Encoding hierarchical classification codes for Privacy-preserving Record Linkage using Bloom filters

Rainer Schnell[1] and Christian Borgs[1]

German Record Linkage Center, University of Duisburg-Essen, Lotharstr. 65, 47057 Duisburg, Germany
{rainer.schnell,christian.borgs}@uni-due.de

Abstract. Hierarchical classification codes are widely used in many scientific fields. Such codes might reveal sensitive personal information, for example medical conditions or occupations. This paper introduces a new encoding technique for encrypting sensitive codes, which preserves the hierarchical similarity of the codes. The encoding was developed for the use of hierarchical codes in Privacy-preserving Record Linkage (PPRL). The technique is demonstrated with real-world survey data containing occupational codes (ISCO codes). After describing the construction and its similarity preserving properties, Hierarchy Preserving Bloom Filters (HPBF) are compared with positional q-grams and standard Bloom filters in a PPRL context. The method presented here is similarity preserving for hierarchies, privacy-preserving and will increase linkage quality when used in Bloom filter-based PPRL.

Keywords: Positional Bloom filters · Hierarchy Preserving Bloom filters · entity resolution · ISCO codes · hierarchical similarity · PPRL

1 Introduction

In many research settings, categorising elements with hierarchical categorical schemes is daily practice. Examples include taxonomies in biology, the classification of occupations [18], accident statistics [10, 6], entity resolution of corporations using NACE-codes [20], or classifying diseases or causes of death with the ICD.

In data science, matching identifiers of different records (Record Linkage) is a central challenge [2]. In most applications, linkage is done on clear text identifiers. However, numerical attributes, dates and geographical information might also be used for linking. Hierarchical codes can be used for directly linking data as well [9].

For many applications, such hierarchical codes often relate to individuals, i.e. job classifications. Therefore, these codes can be sensitive, requiring special data protection, as would be the case for recording diseases of a person. Using these codes in a data linkage scenario would require encrypting them, which, while retaining discriminatory power, would lead to a loss of the hierarchical properties

of the codes. However, a proper encoding technique for hierarchical codes should: (1) preserve the similarity of different codes if they agree on higher levels and disagree only on lower level details, (2) improve linkage quality by using the information contained in the hierarchy.

To the best of our knowledge, no other privacy-preserving method for the encryption of hierarchical codes has been published so far.Therefore, in this paper we suggest a new encoding technique for hierarchical codes. The new method is tested for use as an additional identifier in Privacy-preserving Record Linkage (PPRL) settings.

2 Methods

The newly suggested method for privacy-preserving hierarchy will be based on Bloom filters, which are commonly used for linking data privately [26, 5].

2.1 Bloom filters

Bloom filters [1] (BF) are binary vectors with a length of l bits in which information is stored. They were first devised to rapidly check set membership [1], but have been used in other applications as well. For Privacy-preserving Record Linkage, they were first suggested [24] to be used by splitting strings into subsets of the length q (q-grams or n-grams). These q-grams determine a number k of bit positions $B_i \in \{1, \ldots, l\}$ to be set to one in a bit vector initially consisting of l zero bits. Currently, it is recommended to randomly select these bit positions by using the input q-gram together with a password as a seed for a PRNG that selects k random bit positions that are set to a value of one [23]. An example is shown in Figure 1.

The attractive main property of Bloom filters is that they can be used to encrypt strings in a similarity-preserving way. One method to compute the similarity of two sets A and B of bigrams is the Dice coefficient, which is calculated as the doubled intersect of the two sets divided by the number of elements in both sets:

$$D = \frac{2|A \cap B|}{|A| + |B|}. \tag{1}$$

As can be seen in Figure 1, the names Sahra and Sarah share three out of four bigrams (subsets of $q = 2$). This gives an unencrypted clear-text Dice similarity of $D = \frac{2*3}{4+4} = 0.75$.

For Bloom filters, the Dice similarity can be computed by comparing the sets of bit positions of two Bloom filters. Here, the Dice similarity of the Bloom filters is very close to the unencrypted bigram similarity, as both Bloom filters have 6 resp. 7 bits set to one, while sharing 5 bit positions. This gives a Dice coefficient for the encrypted names of $D_{BF} = \frac{2*5}{7+6} \approx 0.77$.

Bloom filters have been used for encoding numerical attributes, dates [29] and geographical information [7] as well. Up to now, no encoding technique

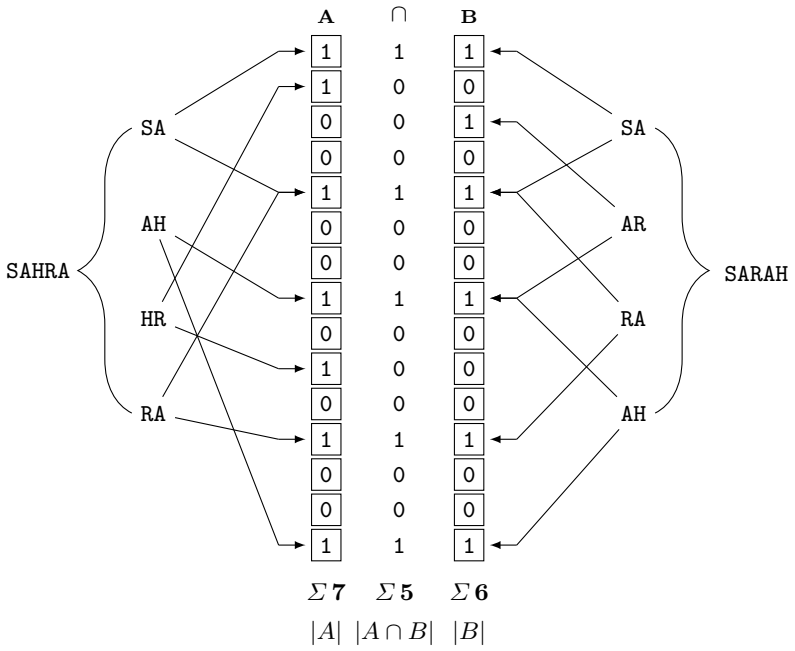

Fig. 1. Bloom filters constructed for two similar names using a length of $l = 15$ bits and $k = 2$ hash functions for each bigram.

for encoding hierarchical codes into Bloom filters has been suggested. A naive approach would use unigrams ($q = 1$). Of course, this approach would result in the loss of all hierarchical information, so using regular Bloom filters for hierarchical codes is not advised.

2.2 Positional BFs (PBFs)

In hierarchical codes, the positions of the code matter. For many applications, the first code position is the top of the hierarchy and important for all following positions, as they change the meaning of all following codes. To the best of our knowledge, no encoding of hierarchical codes into Bloom filters has been suggested before. However, the literature mentions some encodings of positional information of q-grams in strings. The concatenation of an index and the q-gram at the index position was first proposed by [21] as positional q-grams. They have, for example, been used in conjunction with Bloom filters in genome searches [11]. For PPRL settings, it has been used before [2, 25]. The application of positional unigrams to hierarchical codes is straightforward: Taking the ISCO-88 code 3213 as an example, a Bloom filter using positional unigrams would hash the values 31, 22, 13, and 34 into the bit vector, where the second digit is the q-gram position.

In contrast, a standard Bloom filter would only map the elements 2, 1 and 3 of the ISCO code 3213 to the bit vector. Although positional unigrams identify

the position of a code element, it does not reflect the relative importance of the code positions, as the first code positions in hierarchical codes are usually more important than the last bits. Therefore, we expect that Record Linkage using positional q-grams yields better results than naive representations of hierarchical information, but will still omit information on the relative importance given by the index position.

2.3 Hierarchy Preserving Bloom filters (HPBFs)

As described in the last section, the code positions are crucial for preserving hierarchies, as the first position determines the meaning of all following codes. The same is true for all following positions. Since existing encoding methods ignore this information, a new method is proposed. It is shown in Figure 2. The new encoding is based on two modifications of the standard procedure.

First, the code is split into unigrams. The first code position remains as is. All following code positions will contain all unigrams of the previous code positions. This will give more weight to the first positions.

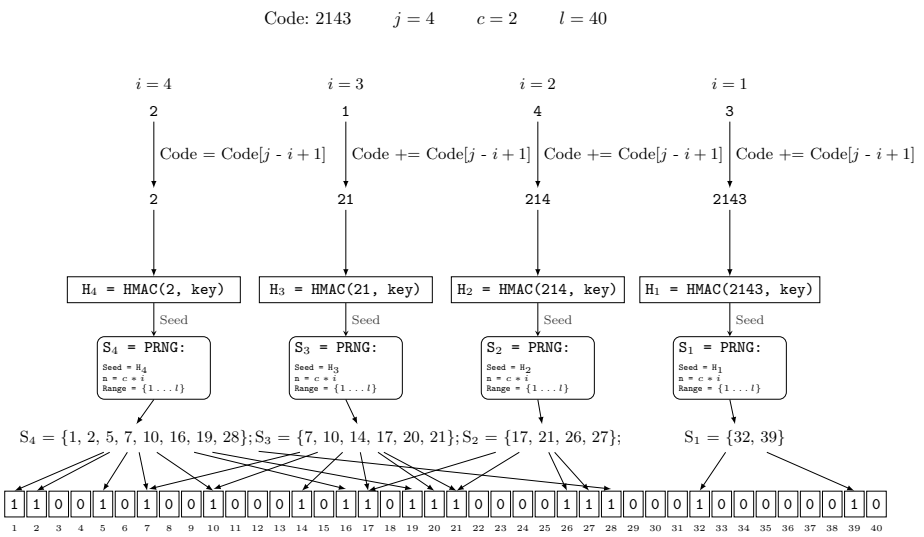

Fig. 2. Constructing Bloom filters for hierarchical codes from the ISCO 88-Code for electrical engineers (2143, code length $j = 4$). Bloom filter length is $l = 40$ with a stream length modifier of $c = 2$. This way, 16 bit positions are set to one. Four bit positions are set to one by more than one PRNG stream (S_i).

Second, the number of bit positions for coding is dependent on the position of the unigram within the code. To achieve this, the first positions receive more bits in the representation than the last positions. In the implementation shown here, the q-grams are hashed with an HMAC [15], such as SHA-3 [19], using a

secret key. The numeric representation of the resulting hash is used as a seed for a PRNG, which draws bit positions in the range from 1 to l. The number of bit positions depends on the modifier c and the position within the code.

Since the first positions are more important than the last positions, for a code of length j, the variable $i \in \{j, j-1, \ldots, 1\}$ is multiplied by c, giving $c * i$ elements to draw. In Figure 2, c is set to a value of 2, leading to eight bit positions to be drawn for the first code position, while the last code position sets only two bit positions to one. This way, the more important the code position in the hierarchy, the more bits are set to one.

The range of possible values for c is small since most codes are limited to three or four digits. This way, possible choices for c depend on the length and the frequency distribution of the actual codes in the space of all possible codes. Up to now, we have chosen c empirically. Finding an optimal value of c will require additional research.

Algorithm 1: HierarchyPreservingBloomFilters($input, pwd = 42, l = 512, c = 1$)

$split \leftarrow$ strsplit($input$)
$BF \leftarrow [0] * l$
local H
local S
for $i \leftarrow length(split)$ **downto** 1
do
- $Code \leftarrow Code + split[length(split) - i + 1]$
- $H[i] \leftarrow$ SHA2($Code, key = pwd$)
- **comment:** Use numeric representation of hash as seed for PRNG
- $S[i] \leftarrow$ PRNG($Seed = H[i], n = c * i, min = 1, max = l$)
- $BF[S[i]] \leftarrow 1$

return (BF)

The pseudocode for the suggested procedure is shown in Algorithm 1. We denote this encoding as Hierarchy Preserving Bloom Filters (HPBF) since the term Hierarchical Bloom Filters has been used for different data structures [14, 4, 17], which are not intended do preserve hierarchies in codes. To empirically test this encoding, three datasets were used.

2.4 Data

Each of the three datasets used in the evaluation is described briefly.

Synthetic data Using ICD11-codes for classifying diseases, a sample of $n = 20,000$ ICD11-codes was drawn. A second sample of $n = 14,000$ codes was generated. By chance, both codes will agree or disagree on several code positions.

PASS ISCO-Codes The PASS panel [28] is a longitudinal study on the effects of unemployment. A classification of the occupation is given by the ISCO-codes (ISCO-88).

Table 1. Exemplary jobs with their ISCO-88 major groups (M), sub-major groups (SM), minor groups (MI) and units (UN) as well as descriptions of them.

M	SM	MI	UN	Description
1				Legislators, senior officials and managers
		131		General managers
			1314	General managers in wholesale and retail trade
			1315	General managers of restaurants and hotels
3				Technicians and associate professionals
	34			Other associate professionals
		341		Finance and sales associate professionals
			3413	Estate agents

Exemplary ISCO codes can be seen in Table 1. The first positions are the major groups, where the largest differences between occupational groups are obvious. Every subsequent code position describes the occupation more precisely.

To study the reliability of the codes the occupation of each person was coded by two independent coding units. For our purpose here, we consider this as an example for the intended application of HPBFs: if the data collection has been done independently, the linkage between two datasets could be enhanced by using the encoded ISCO-codes. This allows a direct comparison of the true positive matches attained by HPBFs compared to exact matching and (positional) Bloom filters.

Synthetic data for PPRL Pairs of randomly selected ISCO-codes of the PASS study were randomly assigned to two real-life mortality datasets [25], for which a gold standard linkage solution existed. The personal information (first and last name and date of birth) were encrypted using CLKs with $k = 20$ hash functions and a length of $l = 1000$, while the ISCO-codes were encrypted as either standard Bloom filters, HPBFs or PBFs. These were then included in the CLKs. As the true match state was known, PPRL linkage quality using hierarchical information in different encodings can be evaluated.

2.5 Evaluation methods

First, evaluation methods relating to the hierarchy-preserving properties of the encryption methods are reviewed. Next, evaluating linkage quality in a PPRL setting is discussed.

Hierarchical recall and precision. To map a hierarchical code into a tree structure, code positions form the tree leaves, where each position determines a level. An example is shown in Figure 3.

To calculate precision and recall for a tree-based classification, a modification of the standard definitions of precision and recall is necessary [27]. This can best be explained by an example (see Figure 3). Here, the true value (dark green, denoted as Y_{labels}) is compared to a second classification (dark blue, denoted as $\hat{Y}_{labels}$). Let the true code be `ZBC`, while the second classification is the code `ZBD`.

The hierarchical precision is then calculated as the number of agreements on the labels ($\hat{Y}_{labels} \cap Y_{labels}$) divided by the number of labels given by a classifier ($\hat{Y}_{labels}$).

The hierarchical recall is calculated as the number of agreements on the labels ($\hat{Y}_{labels} \cap Y_{labels}$) divided by the number of labels given by the true label (Y_{labels}).

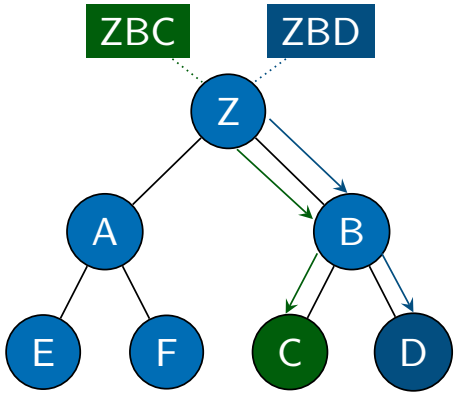

Fig. 3. Two example codes, Y_{labels} = `ZBC` (dark green) and $\hat{Y}_{labels}$ = `ZBD` (dark blue) and their resulting tree structure. The path for querying both codes is drawn with arrows. Both codes only differ on the final node, sharing two nodes (ancestors) on a higher hierarchical level. Note that codes containing `A`, `E` and `F` are used in the full classification, but not in the example codes `ZBC` and `ZBD`.

In Figure 3, the two exemplary codes (Y_{labels} = `ZBC` and $\hat{Y}_{labels}$ = `ZBD`) are compared by tracing their sub-tree within the full tree. The final nodes for both codes are C and D. The *ancestors* of a given node are all nodes on the path to the root node (`Z` in this case)[12]. In this example, the sets of ancestors would be $Y = \{Z, B, C\}$ and $\hat{Y} = \{Z, B, D\}$. For this particular set of codes, both hierarchical precision and hierarchical recall can be calculated as:

$$\text{Hierarchical Recall} = R_h = \frac{|\hat{Y}_{labels} \cap Y_{labels}|}{|Y_{labels}|}, \tag{2}$$

and

$$\text{Hierarchical Precision} = P_h = \frac{|\hat{Y}_{labels} \cap Y_{labels}|}{|\hat{Y}_{labels}|}, \tag{3}$$

with the mean of both serving as the F-Score:

$$F_h = \frac{1}{2}(P_h + R_h). \tag{4}$$

Both Y and $\hat{Y}$ agree on two ancestors ($\{Z, B\}$), giving a size of the elements in their intersect of $|\hat{Y}_{labels} \cap Y_{labels}| = 2$. Both have three ancestors ($|\hat{Y}_{labels}| = 3$ and $|Y_{labels}| = 3$). Therefore, $P_h = R_h = F_h = \frac{2}{3} \approx 0.667$.

Please note that hierarchical precision and recall require information on the actual position within a tree. Therefore, it can only be used if the hierarchical information is preserved in an encoding. In the following empirical study, we compare the pairwise similarity of hierarchical precision and recall in the clear-text with the pairwise Dice similarity of HPBFs.

Similarity by linkage category. If an encryption method is hierarchy-preserving, a full match of codes should yield a higher similarity than partial matches. In addition, a difference in the last characters of a code should result in a higher similarity than a difference in the first code positions, as these are usually more important for the code hierarchy. This idea is captured by the classification of partial matches by Klug et al. [13]. In their application, they used ICD codes and classified the type of agreement into six classes:

1. Full match (no code positions differ)
2. Subgroups differ (last two positions disagree)
3. Subgroups and fourth character differ (diagnostic subgroups differ)
4. Only the first two characters match (only diagnostic groups match)
5. Only the first character matches (only diagnostic chapter matches)
6. Full non-match (all code positions differ)

A full match should lead to similarities close to one, while a full non-match should give similarity values close to zero. Ideally, for all categories in between, the similarity values should not overlap, so that the range of similarity coefficients for each category is small. We compared Dice similarities of Bloom filters within each category given by the classification of Klug et al. [13].

Evaluation of linkage quality of PPRL methods using hierarchical codes. For linking all three Bloom filter-based PPRL methods, we used Multibit trees. Multibit trees were suggested for chemometrics by Kristensen et al. [16] and proposed for PPRL by Schnell [22].

The efficiency of Multibit trees for comparing Bloom filters is due to the fact that possible pairs below a pre-set similarity threshold are not evaluated. Therefore, Multibit trees are being used as an error-tolerant blocking method. The implementation of Multibit trees uses the Tanimoto similarity T as a similarity

measure. T is defined as number of bits set to 1 in both vectors A and B divided by the total number of bits set to 1 in A and B:

$$T(A, B) = \frac{\Sigma_i(A_i \wedge B_i)}{\Sigma_i(A_i \vee B_i)} \tag{5}$$

Lower thresholds will result in more pair comparisons and a higher number of false positive classifications. Conversely, the amount of true matches will increase as well.

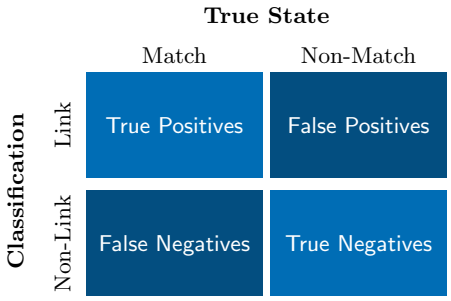

Fig. 4. Outcomes for linkage pairs by classification and true matching state.

For the empirical evaluation of the suggested method, we use the traditional evaluation criteria of precision and recall

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{6}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{7}$$

$$\overline{F} = \frac{1}{2}(\text{Recall} + \text{Precision}), \tag{8}$$

as given by Table 4 and the corresponding Equations 6 to 8. Following the critique by [8], we use the unweighted arithmetic mean of precision and recall ($\overline{F}$) instead of the harmonic mean (F-Score).

3 Results

First, results concerning the hierarchy-preserving properties of the encryptions is reported, before linkage quality in PPRL settings is addressed.

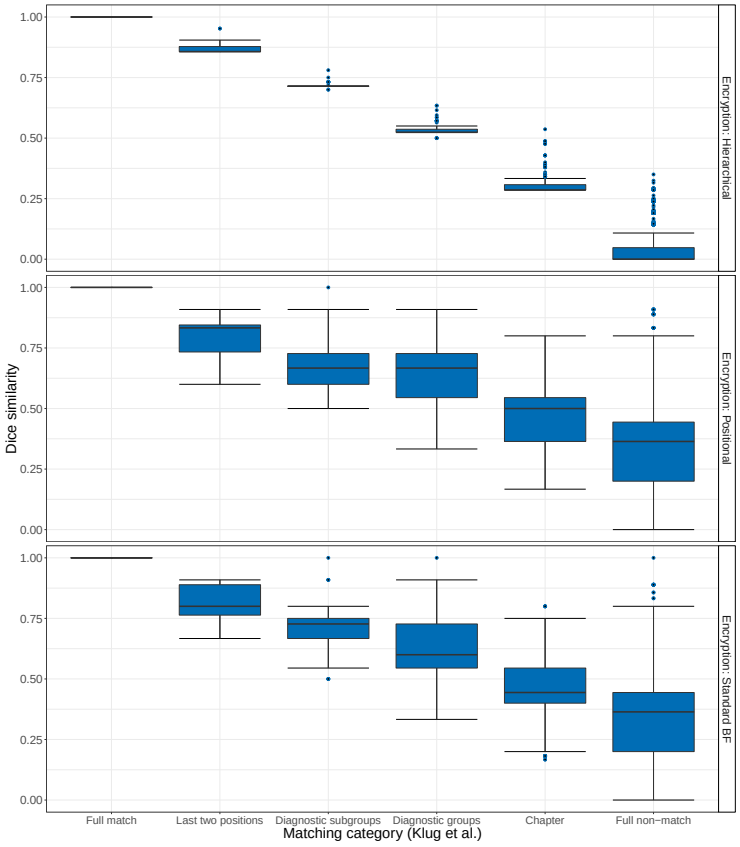

Fig. 5. Linkage categories and Dice similarity by method (HPBF with $c = 1$).

Similarity by linkage category. Hierarchy-preserving encryptions should lead to similar Dice coefficients for each linkage category (as defined by [13]; see Section 2.5). To test this, the pairwise Dice similarities of the Bloom filters were computed for each encryption method (HPBF, PBF and standard BF).

The results are shown in Figure 5, where the box plots for each category and encryption method are shown.

The HPBFs discriminate the categories very well and with little spread. In contrast, both the positional and standard BFs show a wide spread of Dice coefficients for all categories but the full match category. The plot demonstrates that Hierarchy Preserving Bloom filters have more discriminating power of encoded hierarchical codes than previous methods. To explore the properties of the HPBFs further, we examine the functional relationship between the Dice coefficient of pairs of Bloom filters and the corresponding hierarchical precision and recall of unencrypted codes.

Hierarchical precision and recall. For each pair of ISCO codes, the hierarchical precision and recall were computed [27]. Since ISCO occupational codes always have four characters, the number of ancestors will always be four. This way, $R_h = P_h$, which is why only hierarchical recall (R_h) will be reported here. If the encryption is hierarchy-preserving, the hierarchical recall should be a monotone function of the Dice similarity of the two encrypted codes. By comparing the ISCO-codes of the same person generated by two different coding units, this relationship is shown for all methods in Figure 6.

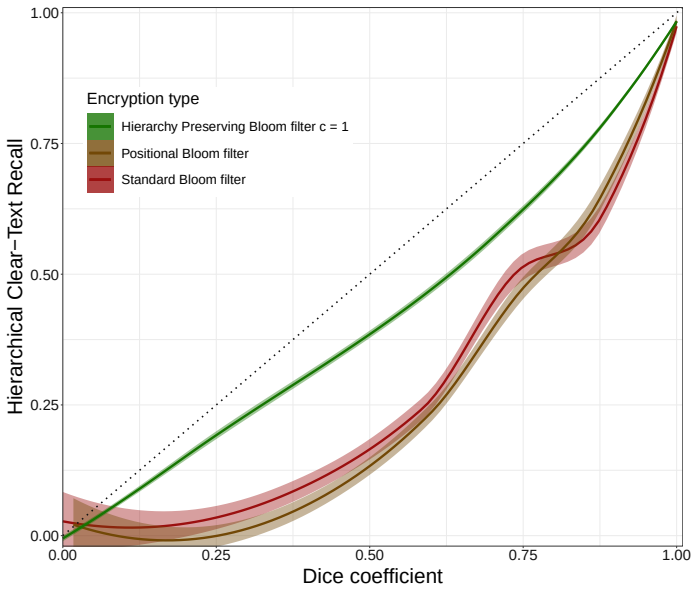

Fig. 6. Hierarchical recall plotted against the Dice similarity of the encrypted codes for Hierarchy Preserving Bloom filters (HPBF), positional Bloom filters (PBF) and Standard Bloom filters (BF). The lines shown are loess smoothers (based on $n = 5,033$ code pairs) with 2 standard errors (the shaded areas).

Given this dataset, standard and positional Bloom filters perform worse than Hierarchy Preserving Bloom filters. Furthermore, the standard errors are considerably larger at lower hierarchical recall and Dice coefficient values. In contrast, the HPBFs have smaller standard errors. Furthermore, the numeric value of the hierarchical recall is better approximated by the Dice coefficients of the HPBFs (the smoothed curve is much closer to the diagonal reference line).

3.1 Privacy-preserving Record Linkage (PPRL)

In a PPRL setting, using as many stable identifiers as possible is recommended [23]. However, hierarchical codes can be unstable, especially when relying on

humans to classify hierarchical codes. In the given dataset, two encoding units classified the persons' jobs independently. Of the resulting $n = 5,033$ ISCO code pairs, only $2,497$ matched exactly.

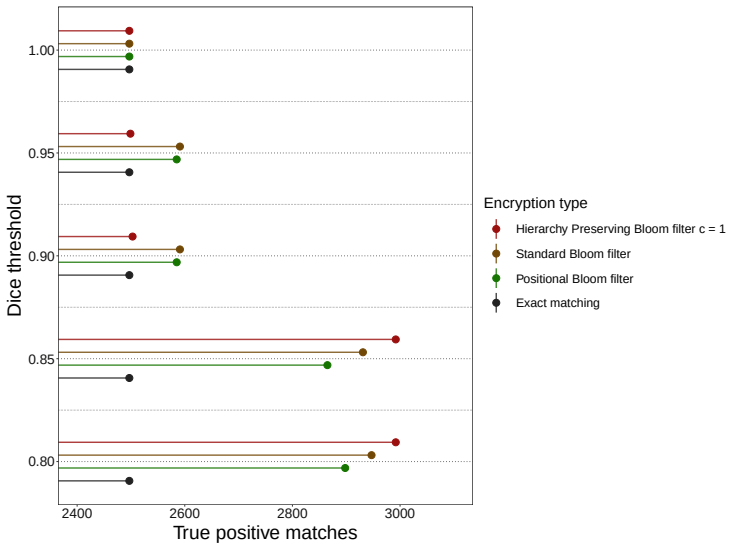

Fig. 7. True positive ISCO code matches by method and similarity threshold.

Increasing the number of matches by accepting differing codes at lower hierarchical levels (for example, 3122 and 3121: Computer equipment operators (3122) and computer assistants (3121)) is shown in Figure 7. Here, the number of true positive matches increases substantially when lowering the level of accepted minimal similarity. HPBFs with a stream length modifier of $c = 1$ show the highest number of true positives at all similarity thresholds below 0.90.

However, even allowing for errors yields only about $3,000$ matching code pairs. Obviously, using the ISCO code as single identifier, even when allowing for errors, is not sufficient for linking. Therefore, we studied the performance of HPBFs in a PPRL setting.

The bit vectors resulting from the standard, positional (PBF) and Hierarchy Preserving Bloom filter (HPBF) encryptions are inserted with an OR operation into standard CLKs (composite Bloom filters [23]) for names and dates of birth (with $k = 20$ hash functions and $l = 1000$ bits[1]). These CLKs are evaluated as described in Section 2.5.

The resulting mean of precision and recall is shown in Figure 8. Although the same amount of information is encoded by all three methods, the combination

[1] Above a certain minimal length, choices for l are arbitrary as long as k is adjusted accordingly.

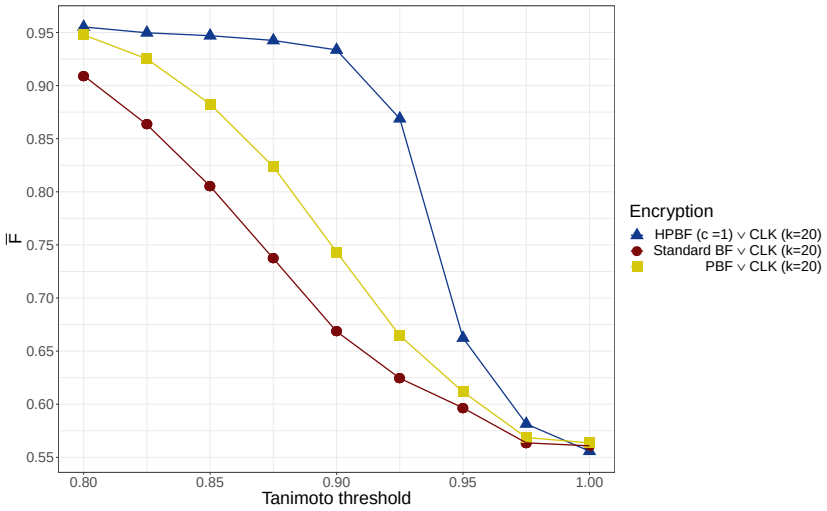

Fig. 8. Arithmetic mean of precision and recall ($\overline{F}$) of standard CLKs with $k = 20$ hash functions combined with ISCO codes encoded in Bloom filters (BF), positional Bloom filters (PBF) and Hierarchy Preserving Bloom filters (HPBF), linked using several Tanimoto thresholds.

of HBPFs with CLKs considerably improves the linkage quality compared to combining PBFs with CLKs and standard Bloom filters with CLKs.

4 Conclusion

In many research applications, codes representing a hierarchical relation are used. Preserving similarities of hierarchical codes with encrypted identifiers was shown to be possible with Hierarchy Preserving Bloom filters (HPBFs) presented in this paper. Using these encodings in Privacy-preserving Record Linkage (PPRL) settings, linkage quality will improve compared to previous methods for encoding hierarchical codes.

As has been shown by Christen et al. [3], frequency distributions of bit patterns in Bloom filters can be used as attack vectors in cryptographic attacks on Bloom filter encodings. Very frequent bit patterns as well as unique bit patterns might give additional identifying constraints for an attack. Therefore, a study on the cryptographic properties and attack methods, as well as options to prevent these attacks for Hierarchy Preserving Bloom filters (and other methods for encoding numerical and geographical information [29, 7]) is subject of ongoing research.

References

1. Bloom, B.H.: Space/Time Trade-offs in Hash Coding with Allowable Errors. Communications of the ACM **13**(7), 422–426 (1970)

2. Christen, P.: Data Matching – Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer, Berlin (2012)
3. Christen, P., Randaduge, T., Vidanage, A., Schnell, R.: Pattern-mining based cryptanalysis of Bloom filters for privacy-preserving record linkage. In: The 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD), June 3rd – 6th, 2018, Melbourne, Australia (2018)
4. Crainiceanu, A.: Bloofi: A hierarchical Bloom filter index with applications to distributed data provenance. In: Darmont, J., Pedersen, T.B. (eds.) Proceedings of the 2nd International Workshop on Cloud Intelligence (Cloud-I '13) Riva del Garda, Trento, Italy, August 26–26. ACM, New York (2013)
5. Dantas Pita, R., Pinto, C., Sena, S., Fiaccone, R., Amorim, L., Reis, S., Barreto, M., Denaxas, S., Barreto, M.: On the accuracy and scalability of probabilistic data linkage over the Brazilian 114 million cohort. IEEE Journal of Biomedical and Health Informatics **22**(2), 346–353 (2018)
6. European Commission, Eurostat: European Statistics on Accidents at Work (ESAW): Summary Methodology. Publications Office of the European Union, Luxembourg (2013)
7. Farrow, J.: Privacy preserving distance-comparable geohashing (2014), Vancouver, International Health Data Linkage Conference, 28–30 April
8. Hand, D., Christen, P.: A note on using the F-measure for evaluating record linkage algorithms. Statistics and Computing **28**(3), 539–547 (2018)
9. Hejblum, B.P., Weber, G.M., Liao, K.P., Palmer, N.P., Churchill, S., Shadick, N.A., Szolovits, P., Murphy, S.N., Kohane, I.S., Cai, T.: Probabilistic record linkage of de-identified research datasets with discrepancies using diagnosis codes. Scientific Data **6**, 180298 (2019)
10. Jacinto, C., Santos, F.P., Soares, C.G., Silva, S.A.: Assessing the coding reliability of work accidents statistical data: How coders make a difference. Journal of Safety Research **59**, 9–21 (2016)
11. Kerschbaum, F., Beck, M., Schönfeld, D.: Inference control for privacy-preserving genome matching. CoRR abs/1405.0205 (2014)
12. Kiritchenko, S., Matwin, S., Nock, R., Famili, A.F.: Learning and evaluation in the presence of class hierarchies: Application to text categorization. In: Conference of the Canadian Society for Computational Studies of Intelligence. pp. 395–406. Springer, Berlin (2006)
13. Klug, S.J., Bardehle, D., Ressing, M., Schmidtmann, I., Blettner, M.: Vergleich von ICD-Kodierungen zwischen Mortalitätsstatistik und studieninterner retrospektiver Nachkodierung. Gesundheitswesen **71**(4), 220–225 (2009)
14. Koloniari, G., Pitoura, E.: Bloom-based filters for hierarchical data. 5th Workshop on Distributed Data and Structures, Thessaloniki, 13 – 14 June 2003 (2003)
15. Krawczyk, H., Bellare, M., Canetti, R.: HMAC: Keyed-hashing for message authentication. Internet RFC 2104 (1997)
16. Kristensen, T.G., Nielsen, J., Pedersen, C.N.S.: A tree-based method for the rapid screening of chemical fingerprints. Algorithms for Molecular Biology **5**(1), 9–20 (2010)
17. Lillis, D., Breitinger, F., Scanlon, M.: Hierarchical Bloom filter trees for approximate matching. Journal of Digital Forensics, Security and Law **13**(1), 81–96 (2018)
18. McLean, D., van Tongeren, M., Richardson, L., Schlehofer, B., Villegas, R., Benke, G., Jarus-Hakak, A., Hours, M., Nadon, L., Samkange-Zeeb, F., Sleeuwenhoek, A., Cardis, E.: Evaluation of the quality and comparability of job coding across seven countries in the INTEROCC study. Occupational and Environmental Medicine **68**(Suppl 1), A61 (2011)

19. National Institute of Standards and Technology: Secure hash standard (SHS). FIPS PUB 180-4 (2012)
20. Peruzzi, M., Zachmann, G., Veugelers, R.: Remerge: Regression-based record linkage with an application to PATSTAT. Tech. Rep. 2014/10iii, Bruegel Working Paper, Brussels (2014)
21. Riseman, E.M., Hanson, A.R.: A contextual postprocessing system for error correction using binary n-grams. IEEE Transactions on Computers (5), 480–493 (1974)
22. Schnell, R.: An efficient privacy-preserving record linkage technique for administrative data and censuses. Journal of the International Association for Official Statistics **30**(3), 263–270 (2014)
23. Schnell, R.: Privacy-preserving record linkage. In: Harron, K., Goldstein, H., Dibben, C. (eds.) Methodological Developments in Data Linkage, pp. 201–225. Wiley (2016)
24. Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using Bloom filters. BMC Medical Informatics and Decision Making **9**(1), 41–52 (2009)
25. Schnell, R., Richter, A., Borgs, C.: A comparison of statistical linkage keys with Bloom filter-based encryptions for privacy-preserving record linkage using real-world mammography data. In: Proceedings of the 10$^{\text{th}}$ International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2017). pp. 276–283 (2017)
26. Smith, D.: Secure pseudonymisation for privacy-preserving probabilistic record linkage. Journal of Information Security and Applications **34**, 271–279 (2017)
27. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Information Processing & Management **45**(4), 427–437 (2009)
28. Trappmann, M., Beste, J., Bethmann, A., Müller, G.: The PASS panel survey after six waves. Journal for Labour Market Research **46**(4), 275–281 (2013)
29. Vatsalan, D., Christen, P.: Privacy-preserving matching of similar patients. Journal of Biomedical Informatics **59**, 285–298 (2016)