Instance-aware Evaluation of Sensitive Columns in Tabular Dataset[⋆]

Zheng Gong[1], Kechun Zhao[1], Hui Li[1][0000−0003−2382−6289], and Yingxue Wang[2]

[1] Xidian University, Xi'an, China
marcogong22@gmail.com,zkc0422@outlook.com,hli@xidian.edu.cn
[2] National Engineering Laboratory for Public Safety Risk Perception and Control by Big Data, Beijing, China wangyingxue@csdslab.net

Abstract. Fully discovering knowledge from big data has to publish and share corresponding datasets whenever required. However, the risk for privacy leakage, i.e., record re-identification through some released columns, in the datasets is a fatal problem that prevents these tasks. Therefore, evaluating the sensitivity for different attributes is a prerequisite for dataset desensitization and anonymization, after which datasets can be published and shared in a privacy-preserving way. However, automatically evaluating the sensitivity for attributes is challenging and remains an open problem. In this work, we present a novel-but-simple technique for quantifying the sensitivity in structural database. It automatically evaluates the risks for re-identification for different columns according to *Record-linkage Attack*. Under the support of our scheme, the output sensitivity for the same attribute in different instances of a relational schema varies. Moreover, our scheme can quantify the risks of the columns no matter the semantics of columns are known or not. We also empirically show that the proposed scheme is effective in dataset sensitivity governance comparing with baselines.

Keywords: data privacy · record-linkage attack · sensitivity · attribute · relational table.

1 Introduction

Exploiting the value behind big data can benefit many applications such as policy-making, interpersonal relationship discovery, etc. However, accomplishing the task will inevitably introduce serious privacy issues. Many attacks that compromise privacy have been proposed over public datasets, which were initially released for research. The majority of these attacks, referred to as *Record-linkage Attack*, combine network analysis, data mining and other technologies [2,5] to infer the identities of specific record based on some background knowledge. Specifically, according to *Record-linkage Attack*, for the multivariate structured data,

⋆ Zheng Gong and Kechun Zhao are co-first authors and contribute equally to this work. This work is granted by National Natural Science Foundation of China (No. 61672408, 61972309) and National Engineering Laboratory (China) for Public Safety Risk Perception and Control by Big Data (PSRPC)

an adversary could link personally identifiable attributes (*e.g.,* social security number), or quasi-identifier (*QI*) attributes (*e.g.,* age, gender), which may not expose the identity of the record independently but can leak the information if combined with other attributes, with his background knowledge to identify a particular or group of individuals in the dataset.

Existing *Record-linkage Attack* resist approach [1] requires user to predefine the sensitive/insensitive attributes. However, this prerequisite can be hardly satisfied, as the data owners could not gain insight into the sensitivity of each attribute. Meanwhile, manually labelling the *QI* or sensitive attributes is inaccurate and subjective. Obviously, it is of great importance if we can *identify and quantify all the "dangerous" attributes according to their risks of unveiling personal information if compromised.* Unfortunately, none of existing technique has explicitly defined such a qualification rule. Existing system[3] can arrange different sensitivity levels to corresponding columns by predefining the formats of some sensitive attributes through regular expressions. However, it is impossible to predefine templates for all attributes we may encounter. Besides, the formats of the entries in the same column may be diverse, even if they exhibit the same semantic meaning. Other techniques adopt some statistical tools to reflect the sensitivity of each column, such as cardinality [3], the proportion of unique values [8] and information entropy [7]. These methods do not take into account the correlation between columns, they consider each column independently. However, *Record-linkage Attack* can be successfully carried out by identifying tuples through *column combinations.*

Therefore, in this work, to address the problem, we present a novel technique for evaluating the sensitivity of each attribute in relational tables. The main contributions of this paper are as follows.

- To the best of our knowledge, we are the first to formally quantify the sensitivity of each column according to their re-identification risks.
- We demonstrate that our technique provides an objective measure for evaluating the sensitivity of each attribute empirically in a series of tables, and can act as a fundamental scheme for further privacy protection and desensitization. This technique is shown to be more realistic and complete than the existing empirical or statistics-driven methods.

The rest of the paper is organized as follows. In Section 2, we formally present the adversary model and key definition used in this paper. Our sensitivity evaluation method is proposed in Section 3. Empirical studies conducted over a group of datasets are presented in Section 4. Section 5 concludes the paper.

2 Definitions and Adversary Model

Before introducing the details of our framework, we shall first present basic definition and formally define the adversary model.

We adopt the definition of minimal unique column combination in [4], denoted as *UCC*. In layman's terms, the set of *UCC* is the set of independent

[3] `https://www.dbsec.cn/pro/dms-s.html`

primary keys and composite primary keys in a relational database R. Assume that the adversary has background knowledge κ, with which *Record-linkage Attack* can be performed to infer the real identification of some persons i in R. Obviously, as long as attribute(s) in κ can constitute a *UCC* of R, the adversary would successfully identify i, which is recognized as a successful attack. On the contrary, if κ doesn't contain any UCC, the adversary can never know if any information he got certainly corresponds to a unique tuple i in R.

Table 1: An Example dataset to be published

MINum	Sex	Age	Zip Code	Birthday	Disease
EN569244	Female	19	721001	1230	Fever
EF863453	Male	23	121000	0422	Pneumonia
EX756421	Female	56	831100	0719	Fever
EA556754	Female	14	201100	0926	Appendicitis
EP974423	Male	23	012000	1111	Leukemia
EN540305	Female	67	831100	1230	Fever
EY775612	Male	19	721001	0717	Leukemia

Table 2: The *UCC*s of Table 1

UCC
MINum
Age, *Disease*
Age, *Birthday*
Birthday, *Zip Code*
Age, *Sex*, *Zip Code*

Table 3: Background knowledge 1

Name	Age	Birthday
Lin	19	1230
Klay	23	0422
Haibara	56	0719
Jessie	14	0926
Charlotte	23	1111
Selena	67	1230
Quinn	19	0717

Table 4: Background knowledge 2

ID Number	MINum
4852	EN569244
8617	EF863453
1713	EX756421
2125	EA556754
3713	EP974423
9905	EN540305
5430	EY775612

For instance, Table 1 (R_e) is an ego dataset containing personal sensitive information, where $MINum$ refers to *Medical Insurance Number*. *UCC*s of R_e are listed in Table 2. Table 3 (R_1) and 4 (R_2) are the background knowledge κ of an adversary. When an adversary only has Table 3 as his background knowledge (*i.e.,* κ contains only R_1), it is impossible to determine the disease of any person through either *Age* or *Birthday*, but by a combination of both. As the *Age* and *Birthday* can form a *UCC*, the adversary can successfully identify every record in R_e. When κ contains only R_2, since *MINum* is a *UCC* of R_e, the adversary can get the ID number of every patient in Table 1, which may lead to the more disclosure of the private information.

3 The Risk of Re-identification

Given a database instance R with n columns, denoted as $A_1, \ldots, A_n$, for each column combination U over them (it can be viewed as an extensive projection without eliminating duplicate rows), if some rows of U are contained in κ, we denote them by $U \ltimes \kappa$. Suppose the probabilities for $\{A_i\} \ltimes \kappa$ are independent with each other and referred to as $p(A_i)$, respectively, then the success rate for *Record-linkage Attack* with respect to a particular column can be carried out as follows.

Firstly, if a column A_i itself constructs a UCC (*i.e.,* a key), the probability of successful attack through A_i should be $p(A_i)$. The reason is that as long as some rows of these columns are contained in κ, the adversary can definitely execute the attack. Secondly, if A_i is an element of some UCCs but not a UCC itself, the probability of successful attack through it would also depend on other columns that appear in those UCCs. Generally, we denote these UCCs as $U(A_i) = \{UCC | A_i \in UCC\}$. For each $UCC_j \in U(A_i)$, the adversary needs to cover other columns to successfully implement the attack once A_i is revealed. Suppose the columns that appear along with A_i in UCC_j is $B_1, ..., B_k$, let $P(UCC_j \ltimes \kappa)$ be the probability for UCC_j to be revealed (*i.e.,* all the columns of UCC_j for some rows are covered by κ) then its posterior probability, given that A_i is exposed, can be computed as $P(UCC_j \ltimes \kappa | \{A_i\} \ltimes \kappa) = \prod_{r=1}^{k} p(B_r)$. Notably, we also have to take into account the success rate for attacking through other UCCs in $U(A_i)$. Given $U(A_i)$, the adversary will successfully complete the attack if $\exists UCC_j \in U(A_i)$ such that $UCC_j \ltimes \kappa$. Therefore, once A_i is exposed, the successful attack probability through any UCC that contains A_i would be

$$P(success | \{A_i\} \ltimes \kappa) = 1 - \prod_{UCC_j \in U(A_i)} (1 - P(UCC_j \ltimes \kappa | \{A_i\} \ltimes \kappa)) \qquad (1)$$

Equation 1 in fact evaluates the posterior probability given that A_i has been exposed. Then the eventual probability for successful attack via A_i should be

$$S(A_i) = p(A_i) P(success | \{A_i\} \ltimes \kappa). \qquad (2)$$

For instance, if we uniformly set the reveal probability of each column in Table 1 as 0.5, respectively, then the sensitivity for the columns can be computed accordingly as follows. $S(MINum) = p(MINum) \times (1 - 0) = 0.5$, as *MINum* itself constructs a *UCC*. $S(Sex) = p(Sex)(1-(1-p(Age)p(ZipCode))) = 0.125$. Similarly, the sensitivity for the rest columns are $0.406, 0.313, 0.438, 0.375$, respectively. As discussed above, $S(A_i)$ reflects the probability for successful attack through attribute A_i, according to which we can quantitatively evaluate the attributes based on their risks for *Record-linkage Attack*.

Notably, according to Equation 2, $S(A_i)$ depends on $p(A_i)$, which refers to the general probability for A_i being revealed to an arbitrary adversary. Obtaining those probabilities seems to be a challenging task. For ease of discussion, we set $p(A_i)$ uniformly as 0.5 in the followings such that we can focus on discussing the intrinsic distribution-driven characteristics of each column, excluding all external scenario-dependent factors.

4 Experiments

To evaluate the performance of our framework, we conduct empirical tests over 2 real-world datasets as follows. The first dataset, denoted as RPI, is a real personal information tabular table containing 14 columns and 6478 tuples. These columns semantically refer to user ID number, birthday (*abbr.,* Birth.), address (*abbr.,* Addr.), mobile number (*abbr.,* Hp.), gender, etc. Another dataset is Pokec[4], the

[4] `http://snap.stanford.edu/data/soc-Pokec.html`

Table 5: the UCCs of RPI

UCC	Column combinations	UCC	Column combinations	UCC	Column combinations
UCC_1	*Id*	UCC_6	*Addr.,District3,Hp.,Name*	UCC_{11}	*Birth.,District4,Hp.*
UCC_2	*CtfId*	UCC_7	*Addr.,District4,Hp.,Name*	UCC_{12}	*Birth.,Hp.,Name*
UCC_3	*Birth.,Hp.,Zip*	UCC_8	*Addr.,Fax,Hp.,Name*	UCC_{13}	*Birth.,CtfTp,Gender,Hp.*
UCC_4	*Gender,Hp.,Name*	UCC_9	*Addr.,Hp.,Name,Zip*	UCC_{14}	*Birth.,Gender,Hp.,Tel*
UCC_5	*Addr.,Hp.,Name,Tel*	UCC_{10}	*Birth.,District3,Hp.*	UCC_{15}	*Addr.,Birth.,Hp.*

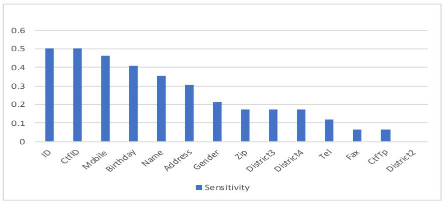


Fig. 1: Sensitivity of RPI ($p = 0.5$)

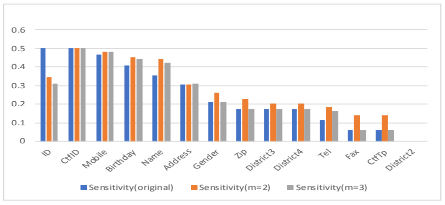


Fig. 2: Sensitivity of RPI after desensitization

most popular on-line social media in Slovakia. It contains 59 columns including gender, age, hobbies, interest, education, etc. As there are too many null entries, we extract the tuples with relatively less null values. For UCC discovery, we use Metanome Tool [6], an extensible implementation of [4].

4.1 Experimental results

Sensitivity justification In the first group of experiments, given RPI, whose UCCs are listed in Table 5, we *uniformly* set the reveal probabilities for columns as 0.5. Fig. 1 shows sensitivity results for each column according to Equation 2. ID and $CtfID$ exhibit the highest sensitivity. In fact, either ID or $CtfID$ can construct a UCC itself as shown in Table 5. For *Birthday*, an attack is successful only when the adversary gets other columns in UCCs that contain *Birthday*, such as $Mobile$ and Zip of UCC_3 in Table 5. Since the attack on *Birthday* requires more information, it would be more difficult to succeed when compared with the attacks on ID and $CtfID$. Naturally, its sensitivity should be lower than them, which is justified in Fig. 1.

To further illustrate the effectiveness, we use data masking method to desensitize columns with high sensitivity, and re-compute the sensitivity afterwards. For ID, we separately mask the the last m digits of each entry with '$\star$'. Fig. 2 shows the sensitivity results before and after the masking. Obviously, a higher level desensitization for ID results in a lower sensitivity, which also justifies the rationality of our evaluation scheme. Notably, the sensitivity of ID (originally 0.5) has dropped significantly to 0.344 and 0.31, respectively; the sensitivity of some other columns increase accordingly. Due to the masking, column ID is no longer a key in the table, and forms new UCCs together with other columns. Consequently, for the columns of newly-formed UCCs with respect to ID, their sensitivity will increase; for the rest columns, their sensitivity will remain unchanged.

Fig. 3 shows the results of Pokec. As there are too many columns in Pokec, we limit the number of columns in κ to be no more than 5 in Pokec. The reason

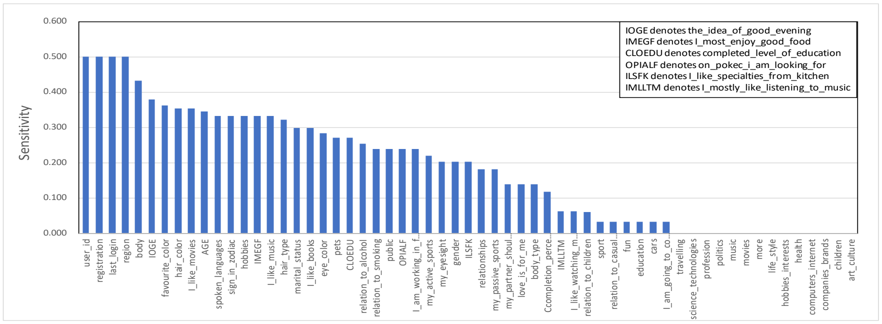


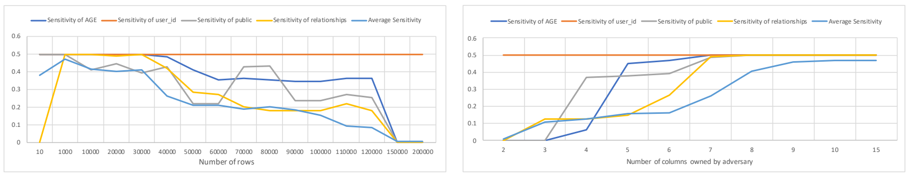
Fig. 3: Sensitivity of Pokec ($p = 0.5$)

for such setting is as follows. There is little chance for an adversary to obtain an extremely strong κ that contains so many columns at the same time, the probability of which obviously decreases exponentially in term of the number of columns. Similar to the phenomenon in the other datasets, the primary key ($user\ id$) has the greatest sensitivity, and the sensitivity of other columns is affected by the coverage of all UCCs.

Impact of other factors Furthermore, we also adjust the values of some parameters and explore how our model perceives these adjustments. First, we adjust the number of tuples in Pokec, gradually increasing from 10 to 150000, and explore the changes in the sensitivity of each column and the average sensitivity of all columns. According to Fig. 4a, when the number of rows is small, the average sensitivity of all columns (denoted as AS) fluctuates around 0.4. However, when the number of tuples is beyond 35000, the sensitivity of non-key columns decreases significantly. As the number of rows gradually increases, more duplicate values would be produced on a single column or multiple column combinations, which makes it more difficult for an adversary to uniquely re-identify individuals. Meanwhile, the differences on the numbers of tuples lead to different sensitivity for the same columns. It strongly justifies that our model is completely instance-aware.

In order to explore the impact of adversary's κ on sensitivity, we adjust the volume of κ, the maximal number of columns κ contains. Fig. 4b shows the corresponding results. Apparently, with the enhancement of κ, AS and the sensitivity of other non-primary key columns gradually increase. This is because the more columns an adversary obtains, the easier it is to successfully re-identify individuals. Note that the sensitivity of the primary key of Pokec($user\_id$) always remains stationary during the adjustment above.

These experiments strongly confirm that our model is completely consistent with our subjective perception of sensitivity. That is, as the number of rows of dataset increases and the κ of the adversary enhances, the sensitivity of each column generally decreases and increases, respectively. However, the key of database is not affected by these factors, and its sensitivity remains always the same and the highest.



(a) Effect of #*tuples* (b) Effect of $|\kappa|$

Fig. 4: Sensitivity changes w.r.t. other factors (in Pokec).

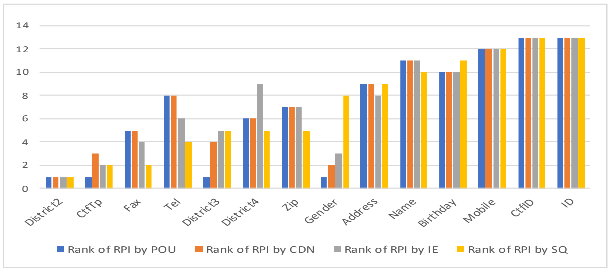


Fig. 5: Comparison with baselines (in RPI).

4.2 Comparison with other methods

To the best of our knowledge, we are the first to formally propose the sensitivity of re-identification of a column and quantify it. Traditional techniques adopt some statistical tools to reflect the risks for re-identification, such as cardinality [3] (CDN), the proportion of unique values [8] (POU) and information entropy [7] (IE). We compare with them in following experiments.

Considering that the absolute sensitivity values produced by different methods are incomparable, we only compare the rank of each column accordingly. The more sensitive the column is, the higher it ranks. The comparison results for RPI are shown in Fig. 5, where we denote our method as SQ for short. Apparently, CDN,POU and IE produce similar results, which validates that they only consider the distribution of the values in each single column. The ranking results of SQ are consistent with them on some columns (*e.g.,* $CtfID$ and ID), but are quite different on some others (*e.g.,* Tel and *Gender*). Tel ranks higher than *Gender* in CDN, POU and IE, but it is opposite with respect to SQ. The key reason for this phenomenon is that other methods do not take into account the correlation between columns, they consider each column independently. However, *Record-linkage Attack* can be successfully carried out by identifying tuples through *column combinations*, in which case *Gender* plays an important role. It is difficult to identify tuples by *Gender* itself, but its combination with other columns (*e.g.,* UCC_4, UCC_{13} and UCC_{14} in Table 5) can be easily identified. Consequently, it is irrational for CDN, POU and IE to give *Gender* a relatively low sensitivity level. This justifies our scheme takes into account not only the instance for each column but also the correlation between columns. Similar comparison results for Pokec are shown in Fig. 6, where the differences between SQ and other methods are more observably.

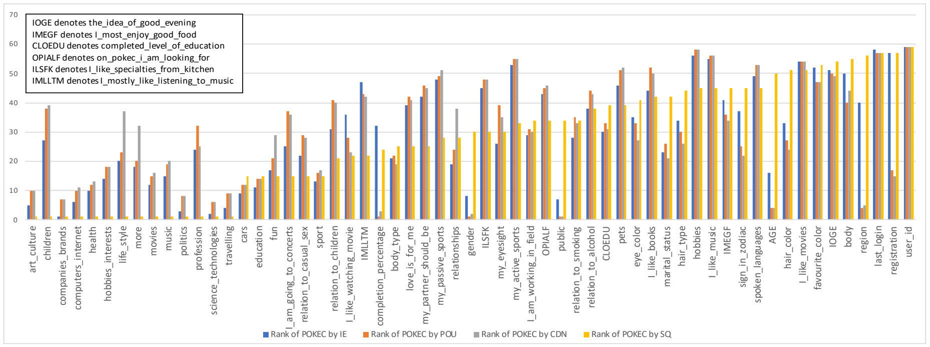


Fig. 6: Comparison with baselines (in Pokec).

5 Conclusion

In this paper, we present an efficient and pervasive technique for quantifying the sensitivity in structural database. It automatically evaluates the risks for re-identification for different columns according to *Record-linkage Attack*. Under the support of our scheme, the output sensitivity for the same attribute in different instances of a relational schema varies. We have objectively quantified it through a combination of probability model and adversary model. Moreover, the sensitivity quantifying results are basically in line with the common cognition of the public. We also thoroughly show that the proposed scheme is effective in dataset sensitivity governance comparing with baselines.

References

1. Abdelhameed, S.A., Moussa, S.M., Khalifa, M.E.: Privacy-preserving tabular data publishing: A comprehensive evaluation from web to cloud. Comput. Secur. **72**, 74–95 (2018)
2. Backstrom, L., Dwork, C., Kleinberg, J.M.: Wherefore art thou r3579x?: anonymized social networks, hidden patterns,and structural steganography. In: Proceedings of the 16th International Conference on World Wide Web (WWW). pp. 181–190. ACM (2007)
3. Chia, P.H., Desfontaines, D., Perera, I.M., et al.: Khyperloglog: Estimating reidentifiability and joinability of large data at scale. In: Proceedings of the 40th Symposium on Security and Privacy (SP). pp. 350–364. IEEE (2019)
4. Heise, A., Quiané-Ruiz, J., Abedjan, Z.: Scalable discovery of unique column combinations. PVLDB **7**(4), 301–312 (2013)
5. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: Proceedings of the 30th Symposium on Security and Privacy (SP). pp. 173–187. IEEE (2009)
6. Papenbrock, T., Bergmann, T., Finke, M., et al.: Data profiling with metanome. PVLDB **8**(12), 1860–1863 (2015)
7. Shlomo, N.: Methods to assess and quantify disclosure risk and information loss under statistical disclosure control. Tech. rep., Government Statistical Service (2018)
8. Skinner, C.J., Elliot, M.: A measure of disclosure risk for microdata. Journal of the Royal Statistical Society: series B (statistical methodology) **64**(4), 855–867 (2002)