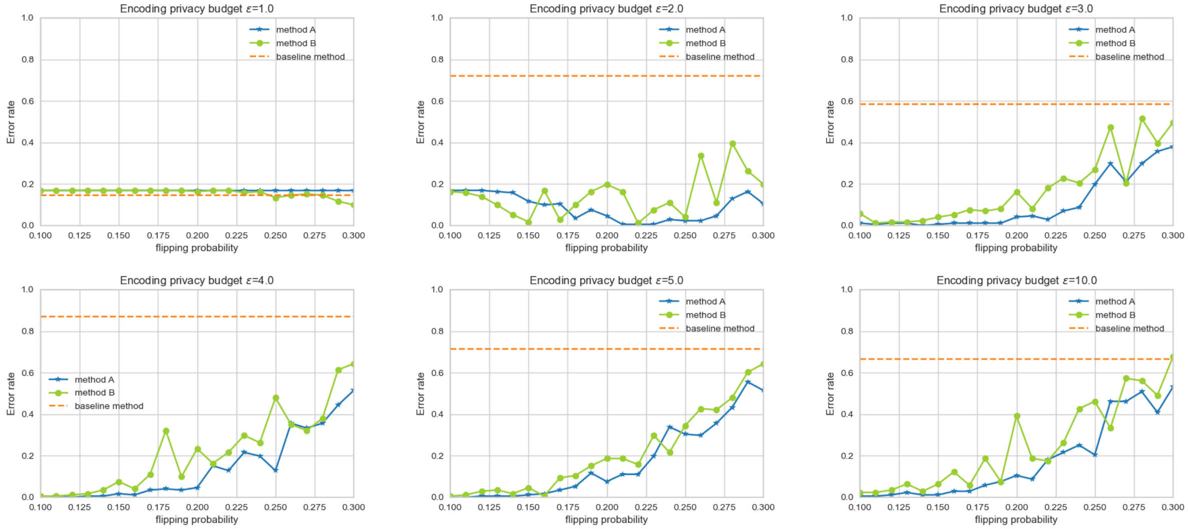


Figure 7: Error rate of cardinality estimation of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the clean datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

in dummy Bloom filters reduces to 0.0, the estimated cardinality remains constantly as 200 with Method A. Compared to Method A, Method B has better performance in terms of smaller error of 7. The reason is that Method B selects a subset of real Bloom filters from all data providers as the reference Bloom filters, whereas Method A randomly generates fake Bloom filters as reference Bloom filters.

With privacy budgets larger than 1.0 the error in estimation becomes 0.0 for smaller flipping probabilities, i.e. the optimal k becomes equal to the ground-truth with both our methods. With increasing ϵ, zero or smaller error can be achieved even with larger flipping probabilities. When $\epsilon = 10.0$, zero error is achieved with the estimation for flipping probabilities up to 0.15. However, as can be seen, the baseline method's estimated cardinality (optimal k) is far from the ground truth value with all privacy budgets, even with larger ϵ. Our methods significantly outperform the baseline method by providing the optimal k value equal or closer to the ground truth value leading to smaller or zero error rate on all datasets. Error

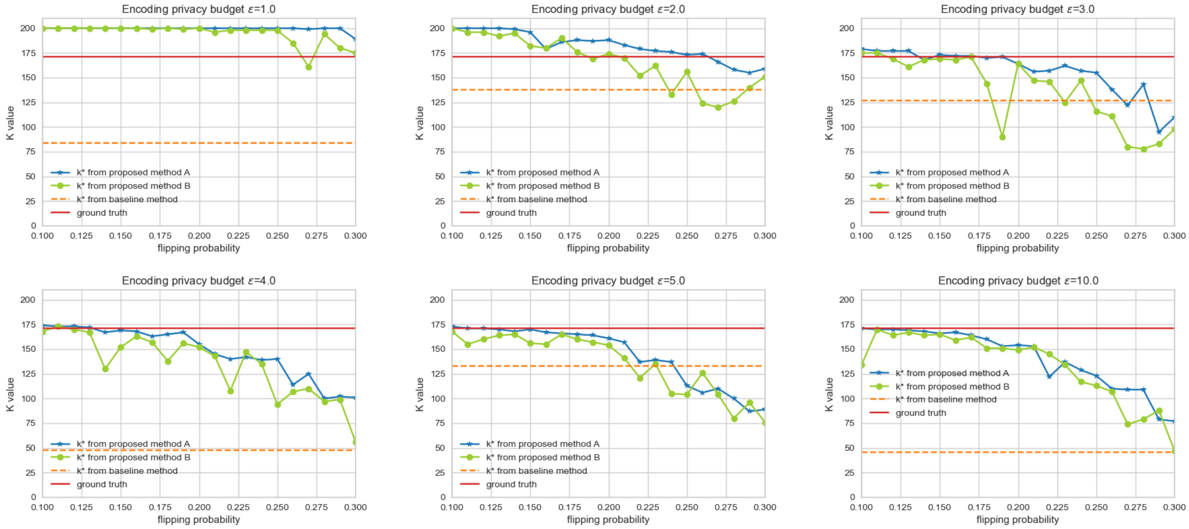


Figure 8: Estimated cardinality (k value) of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the corrupted datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

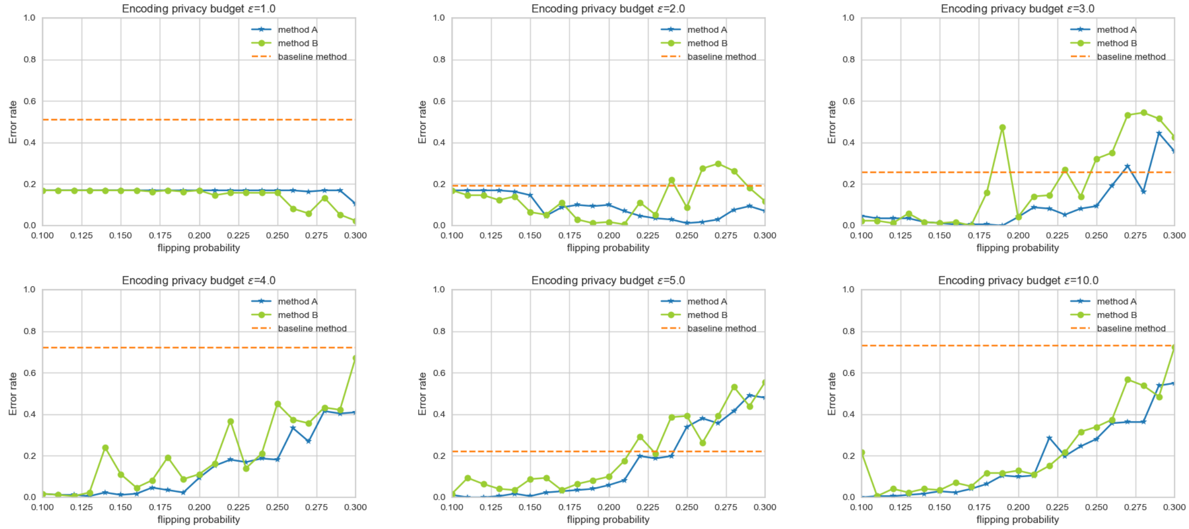


Figure 9: Error rate of cardinality estimation of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the corrupted datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

rates are shown in Fig. 7. These results show the accuracy of our methods compared to the baseline method.

Given the global and distributed nature of the data provider network, we expect inconsistencies in the PII attributes among others. We hence evaluate our proposed methods on corrupted datasets to validate the effectiveness of our methods with inconsistent and low quality data. The estimated cardinalities of Method A and Method B for different flipping probabilities and different privacy budgets compared with the baseline method on corrupted datasets are shown in Fig. 8, and the error rates in estimations are shown in Fig. 9. Both Method A and Method B perform mostly similar on this dataset and achieved similar error rates in cardinality estimation. Since the datasets are already corrupted, random generation of reference Bloom filters and sampling a subset of corrupted records' Bloom filters do not make significant difference with these datasets. As can be seen in the results, optimal k with a very small error rate closer to 0.0 can be found with both methods, and with increasing ϵ, optimal error rates of 0.0 (for smaller flipping probabilities) are

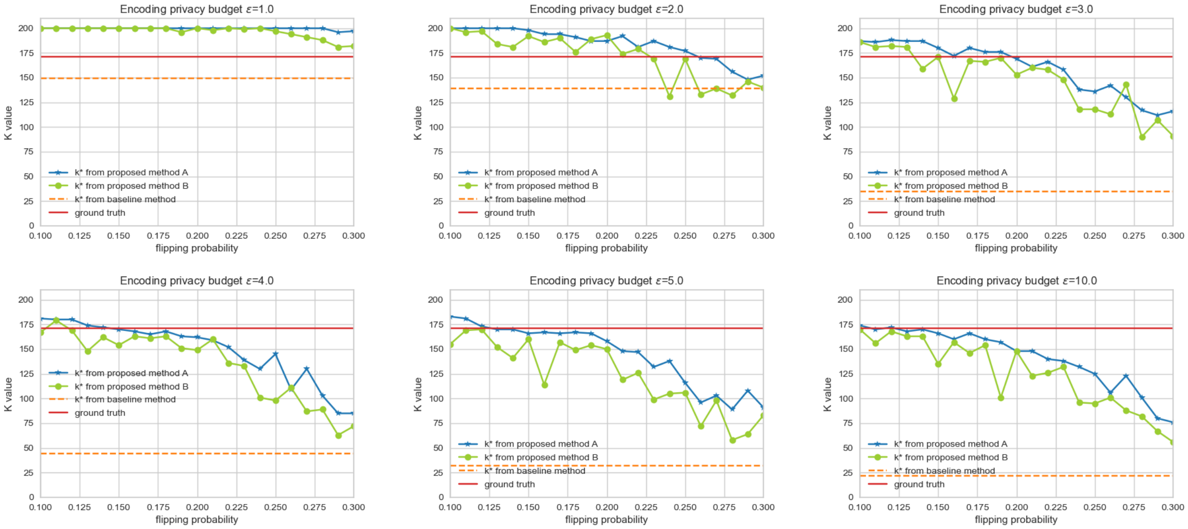


Figure 10: Estimated cardinality (k value) of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the highly corrupted datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

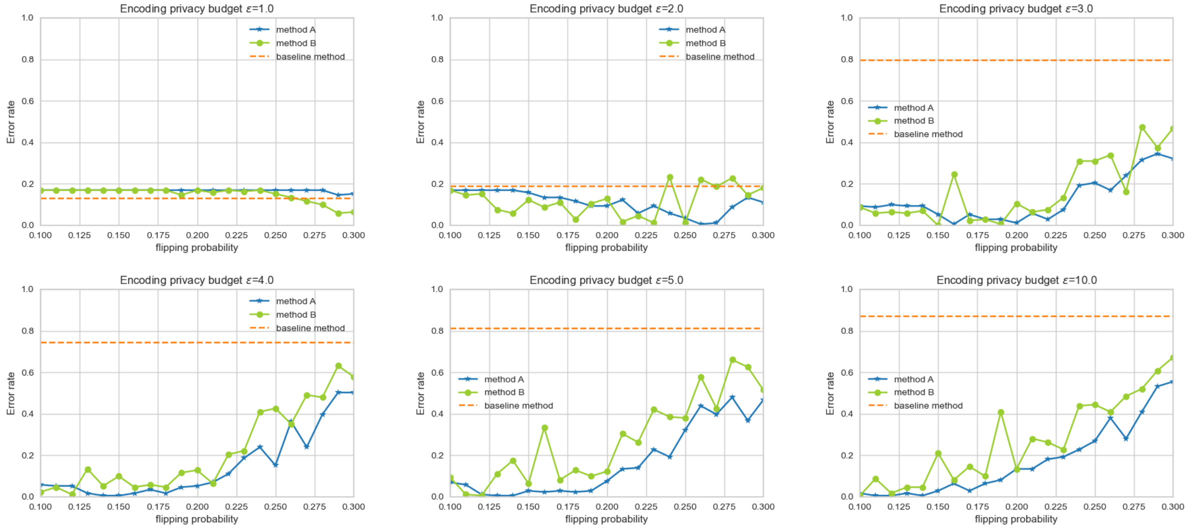


Figure 11: Error rate of cardinality estimation of Method A and Method B with different flipping probabilities compared with the baseline method [36] on the highly corrupted datasets with $\epsilon \in [1.0, 2.0, 3.0, 4.0, 5.0, 10.0]$. The reference Bloom filters pick ratio is 0.1 and dummy Bloom filters ratio is 0.1 in these experiments.

achieved. We can observe that achieving a high utility in terms of lower error rate with a small budget becomes challenging with data with errors and variations compared to clean datasets. However, our methods significantly outperform the baseline method and still be able to achieve a small optimal error rate < 0.02. For example, when $\epsilon = 3.0$, our methods have an optimal error rate of 0.02 when the flipping probabilities are less than 0.175, while the Elbow method with silhouette coefficient metric has an error rate of 0.26.

Generally, as expected for larger ϵ (e.g. $\epsilon \geq 3.0$), with increasing flipping probabilities the error rate increases as well. On the other hand, with smaller ϵ ($\leq$ 2.0), more differential privacy noise is added to Bloom filters, and hence, larger flipping probabilities might mean that the added noise in the Bloom filters is reduced, leading to smaller error rate with larger flipping probabilities. This is especially the case with corrupted datasets. Hence, depending on the data quality level and privacy budget, an appropriate flipping probability can be chosen to get the best performance. Optimising the flipping probability based on the privacy and data quality constraints is left as a future work.

We finally evaluate the performances on highly corrupted datasets. The results for Method A and Method B are shown in Fig. 10 and Fig. 11. Similar to corrupted datasets, both methods provide similar error rates with Method B being slightly better in terms of achieving optimal lower error rates for smaller ϵ than Method A. As can be seen, achieving a small error rate is even more challenging with these highly corrupted datasets compared to corrupted datasets. Our methods still achieve a small error rate, for example less than 0.05 with $\epsilon = 3.0$ in the presence of data errors and variations, whereas the baseline Elbow method achieves 0.8 error rate. Providing a higher error rate even when no differential privacy noise ($\epsilon = 10.0$) is added indicates the ineffectiveness of the baseline method. Our methods can achieve a very small error rate even with a small privacy budget and on highly corrupted datasets by fine-tuning the flipping probability parameter depending on the privacy and data quality constraints.

5 RELATED WORK

Different approaches have been proposed to estimate the cardinality of multiple sets, as surveyed in [28]. The naive approach of using a bitmap of size of the universe, where all the bit positions are initialized to 0 and each item is assigned with a number and therefore corresponding bit position in the bitmap is set to 1 whenever an item is observed, is not feasible. Sorting is used as another traditional method where the items are sorted to eliminate duplicates in the sets [54]. However, sorting is an expensive operation for large sets. Hashing allows de-duplication of sets in one pass over the sets without sorting them, however, it requires more memory space.

While these methods allow calculating the exact cardinality of sets, they are not only expensive in terms of both memory size and runtime, but also are not effective with real data that contain data errors, typos, and variations. Fuzzy matching for record linkage methods have been investigated [8]. A Bloom filter is a probabilistic data structure used for efficiently checking set membership [4]. This can be used for fuzzy matching problem [33, 47] effectively with appropriate parameter settings for the Bloom filter [35, 39, 45]. Another general approach is sampling [20, 22, 27] which assumes that the sample generally reflects the properties of the whole. Ensuring true randomness is a difficult task, so the success of random sampling may be limited by the selection process and/or the properties of the data itself. Haas et al. [27] showed that almost all the data need to be sampled in order to bound the estimation error within a small constant, which reflects the problem with sampling-based approaches.

Employing other types of probabilistic data structures, such as Sketches and HyperLogLog, is used as an efficient and effective method in several cardinality estimation algorithms. A family of such algorithms are developed by Flajolet and Martin [23]. HyperLogLog is one of these algorithms that have widely been used in many applications and research [2, 7, 43]. Several recent works have studied privacy-preserving cardinality estimators using probabilistic data structures combined with differential privacy. Randomized response-based differentially private algorithms for Bloom filters [42], FM-sketch [50], and K Minimum Values (KMV)-based sketch [41] have been developed. A recent work has shown that cardinality estimators, such as HyperLogLog and Sketches, do not preserve privacy without impacting the utility to a significant level [12]. Further, none of these works allow fuzzy matching to count the cardinalities, making the cardinality estimators not robust or tolerant to data errors and variations in the duplicate records.

A long line of research has been conducted in privacy-preserving fuzzy matching and linkage over the past three decades, as surveyed in [24, 48, 49]. While machine learning-based techniques show promising results in terms of high linkage quality, these are often supervised, i.e. they are dependant on significantly large training data and the existence of ground-truth labels. Only few unsupervised techniques have been developed for linkage [11, 29, 37, 46]. However, most of these techniques either do not consider privacy constraints or are not capable of fine-tuning/optimising the clustering performance due to unlabelled data.

6 DISCUSSION, LIMITATIONS AND FUTURE WORK

In this paper we have addressed the problem of privacy-preserving cardinality estimation of individuals/entities represented by records from multiple databases. Our proposed method uses Bloom filter encoding with local differential privacy to encode the data and unsupervised clustering to fuzzy link records and calculate the optimal number of clusters as the cardinality of unique individuals. We propose a novel method to calculate the optimal number of clusters in the absence of ground-truth labels of matching and non-matching records, which is often the case with privacy-preserving applications. Our experimental results show that, compared to the baseline Elbow method, our method can achieve a high accuracy of cardinality estimation even on corrupted records with a small privacy budget.

In the future, we aim to apply our proposed algorithm for the rare disease patient counting application. Rare disease patient counting application involves small-scale datasets as the number of patients with rare disease is generally small – most rare diseases often have 10, 100 or just 1000 patients spread across the world. However, experimenting on large datasets for other applications of cardinality estimation and improving the scalability to large databases is one important future work. Moreover, optimising the flipping probability constrained on the level of data quality and privacy budget is yet to be investigated and implemented in our algorithm. Our proposed ϵ-local differentially private algorithm provides robustness against cryptanalysis attacks on Bloom filter encodings. Evaluating the robustness of the linkage algorithm to other types of attacks, such as membership inference attacks or adversarial attacks is left as an important future work. Finally, facilitating real-time counting and efficient dynamic updates without requiring to re-do clustering is an important yet challenging research direction.

ACKNOWLEDGMENTS

This research has been partially funded by the OpenTreatments Foundation and the strategic research funds of the Macquarie University Cyber Security Hub.

REFERENCES

[1] Differential Privacy Team Apple. 2017. Learning with privacy at scale. *Apple Machine Learning Journal - Online at: https://machinelearning.apple.com/2017/12/06/learning-with-privacy-at-scale.html* (2017).

[2] Ramesh Balasubramaniam and K Nandhini. 2019. Algorithms Associated with Streaming Data Problems. *International Journal of Applied Engineering Research* 14, 9 (2019), 2238–2243.

[3] Ziv Bar-Yossef, TS Jayram, Ravi Kumar, D Sivakumar, and Luca Trevisan. 2002. Counting distinct elements in a data stream. In *International Workshop on Randomization and Approximation Techniques in Computer Science*. Springer, 1–10.

[4] B.H. Bloom. 1970. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* 13, 7 (1970), 422–426.

[5] Tadeusz Caliński and Jerzy Harabasz. 1974. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods* 3, 1 (1974), 1–27.

[6] Yousra Chabchoub and Georges Hébrail. 2010. Sliding hyperloglog: Estimating cardinality in a data stream over a sliding window. In *International Conference on Data Mining Workshops*. IEEE, 1297–1303.

[7] Guoqiang Jerry Chen, Janet L Wiener, Shridhar Iyer, Anshul Jaiswal, Ran Lei, Nikhil Simha, Wei Wang, Kevin Wilfong, Tim Williamson, and Serhat Yilmaz. 2016. Realtime data processing at Facebook. In *International Conference on Management of Data*. 1087–1098.

[8] Peter Christen. 2012. *Data matching - concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer.

[9] Peter Christen, Thilina Ranbaduge, Dinusha Vatsalan, and Rainer Schnell. 2018. Precise and fast cryptanalysis for Bloom filter based privacy-preserving record linkage. *IEEE Transactions on Knowledge and Data Engineering* (2018), 1.

[10] Peter Christen, Anushka Vidanage, Thilina Ranbaduge, and Rainer Schnell. 2018. Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In *PAKDD, Springer LNAI*. Melbourne, 530–542.

[11] William W. Cohen and Jacob Richman. 2002. Learning to Match and Cluster Large High-dimensional Data Sets for Data Integration. In *ACM SIGKDD*. 475–480.

[12] Damien Desfontaines, Andreas Lochbihler, and David Basin. 2019. Cardinality estimators do not preserve privacy. *Proceedings on Privacy Enhancing Technologies* 2019, 2 (2019), 26–46.

[13] Duy-Tai Dinh, Tsutomu Fujinami, and Van-Nam Huynh. 2019. Estimating the optimal number of clusters in categorical data clustering by silhouette coefficient. In *International Symposium on Knowledge and Systems Sciences*. Springer, 1–17.

[14] C. Dwork. 2006. Differential privacy. *International Colloquium on Automata, Languages and Programming* (2006), 1–12.

[15] Cynthia Dwork. 2008. Differential privacy: A survey of results. In *Theory and Applications of Models of Computation*. Springer, 1–19.

[16] Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N Rothblum, and Sergey Yekhanin. 2010. Pan-Private Streaming Algorithms.. In *ICS*. 66–80.

[17] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *SIGSAC conference on computer and communications security*. ACM, 1054–1067.

[18] Otmar Ertl. 2017. New cardinality estimation algorithms for HyperLogLog sketches. *arXiv preprint arXiv:1702.01284* (2017).

[19] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. 2003. Limiting privacy breaches in privacy preserving data mining. In *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 211–222.

[20] Philippe Flajolet. 1990. On adaptive sampling. *Computing* 43, 4 (1990), 391–400.

[21] Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. 2007. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Conference on Analysis of Algorithms (AofA)*. Nancy, France.

[22] Phillip B Gibbons. 2001. Distinct sampling for highly-accurate answers to distinct values queries and event reports. In *VLDB*, Vol. 1. 541–550.

[23] Phillip B Gibbons. 2016. Distinct-values estimation over data streams. In *Data Stream Management*. Springer, 121–147.

[24] A. Gkoulalas-Divanis, D. Vatsalan, D. Karapiperis, and M. Kantarcioglu. 2021. Modern Privacy-Preserving Record Linkage Techniques: An Overview. *IEEE TIFS* (2021).

[25] Nikolay Golov, Alexander Filatov, and Sergey Bruskin. 2019. Efficient Exact Algorithm for Count Distinct Problem. In *International Workshop on Computer Algebra in Scientific Computing*. Springer, 67–77.

[26] Andy Greenberg. 2016. Apple's 'differential privacy'is about collecting your data—but not your data. *Wired, June* 13 (2016).

[27] Peter J Haas, Jeffrey F Naughton, S Seshadri, and Lynne Stokes. 1995. Sampling-based estimation of the number of distinct values of an attribute. In *VLDB*, Vol. 95. 311–322.

[28] Hazar Harmouch and Felix Naumann. 2017. Cardinality estimation: An experimental survey. *Proceedings of the VLDB Endowment* 11, 4 (2017), 499–512.

[29] Oktie Hassanzadeh, Fei Chiang, Hyun Chul Lee, and Renée J Miller. 2009. Framework for evaluating clustering algorithms in duplicate detection. *Proceedings of the Very Large Database Endowment* 2, 1 (2009), 1282–1293.

[30] Stefan Heule, Marc Nunkesser, and Alexander Hall. 2013. HyperLogLog in practice: algorithmic engineering of a state of the art cardinality estimation algorithm. In *International Conference on Extending Database Technology*. 683–692.

[31] Bernard J Jansen. 2006. Search log analysis: What it is, what's been done, how to do it. *Library & information science research* 28, 3 (2006), 407–432.

[32] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What can we learn privately? *SIAM J. Comput.* 40, 3 (2011), 793–826.

[33] Abhishek Kumar, Jun Xu, and Jia Wang. 2006. Space-code bloom filter for efficient per-flow traffic measurement. *IEEE Journal on Selected Areas in Communications* 24, 12 (2006), 2327–2339.

[34] Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. 2017. Generating synthetic decentralized social graphs with local differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security*. 425–438.

[35] Sean M Randall, Anna M Ferrante, James H Boyd, and James B Semmens. 2014. Privacy-preserving record linkage on large real world datasets. *Journal of Biomedical Informatics* 50, 1 (2014), 1.

[36] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.

[37] Alieh Saeedi, Markus Nentwig, Eric Peukert, and Erhard Rahm. 2018. Scalable matching and clustering of entities with FAMER. *Complex Systems Informatics and Modeling Quarterly* 16 (2018), 61–83.

[38] R. Sakate, A. Fukagawa, Y. Takagaki, H. Okura, and A. Matsuyama. 2018. Trends of Clinical Trials for Drug Development in Rare Diseases. *Curr Clin Pharmacol* 13, 3 (2018), 199–208.

[39] Rainer Schnell. 2016. Privacy preserving record linkage. In *Methodological developments in data linkage*, Katie Harron, Harvey Goldstein, and Chris Dibben (Eds.). Wiley, Chichester, 201–225.

[40] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Alessandro Flammini, and Filippo Menczer. 2017. The spread of fake news by social bots. *arXiv preprint arXiv:1707.07592* 96 (2017), 104.

[41] Hagen Sparka, Florian Tschorsch, and Björn Scheuermann. 2018. P2KMV: a privacy-preserving counting sketch for efficient and accurate set intersection cardinality estimations. (2018).

[42] Rade Stanojevic, Mohamed Nabeel, and Ting Yu. 2017. Distributed cardinality estimation of set operations with differential privacy. In *2017 IEEE Symposium on Privacy-Aware Computing (PAC)*. IEEE, 37–48.

[43] Hong Su, Mohamed Zait, Vladimir Barrière, Joseph Torres, and Andre Menck. 2016. Approximate aggregates in oracle 12c. In *ACM International on Conference on Information and Knowledge Management*. 1603–1612.

[44] Khoi-Nguyen Tran, Dinusha Vatsalan, and Peter Christen. 2013. GeCo: an online personal data generator and corruptor. In *ACM Conference in Knowledge Management*. San Francisco, 2473–2476.

[45] Dinusha Vatsalan and Peter Christen. 2016. Privacy-preserving matching of similar patients. *Journal of Biomedical Informatics* 59 (2016), 285–298.

[46] Dinusha Vatsalan, Peter Christen, and Erhard Rahm. 2020. Incremental clustering techniques for multi-party Privacy-Preserving Record Linkage. *Data & Knowledge Engineering* (2020).

[47] D. Vatsalan, P. Christen, and Vassilios S. Verykios. 2011. An Efficient Two-Party Protocol for Approximate Matching in Private Record Linkage. In *Australasian Data Mining Conference*. Ballarat, Australia.

[48] Dinusha Vatsalan, Peter Christen, and Vassilios S. Verykios. 2013. A Taxonomy of Privacy-Preserving Record Linkage Techniques. *Information Systems* 38, 6 (2013), 946–969.

[49] Dinusha Vatsalan, Ziad Sehili, Peter Christen, and Erhard Rahm. 2017. Privacy-preserving record linkage for Big data: Current approaches and research challenges. In *Handbook of Big Data Technologies*. Springer, 851–895.

[50] Saskia Nuñez von Voigt and Florian Tschorsch. 2019. RRTxFM: Probabilistic Counting for Differentially Private Statistics. In *Conference on e-Business, e-Services and e-Society*. Springer, 86–98.

[51] Gang Wang, Xinyi Zhang, Shiliang Tang, Christo Wilson, Haitao Zheng, and Ben Y Zhao. 2017. Clickstream user behavior models. *ACM Transactions on the Web (TWEB)* 11, 4 (2017), 1–37.

[52] Xu Wang and Yusheng Xu. 2019. An improved index for clustering validation based on Silhouette index and Calinski-Harabasz index. In *IOP Conference Series: Materials Science and Engineering*, Vol. 569. IOP Publishing, 052024.

[53] Stanley L Warner. 1965. Randomized response: A survey technique for eliminating evasive answer bias. *J. Amer. Statist. Assoc.* 60, 309 (1965), 63–69.

[54] Kyu-Young Whang, Brad T Vander-Zanden, and Howard M Taylor. 1990. A linear-time probabilistic counting algorithm for database applications. *ACM Transactions on Database Systems (TODS)* 15, 2 (1990), 208–229.