Reference Values based Hardening for Bloom Filters based Privacy-Preserving Record Linkage

Sirintra Vaiwsri, Thilina Ranbaduge, and Peter Christen

Research School of Computer Science, The Australian National University, Canberra, ACT 2601, Australia, sirintra.vaiwsri@anu.edu.au

Abstract. Privacy-preserving record linkage (PPRL) is the process of identifying records that refer to the same entities across different databases without revealing any sensitive information about these entities. A popular PPRL technique that is efficient and effective is Bloom filter encoding. However, recent research has shown that Bloom filters are vulnerable to cryptanalysis attacks that aim to re-identify sensitive attribute values encoded into Bloom filters. As counter-measures, hardening techniques have been developed that modify the bit patterns in Bloom filters. One recently proposed hardening technique is BLoom-and-flIP (BLIP), which randomly flips bit values according to a differential privacy mechanism. However, while making Bloom filters more resilient to attacks, applying BLIP can lower linkage quality. We propose and evaluate a reference-based BLIP mechanism which ensures that Bloom filters for similar encoded sensitive values are modified in a similar way, resulting in improved linkage quality compared to standard BLIP hardening.

Keywords: Data linkage, Differential Privacy, Encoding, Perturbation.

1 Introduction

Many organizations collect millions of records about individuals (such as customers, patients, or tax payers) in their databases which often need to be integrated to facilitate effective data mining. *Record linkage* aims to link records in different databases that refer to the same entity [5]. Privacy is an important aspect that needs to be considered when databases that contain sensitive personal data, such as names and addresses, are linked across databases.

Privacy-preserving record linkage (PPRL) [20] techniques have been developed to match records between databases without revealing sensitive data. In PPRL the values in a set of attributes common to all databases are encoded in some form to ensure their privacy. Different categories of privacy techniques have been developed for PPRL [20]. The first category are secure multi-party computation (SMC) based techniques that perform matching of encrypted records. While provably secure, SMC techniques generally have high computation costs. The second category are perturbation based techniques which modify the actual attribute values using an encoding technique, resulting in a trade-off between linkage quality, scalability to linking large databases, and privacy [21].

A widely used perturbation technique for PPRL is Bloom filter (BF) encoding [3, 15]. As we discuss in detail in Sect. 3, a BF is a bit vector that encodes values using a set of independent hash functions [15]. BF encoding is now being used in several linkage applications in the health sector [4]. However, recent studies have shown that sets of BFs can be attacked with the aim of re-identifying the sensitive attribute values encoded in them [6, 7, 12, 13]. Most attack methods exploit that frequent BFs or bit patterns correspond to frequent q-grams (sub-strings of length q characters) in the sensitive values encoded in BFs.

To counteract such attack methods, hardening techniques have been developed to improve the security of BF based PPRL techniques [16, 19]. As we discuss in detail in the next section, these hardening techniques further modify BFs to reduce or eliminate any frequency information that could be exploited by attack methods. One drawback of existing hardening techniques is however that they have a trade-off between privacy and linkage quality, because modifications of BF bit patterns will likely lead to an increase in falsely matched and missed true matching record pairs. Certain hardening techniques have also shown to be vulnerable to a frequency-based cryptanalysis attack [6].

One recently proposed hardening technique is BLoom-and-flIP (BLIP) [2, 16] which flips bit values at certain positions in a BF according to a differential privacy mechanism [9]. In our evaluation we show that such random bit flipping can lead to a considerable decrease in linkage quality. To overcome this weakness of BLIP hardening, we propose to use reference values from a global database to determine the bit positions to be flipped. The use of reference values ensures that similar BFs are modified in the same way (thus maintaining high similarities) while different BFs are modified differently (resulting in lower similarities). We name our approach as RBBF for **R**eference based **B**LIP **BF** hardening.

In this paper we specifically contribute (1) a novel approach to select a suitable set of reference values from a publicly available large database; (2) an improved BLIP hardening technique for BFs based on selected single and multiple reference values; (3) an analysis of our approach in terms of complexity and linkage quality; and (4) an experimental evaluation using a real-world database.

2 Related Work

Since the mid 1990s PPRL techniques have been developed to link sensitive data without having to reveal any actual attribute values. PPRL has evolved from simple exact matching of encrypted strings only to sophisticated approximate matching of encoded values in large databases [20].

In 2009 Schnell et al. [15] proposed to use BF encoding for scalable PPRL that also allowed approximate matching of records by calculating similarities between BFs. Inspired by their approach various BF based PPRL techniques have been developed since then [20]. Varying from two-party to multi-party protocols, these techniques classify record pairs as matches and non-matches based on the number of 1-bits their corresponding BFs have in common [15].

Recently, several attacks on BF encodings for PPRL have been proposed that aim to re-identify the attribute values encoded in BFs [6, 7, 12, 13]. Most of these attacks are based on a frequency analysis of bit pattern distributions. To overcome these attacks hardening methods can be applied on BFs [16, 18].

Salting is a hardening technique that can be used for PPRL [17] with the aim to create different bit patterns for the same q-gram by adding an extra value to each q-gram before it is encoded (such as the year of birth for a person). Therefore two attribute values that have the same q-gram set but different salting values (like different years of birth) will be mapped to different bit positions in a BF.

Balancing was proposed by Schnell and Borgs [16] as a hardening method to generate uniform Hamming weight (number of 1-bits) distributions for BFs. To generate a balanced BF, a BF is concatenated with the negated copy of itself, such that the resulting BF will always have 50% of its bits set to 1. The bits in the balanced BFs are then randomly permuted to improve privacy. XOR folding is another hardening method proposed by the same authors [18], where a given BF of length l is first divided into two segments of length $l/2$ and then the bit-wise XOR operation is applied on these segments to generate a new BF.

However, both balanced and XOR folded BFs have been successfully attacked. A recently proposed attack method [6] was able to correctly re-identify some of the attribute values that have been encoded into hardened BFs because the frequency distribution of BFs and their bit patterns does not change even after balancing or XOR folding has been applied.

A novel hardening technique is 'BLoom-and-flIP' (BLIP) [1, 2, 16], which randomly flips values in certain bit positions in BFs based on differential privacy characteristics [9]. The approach is similar to the RAPPOR method [10] which is being used to anonymously collect user responses during sensitive data collections. As we detail in Sect. 3, one drawback of the original BLIP approach is that the random bit flipping can lead to a significant loss of linkage quality.

In contrast to the original BLIP approach, we use reference values to improve the quality of linked records. The idea of using reference values extracted from a (publicly available) global database for PPRL was first investigated by Pang et al. [14]. However, the use of reference values in the BF encoding and hardening process has not been explored so far.

3 Background and Preliminaries

We now describe the building blocks required for our improved BLIP hardening technique which we then discuss in detail in Sect. 4.

Bloom Filter Encoding for PPRL: Bloom filter (BF) encoding [3] is a widely used perturbation techniques for PPRL [20]. A BF $\mathbf{b}$ is a bit vector of length l initially set to 0-bits. In PPRL the string values from the records to be compared in the linkage process are first converted into character q-grams which are then encoded into a BF using a set of independent hash functions [15] by setting corresponding bits to 1, as shown in Fig. 1.

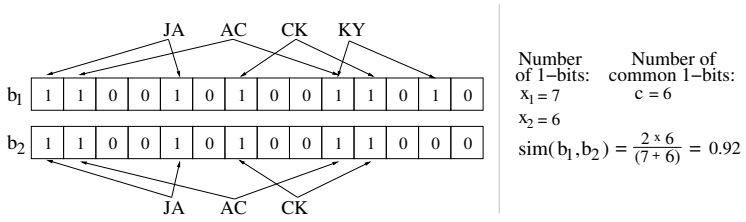

Fig. 1: Two example Bloom filters that are encoding the string value pair 'JACKY' and 'JACK' using two hash functions, with their Dice coefficient similarity calculation.

The similarity between two BFs $\mathbf{b}_1$ and $\mathbf{b}_2$ can be calculated using the Dice-coefficient [5, 15]. First, the number 1-bits of each BF, x_1 and x_2, and the number of 1-bits that occur in common at the same bit positions in both BFs, c, are counted. The similarity is then calculated as: $sim(\mathbf{b}_1, \mathbf{b}_2) = (2 \times c)/(x_1 + x_2)$.

BLoom-and-flIP (BLIP) Hardening: BLIP was originally proposed by Alaggan et al. [2] as a non-interactive differentially private [9] approach to randomize BFs in the context of privacy-preserving comparisons of user profiles in social networks. BLIP randomly flips bits at certain positions in a BF based on a user defined flip probability. We refer the reader to Alaggan et al. [2] for details and a proof showing how BLIP fulfills non-interactive differential privacy. Schnell and Borgs were the first to explore BLIP in the context of PPRL [16].

For a given bit flipping probability, f, following Alaggan et al. [2], a bit $\mathbf{b}[p]$ in a BF $\mathbf{b}$ at position p is flipped according to Eq. (1) resulting in the value $\mathbf{b}'[p]$ at position p in the new randomized BF $\mathbf{b}'$.

$$\mathbf{b}'[p] = \begin{cases} 1 & \text{if } \mathbf{b}[p] = 0 \text{ with probability } f, \\ 0 & \text{if } \mathbf{b}[p] = 1 \text{ with probability } f, \\ \mathbf{b}[p] & \text{with probability } 1 - f. \end{cases} \quad (1)$$

The BLIP approach used by Schnell and Borgs [16] was based on the idea proposed by Erlingsson et al. [10] as part of their RAPPOR technique which allows anonymous collection of user statistics from software products such as Web browsers. Again assuming a flip probability f, the new bit $\mathbf{b}'[p]$ at position p in the new randomized BF $\mathbf{b}'$ is flipped from $\mathbf{b}[p]$ according to Eq. (2):

$$\mathbf{b}'[p] = \begin{cases} 1 & \text{with probability } \frac{1}{2}f, \\ 0 & \text{with probability } \frac{1}{2}f, \\ \mathbf{b}[p] & \text{with probability } 1 - f. \end{cases} \quad (2)$$

If for example the flip probability is set to $f = 0.05$ for a BF of length $l = 1,000$ bits then 50 randomly selected bits will be flipped using the first approach from Eq. (1) while 950 bits are unchanged. With the approach used by Schnell and Borgs [16] in Eq. (2), however, bits will not be flipped according to their original state, but rather 50 randomly selected bits will be set to 0 or 1 with equal probability. As a result, depending upon which BLIP approach is used, the numbers of 1-bits in randomized BFs will likely differ.

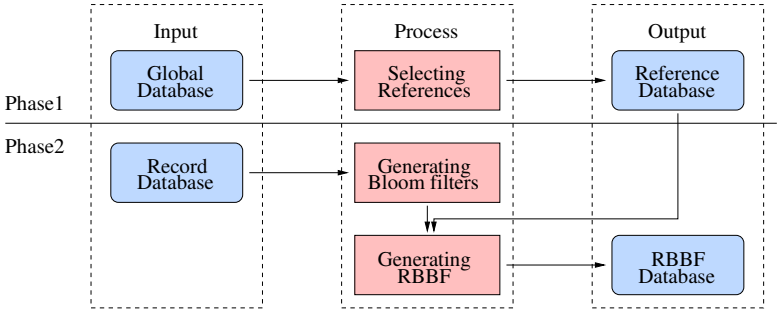

Fig. 2: Overview of reference value based BLIP Bloom Filter hardening.

If a BF has less than 50% 1-bits then applying Eq. (1) will mean it will have more 1-bits after randomization that when applying Eq. (2). This can potentially lead to lower linkage quality because more 1-bits can increase the similarities between randomized BFs and thus leads to more false positive matches.

4 Protocol Description

As outlined in Fig. 2, in the first phase of our approach we select a set of suitable reference values from a global database, and in the second phase we use these reference values to determine how to apply BLIP when randomizing BFs.

4.1 Phase 1: Selecting Reference Values

We assume all database owners (DOs) [20] who aim to encode and harden their sensitive databases have access to a publicly available global database $\mathbf{G}$ from which they can extract a set of reference values $\mathbf{R}$. Note that for phase 2 of our approach, as described in Sect. 4.2, all DOs must use the same set of reference values, $\mathbf{R}$. Therefore this set $\mathbf{R}$ is either generated in the same way by all DOs, or alternatively one DO generates $\mathbf{R}$ and distributes it to all other DOs.

As detailed in Algo. 1, the reference value selection process aims to find string values that are all different to each other according to a given similarity threshold. In other words no pair of selected reference values has an approximate string similarity above a similarity threshold, s_t, according to the used similarity function $sim()$. The first phase (Algo. 1) consists of the following steps:

1. The reference value set, $\mathbf{R}$, is initialized and the first value, g, from the global database, $\mathbf{G}$, is added to the empty set $\mathbf{R}$ (line 1 to 4).
2. Using the similarity function $sim()$, we then compare all other values $g \in \mathbf{G}$ with each previously selected reference value $r \in \mathbf{R}$, and we keep track of the maximum similarity, s_{max}, between g and any $r \in \mathbf{R}$ (line 6 to 9).
3. If the maximum similarity s_{max} between g and any $r \in \mathbf{R}$ is lower than the threshold s_t, then g is different enough from all so far selected reference values and is therefore added to $\mathbf{R}$ (lines 10 to 12). Steps 2 and 3 are repeated until all global values $g \in \mathbf{G}$ have been processed.

Algorithm 1: ***Selecting reference Values (Phase 1)***

Input:
- **G**: Publicly available global data set
- s_t: Similarity threshold
- $sim()$: Approximate string similarity function

Output:
- **R**: Reference value set

```
1:  R = { }                          // Initialize the reference value set
2:  for g ∈ G do:                    // Loop over all values in the global data set
3:    if R == { } do:                // Check if the reference set is empty
4:      R = R ∪ {g}                  // Add the selected first global value to the reference set
5:    else:
6:      s_max = 0                    // Initialize the maximum similarity value
7:      for r ∈ R do:                // Loop over all so far selected reference values
8:        s = sim(g, r)              // Calculate similarity between the global value and the reference value
9:        s_max = max(s_max, s)      // Get the maximum similarity
10:     if s_max ≤ s_t do:           // Check maximum similarity is less than the threshold
11:       R = R ∪ {g}                // Add global value to reference value set
12: return R
```

4.2 Phase 2: Reference Value based BLIP Bloom Filter Hardening

In the second phase of our approach each DO first encodes the records in its own database **V** into BFs, where these BFs are then hardened using a reference based BLIP approach, as detailed in Algo. 2. As described below, the BLIP based randomization of BFs using reference values can employ one or more reference values for a given record value v, where these reference values are the k most similar values in **R** (from Algo. 1). The idea of RBBF is that two similar record values, v_i and v_j, from **V** will likely have similar sets of reference values, $\mathbf{r}_{v_i}$ and $\mathbf{r}_{v_j}$, and when using these reference values as random seeds means that similar BLIP based randomization will be applied for v_i and v_j.

We harden a basic BF $\mathbf{b}_q$ for the q-gram set of a record value v for each of the k reference values for v, and concatenate all hardened BFs into one final BF $\mathbf{b}_v$ for v that is of length $k \times l$, where l is the length of the original BF $\mathbf{b}_q$. A final random permutation of all BFs $\mathbf{B}_v$ (agreed by all DOs) ensures an external attacker cannot identify the bit positions of an individual hardened BF generated using a certain reference value (which is unknown to an attacker). The second phase (Algo. 2) consists of the following steps:

1. In lines 1 and 2 the set of BLIP hardened BFs **B** is initialized first, as is a list of permuted bit positions **p** that will be used to permute all generated BFs in the same way. This list **p** basically contains all bit positions from 1 to $k \times l$ (the length of the final hardened BFs) randomly permuted.
2. The main loop (from line 3) iterates over all record values $v \in \mathbf{V}$, where in line 4 a value v is converted into its q-gram set **q** based on the set **A** of attributes to be encoded into BFs, and length of q-grams q. These q-gram sets are then encoded into a basic BF $\mathbf{b}_q$ (line 5).
3. In line 6, the k most similar reference values to v are identified from the reference values set **R** (as generated by Algo. 1) as the set $\mathbf{r}_v$.
4. We then initialize an empty BF $\mathbf{b}_v$ for record value v (line 7), and loop over the selected reference values $r \in \mathbf{r}_v$ in line 8. We use each reference value r as the random seed for a pseudo-random number generator (PRNG) in line

Algorithm 2: ***Reference value based BLIP Bloom Filter (RBBF) hardening (Phase 2)***

Input:
- $\mathbf{V}$: Record value set
- $\mathbf{R}$: Reference value set
- $\mathbf{H}$: Hash function set
- $\mathbf{A}$: Attribute value set
- k: Number of reference values per BF
- q: Q-gram length
- l: BF Length
- f: BLIP flip probability
- b_m: Blip method, either *ala* [2] or *rap* [10, 16]
- $sim()$: Approximate string similarity function

Output:
- $\mathbf{B}$: Set of RBBF encoded values from $\mathbf{V}$

1: $\mathbf{B} = \{\}$ // Initialize the set of RBBF encoded values
2: $\mathbf{p} = genBitPosPermList(l \times k)$ // Generate a list of permuted bit positions
3: **for** $v \in \mathbf{V}$ **do:** // Loop over all records
4: $\quad\mathbf{q} = genQgramSet(v, \mathbf{A}, q)$ // Generate q-gram set for the record value v
5: $\quad\mathbf{b}_q = genBF(\mathbf{q}, \mathbf{H}, l)$ // Generate basic Bloom filter for q-gram set $\mathbf{q}$
6: $\quad\mathbf{r}_v = getMostSimRefValSet(\mathbf{R}, v, sim, k)$ // Get the k reference values most similar to v
7: $\quad\mathbf{b}_v = [\,]$ // Initialize an empty Bloom filter for record v
8: $\quad$**for** $r \in \mathbf{r}_v$ **do**: // Loop over references values for record value v
9: $\quad\quad setRandomGeneratorSeed(r)$ // Initialize the PRNG
10: $\quad\quad$**if** $b_m == ala$ **then**: // Apply Eq. 1
11: $\quad\quad\quad\mathbf{b}_r = applyAlaBLIP(f, \mathbf{b}_q)$
12: $\quad\quad$**else**: // Apply *rap* (RAPPOR) BLIP method, Eq. 2
13: $\quad\quad\quad\mathbf{b}_r = applyRapBLIP(f, \mathbf{b}_q)$
14: $\quad\quad\mathbf{b}_v = concatenateBF(\mathbf{b}_v, \mathbf{b}_r)$ // Append to final hardened Bloom filter for record v
15: $\quad\mathbf{b}_v = permuteBF(\mathbf{b}_v, \mathbf{p})$ // Permute the final Bloom filter for record v
16: $\quad\mathbf{B}[v] = \mathbf{b}_v$ // Add RBBF hardened Bloom filter to the output set
17: return $\mathbf{B}$

9, and then we apply the selected BLIP method (using one of Eq. (1) or (2)) and the flip probability, f (in lines 10 to 13).
5. The resulting BLIP hardened BF $\mathbf{b}_r$ is then appended (concatenated) to the end of the record BF $\mathbf{b}_v$ in line 14.
6. A final permutation of $\mathbf{b}_v$ in line 15 ensures an attacker cannot identify the individual BFs that were BLIP hardened with a certain reference value. The BF $\mathbf{b}_v$ is then inserted into the list of all BFs $\mathbf{B}$ for record v in line 16.

4.3 Complexity and Linkage Quality Analysis

We now analyze our approach in terms of its complexity and linkage quality.

Complexity: The computational complexity of Algo. 1 depends upon the size of $\mathbf{G}$ as well as the similarity threshold s_t. If s_t is set to a high value then more values in $\mathbf{G}$ are added into $\mathbf{R}$. In the worst case, if $s_t = 1.0$ (assuming the similarity function returns a normalized value $0 \leq sim() \leq 1$) then all values in $\mathbf{G}$ will be added into $\mathbf{R}$ leading to a complexity of Algo. 1 of $O(|\mathbf{G}|^2)$.

The main loop in Algo. 2 iterates over all record values in $\mathbf{V}$ leading to a complexity of $O(|\mathbf{V}|)$. Generating the q-gram set $\mathbf{q}$ and the basic BF $\mathbf{b}_q$ in lines 4 and 5 are of complexity $O(Q \cdot |\mathbf{H}|)$ where we assume Q is the average number of q-grams in a value $v \in \mathbf{V}$, and $|\mathbf{H}|$ is the number of hash functions used to encode q-grams into BFs. Finding the k most similar reference values to a given $v \in \mathbf{V}$ from $\mathbf{R}$ requires $|\mathbf{R}|$ similarity calculations. The BLIP randomization in lines 8 to 14 for the selected k reference values has a complexity of $O(k \cdot l)$ where l is the length of the original BF. Finally, the permutation of the concatenated BF $\mathbf{B}_v$ also has a complexity of $O(k \cdot l)$ as a loop over all bit positions is required. Overall, the complexity of Algo. 2 is $O(|\mathbf{V}| \cdot (Q \cdot |\mathbf{H}| + |\mathbf{R}| + 2 \cdot k \cdot l)$.

Linkage Quality: The two main parameters of our approach that will affect the final linkage quality (besides the quality of the input data and the general parameters used for BF encoding and BLIP randomization) are the similarity threshold, s_t, in Algo. 1 and the number of reference values, k, in Algo. 2.

If a lower s_t is used then the set of reference values $\mathbf{R}$ will be smaller. Hence it will be more likely that two dissimilar record values will have the same value(s) in $\mathbf{R}$ and thus the same BLIP randomization will be applied on their BFs. This potentially lowers the precision of linkage quality because it will lead to an increased BF similarity if the same bit positions are flipped to 1-bits. A higher s_t leads $\mathbf{R}$ to contain more values and thus a higher likelihood that dissimilar record values will have different reference values leading to different BLIP randomization. Therefore, a higher s_t should result in higher linkage quality.

When using more reference values, k, per record value then the linkage quality will likely increase because there is a higher chance that similar record values share the same reference value(s), leading to similar BLIP randomization. On the other hand, when using less reference values it is more likely that dissimilar record values share the same reference value(s) which can lower linkage quality.

5 Experimental Evaluation and Discussion

We evaluated our proposed RBBF hardening approach using the North Carolina Voter Registration (NCVR) database (see: https://dl.ncsbe.gov), where we use a subset of 224,073 records as the global database $\mathbf{G}$ from where we extracted reference values. Using stratified sampling we identified 1,000 record pairs where we had 100 pairs in each of the ten similarity intervals $[0.0, 0.1)$, $[0.1, 0.2)$, ..., $[0.9, 1.0)$. We encoded different attribute combinations into BFs: (1) first name, (2) first and last names, and (3) first, last, street and town names. We set the similarity threshold in Algo. 1 to $s_t = [0.4, 0.6, 0.8]$, and the number of reference values in Algo. 2 to $k = [1, 3]$. We converted attribute values into q-grams using $q = 2$ and encoded them into BFs of length $l = 1{,}000$ using different numbers of hash functions, and set the BLIP flip probabilities $f = [0.01, 0.05, 0.1]$.

We implemented all approaches using Python 2.7 and ran experiments on a server with 2.4 GHz CPUs running Ubuntu 16.04. We compared RBBF with BFs without any hardening (No-BLIP) and the two BLIP approaches by Alaggan et al. [2] (BLIP-A), and Schnell and Borgs [16] (BLIP-S). We named RBBF based on Eq. (1) and Eq. (2) as RBBF-A and RBBF-S, respectively.

In the evaluation we compared Dice similarities, as discussed in Sect. 3, calculated between q-gram sets [5] with Dice similarities calculated between BFs. As Fig. 3 shows, for a pair of records we assumed the q-gram Dice similarity s_Q to be the true similarity. For a given similarity threshold t we then classified the corresponding BF pair with its Dice similarity, s_B, as a true positive (TP) if both $s_Q \geq t$ and $s_B \geq t$, a false negative (FN) if $s_Q \geq t$ and $s_B < t$, a false positive (FP) if $s_Q < t$ and $s_B \geq t$, and a true negative (TN) if $s_Q < t$ and $s_B < t$. We calculated precision as $P = TP/(TP + FP)$ and recall as $R = TP/(TP + FN)$, We do not present F-measure results given recent research [11].

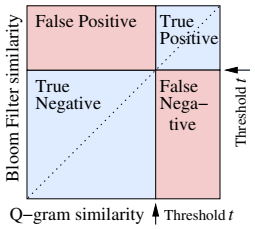

Fig. 3: Calculation of precision and recall based on q-gram and Bloom filter Dice similarities.

Table 1: Number of reference values generated for different attribute combinations using different values for the similarity threshold s_t.

Attribute combinations	$s_t = 0.4$	$s_t = 0.6$	$s_t = 0.8$
First name (FN)	1,217	5,949	14,442
Last name (LN)	2,375	12,943	29,682
Street (ST)	5,084	43,268	119,601
Town names (TN)	280	583	720
FN and LN	4,391	40,725	154,952
FN, LN and ST	16,891	184,606	219,926
FN, LN and TN	2,441	51,793	180,513
FN, LN, ST, and TN	6,027	115,000	216,497

Table 2: Average run times (in seconds) for BLIP and RBBF for different attribute combinations, numbers of reference values, k, and flip probabilities, f.

Attribute combinations	$f = 0.01$				$f = 0.1$			
	BLIP	$k = 1$	$k = 3$	$k = 6$	BLIP	$k = 1$	$k = 3$	$k = 6$
FN	0.159	0.163	0.482	0.968	0.165	0.169	0.511	1.007
FN and LN	0.316	0.322	0.945	1.867	0.321	0.328	0.980	1.954
FN, LN, ST, and TN	0.308	0.316	0.935	1.869	0.323	0.331	0.985	1.957

In Table 1 we show the number of reference values generated by Algo. 1 for different attribute combinations and similarity threshold values, s_t. As can be seen, higher values of s_t resulted in more reference values in $\mathbf{R}$. Furthermore, attributes (or combinations) with more unique values ended up with more reference values, which can lead to better linkage quality of RBBF hardened BFs.

However, the computational requirements of RBBF increase with more reference values, as we discussed in Sect. 4.3. Table 2 shows average run times for BLIP (averaged for BLIP-A and BLIP-S) and RBBF. As can be seen, as expected an increase of k led to increased run times, as did longer encoded q-gram sets. However the flip probability, f, did not seem to affect run times.

In Tables 3 to 5 we show precision and recall results (calculated as described above) for three selected attribute combinations. Due to limited space we only show results where the number of reference values is $k = 3$ and the reference similarity value is $s_t = 0.8$ because these settings gave the best results for all attribute combinations. We used a number of hash functions appropriate to the length of the q-grams sets that needed to be encoded into BFs [8, 15].

As shown in Tables 3 to 5, and as expected, without hardening the BFs (No-BLIP) Dice similarities were very close to the q-gram Dice similarities. While this will result in good linkage quality the known vulnerability to cryptanalysis attacks of not hardened BFs makes basic BF encoding not suitable for secure PPRL. As can be seen from these results, standard BLIP (BLIP-A and BLIP-S) led to very low precision and recall values of 0.0 with higher flip probabilities, while even with higher flip probabilities our RBBF approach achieved results of

Table 3: Precision and recall for attribute first name with 40 hash functions used for Bloom filter encoding. The best results for each f and t setting are shown in bold.

Method	Flip probability (f)	$t = 0.7$		$t = 0.8$		$t = 0.9$	
		Prec	Reca	Prec	Reca	Prec	Reca
No-BLIP	-	0.711	1.0	0.737	1.0	1.0	1.0
BLIP-A	0.01	**0.891**	**1.0**	0.953	0.976	1.0	0.65
BLIP-S	0.01	0.82	1.0	**0.955**	**1.0**	1.0	0.75
RBBF-A	0.01	0.886	1.0	0.883	0.964	1.0	0.725
RBBF-S	0.01	0.825	1.0	0.841	1.0	**1.0**	**0.85**
BLIP-A	0.05	1.0	0.284	1.0	0.167	0.0	0.0
BLIP-S	0.05	1.0	0.658	1.0	0.5	1.0	0.1
RBBF-A	0.05	0.959	0.39	0.778	0.25	1.0	0.325
RBBF-S	0.05	**0.963**	**0.748**	**0.871**	**0.595**	**1.0**	**0.4**
BLIP-A	0.1	1.0	0.013	0.0	0.0	0.0	0.0
BLIP-S	0.1	1.0	0.297	1.0	0.167	0.0	0.0
RBBF-A	0.1	0.889	0.152	0.728	0.226	1.0	0.3
RBBF-S	0.1	**0.962**	**0.432**	**0.778**	**0.333**	**1.0**	**0.35**

high precision while recall suffered for certain parameter settings and attribute combinations. As more attributes were encoded into BFs both precision and recall decreased especially with higher Dice similarity thresholds because the BLIP and RBBF randomization mechanisms led to lower Dice similarities.

Overall the results shown in Tables 3 to 5 also indicate that the RAPPOR [10, 16] based BLIP hardening approach from Eq. (2) seemed to outperform the approach proposed by Alaggan [2] from Eq. (1). Our proposed RBBF approach outperformed both standard BLIP approaches with regard to linkage quality for most parameter settings and attribute combinations.

In Table 6 we show re-identification results using a recently proposed frequency-based cryptanalysis attack [6], showing exact one-to-one, one-to-many, wrong and no re-identification percentages for the top 100 most frequent first names. As can be seen from these results, RBBF slightly improved privacy compared to not hardened BF encoding. Interestingly, the attack was still able to correctly re-identify 100% of all first names in a one-to-many manner in BLIP. This indicates that standard BLIP might not be as secure as originally hoped, and further research is required to investigate these results.

6 Conclusion

We have presented and improved the BLoom-and-flIP (BLIP) hardening technique for Bloom filter encoding for privacy-preserving record linkage. Our approach selects reference values from a large publicly available database and uses these values to modify the BLIP approach such that similar record values are randomized in a similar way. Our results on a real voter database showed that our approach is able to outperform standard BLIP approaches [2, 10, 15] while ensuring the hardened Bloom filters are secure with regard to attacks. In future

Table 4: Precision and recall for attributes first name and last name with 30 hash functions used for Bloom filter encoding. Best results are shown in bold.

Method	Flip probability (f)	$t = 0.7$		$t = 0.8$		$t = 0.9$	
		Prec	Reca	Prec	Reca	Prec	Reca
No-BLIP	-	0.61	1.0	0.719	1.0	0.189	1.0
BLIP-A	0.01	**0.655**	**1.0**	**0.866**	**1.0**	**0.6**	**0.857**
BLIP-S	0.01	0.636	1.0	0.804	1.0	0.368	1.0
RBBF-A	0.01	0.653	1.0	0.834	1.0	0.393	0.929
RBBF-S	0.01	0.634	1.0	0.794	1.0	0.275	1.0
BLIP-A	0.05	**0.91**	**0.905**	1.0	0.203	0.0	0.0
BLIP-S	0.05	0.753	0.995	0.979	0.748	0.0	0.0
RBBF-A	0.05	0.892	0.917	0.962	0.5	**0.518**	**0.5**
RBBF-S	0.05	0.733	0.998	**0.94**	**0.89**	0.383	0.5
BLIP-A	0.1	1.0	0.19	0.0	0.0	0.0	0.0
BLIP-S	0.1	0.909	0.896	1.0	0.203	0.0	0.0
RBBF-A	0.1	0.989	0.441	0.942	0.309	**0.433**	**0.5**
RBBF-S	0.1	**0.885**	**0.934**	**0.911**	**0.537**	0.35	0.5

work we aim to investigate linkage quality and privacy of RBBF on different data sets, develop approaches to calculate the optimal flip probability for RBBF to minimize the number of false positives while providing enough privacy, and improve the privacy of RBBF to avoid re-identification by adding randomness into the reference value selection process based on q-gram frequencies.

References

1. Alaggan, M., Cunche, M., Gambs, S.: Privacy-preserving wi-fi analytics. PET 2018(2), 4–26 (2018)
2. Alaggan, M., Gambs, S., Kermarrec, A.M.: BLIP: non-interactive differentially-private similarity computation on Bloom filters. In: SSS. pp. 202–216 (2012)
3. Bloom, B.: Space/time trade-offs in hash coding with allowable errors. Communications of the ACM 13(7), 422–426 (1970)
4. Boyd, J., Randall, S., Ferrante, A.: Application of privacy-preserving techniques in operational record linkage centres. In: Med Data Privacy Handbook (2015)
5. Christen, P.: Data matching – Concepts and techniques for record linkage, entity resolution, and duplicate detection. Springer (2012)
6. Christen, P., Schnell, R., Vatsalan, D., Ranbaduge, T.: Efficient cryptanalysis of Bloom filters for privacy-preserving record linkage. In: PAKDD (2017)
7. Christen, P., Vidanage, A., Ranbaduge, T., Schnell, R.: Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In: PAKDD (2018)
8. Durham, E., Kantarcioglu, M., Xue, Y., Toth, C., Kuzu, M., Malin, B.: Composite Bloom filters for secure record linkage. IEEE TKDE 26(12) (2014)
9. Dwork, C.: Differential privacy. ICALP pp. 1–12 (2006)
10. Erlingsson, Ú., Pihur, V., Korolova, A.: Rappor: Randomized aggregatable privacy-preserving ordinal response. In: ACM SIGSAC (2014)
11. Hand, D., Christen, P.: A note on using the F-measure for evaluating record linkage algorithms. Statistics and Computing 28(3), 539–547 (2018)

Table 5: Precision and recall for attributes first and last names, street, and town name with 20 hash functions used for Bloom filter encoding. Best results are shown in bold.

Method	Flip probability (f)	$t = 0.7$		$t = 0.8$		$t = 0.9$	
		Prec	Reca	Prec	Reca	Prec	Reca
No-BLIP	-	0.46	1.0	0.421	1.0	0.155	1.0
BLIP-A	0.01	0.471	1.0	**0.444**	**1.0**	**0.254**	**1.0**
BLIP-S	0.01	0.467	1.0	0.438	1.0	0.195	1.0
RBBF-A	0.01	**0.474**	**1.0**	0.442	1.0	0.204	1.0
RBBF-S	0.01	0.466	1.0	0.431	1.0	0.175	1.0
BLIP-A	0.05	**0.547**	**1.0**	**0.679**	**0.984**	1.0	0.067
BLIP-S	0.05	0.488	1.0	0.52	1.0	**1.0**	**0.733**
RBBF-A	0.05	**0.547**	**1.0**	0.626	0.996	0.289	0.567
RBBF-S	0.05	0.489	1.0	0.475	1.0	0.278	0.9
BLIP-A	0.1	**0.741**	**0.992**	1.0	0.071	0.0	0.0
BLIP-S	0.1	0.546	1.0	**0.679**	**0.984**	1.0	0.067
RBBF-A	0.1	0.71	0.998	0.705	0.567	**0.337**	**0.533**
RBBF-S	0.1	0.54	1.0	0.593	0.996	0.236	0.567

Table 6: Re-identification results for a frequency based attack [6] on first name values with 40 hash functions used for Bloom filter encoding and different flip probabilities.

	No BLIP	$f = 0.01$				$f = 0.1$			
		BLIP-A	BLIP-S	RBBF-A	RBBF-S	BLIP-A	BLIP-S	RBBF-A	RBBF-S
1-1 Corr %	0	0	0	0	0	0	0	0	0
1-Many Corr %	9	100	0	8	8	100	100	8	8
Wrong %	2	0	2	2	2	0	0	2	2
No %	89	0	98	90	90	0	0	90	90

12. Kroll, M., Steinmetzer, S.: Automated cryptanalysis of Bloom filter encryptions of databases with several personal identifiers. In: BIOSTEC (2015)
13. Kuzu, M., Kantarcioglu, M., Durham, E., Malin, B.: A constraint satisfaction cryptanalysis of Bloom filters in private record linkage. In: PET (2011)
14. Pang, C., Gu, L., Hansen, D., Maeder, A.: Privacy-preserving fuzzy matching using a public reference table. Intelligent Patient Management pp. 71–89 (2009)
15. Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using Bloom filters. BMC Med Inform Decis Mak 9(1) (2009)
16. Schnell, R., Borgs, C.: Randomized response and balanced Bloom filters for privacy preserving record linkage. In: ICDMW DINA (2016)
17. Schnell, R.: Privacy-preserving record linkage. In: Harron, K., Goldstein, H., Dibben, C. (eds.) Methodological Developments in Data Linkage (2015)
18. Schnell, R., Borgs, C.: XOR-folding for Bloom filter-based encryptions for privacy-preserving record linkage. working paper, German Record Linkage Center (2016)
19. Schnell, R., Rukasz, D., Borgs, C., Brumme, S., et al.: R PPRL toolbox. https://cran.r-project.org/web/packages/PPRL/ (2018)
20. Vatsalan, D., Sehili, Z., Christen, P., Rahm, E.: Privacy-preserving record linkage for Big Data: Current approaches and research challenges. In: Handbook of Big Data Technologies, pp. 851–895. Springer (2017)
21. Vatsalan, D., Christen, P., O'Keefe, C.M., Verykios, V.: An evaluation framework for privacy-preserving record linkage. JPC 6(1) (2014)