LETTER

Manage the Tradeoff in Data Sanitization

Peng CHENG^{†a)}, Student Member, Chun-Wei LIN[†], Jeng-Shyang PAN[†], and Ivan LEE^{††}, Nonmembers

SUMMARY Sharing data might bring the risk of disclosing the sensitive knowledge in it. Usually, the data owner may choose to sanitize data by modifying some items in it to hide sensitive knowledge prior to sharing. This paper focuses on protecting sensitive knowledge in the form of frequent itemsets by data sanitization. The sanitization process may result in side effects, i.e., the data distortion and the damage to the non-sensitive frequent itemsets. How to minimize these side effects is a challenging problem faced by the research community. Actually, there is a trade-off when trying to minimize both side effects simultaneously. In view of this, we propose a data sanitization method based on evolutionary multi-objective optimization (EMO). This method can hide specified sensitive itemsets completely while minimizing the accompanying side effects. Experiments on real datasets show that the proposed approach is very effective in performing the hiding task with fewer damage to the original data and non-sensitive knowledge.

key words: data sanitization, privacy, frequent itemset mining, evolutionary multi-objective optimization, side effects

1. Introduction

It is increasingly common for data to be shared among different organizations in business collaboration. Partners may benefit from shared data by utilizing data mining tools to find useful relationships. However, data sharing might bring the risk of disclosing sensitive knowledge. Some relationships behind shared data could contain crucial strategies, business secrets and et al. Exposure of them will do harm to the data owner. To address this issue, the original database can be transformed into a released counterpart in such a way that sensitive knowledge cannot be discovered from a data mining view. The transforming process to hide sensitive knowledge is termed as data sanitization.

In this paper, we focus on privacy preserving in frequent itemset mining, which often serves as an important first step in many other data mining tasks. Sensitive itemsets can be concealed by reducing their supports below a specified threshold. This is achieved by modifying some items in the source database. Data modification may cause side effects, including data distortion and missing non-sensitive itemsets. In a data sharing environment, the receiver of the shared data would hope that the data accuracy and the non-

sensitive knowledge can be maintained to the largest extent possible. So an important task in data sanitization is to minimize the accompanying side effects.

Existing works on data sanitization fall into two groups. Some are intended for hiding large itemsets, while others aim at hiding association rules. We give a brief review on the former. Atallah et al. [1] studied the problem first and proved that many underlying problems for frequent itemset hiding are NP-hard. Verykios et al. [2] proposed a complete suite of heuristic algorithms for both itemset hiding and rule hiding. In their methods, the transaction length is utilized to identify candidate transactions for modification. More methods were created based on the relation between transactions and sensitive or non-sensitive frequent itemsets. For instance, Amiri [3] proposed three heuristic algorithms which select transactions and items for sanitization in terms of sensitive itemsets and non-sensitive itemsets related. Hong et al. [4] devised an itemset hiding method by using the concept of TF-IDF (Term Frequency-Inverse Document Frequency) in text mining. Their method assigns each supporting transaction a weight according to its relation with sensitive itemsets. The transactions with higher weights are sanitized firstly.

A variety of other solutions have been proposed [5], [6]. Generally, the difficulty of data sanitization is not in concealing sensitive knowledge, but in reducing the accompanying side effects. Existing solutions have implicitly or explicitly tried various ways to minimize side effects. As pointed by Gkoulalas-Divanis et al. [7], more advanced optimization techniques can be adopted to search the space of possible solutions. In this study, we investigated the itemset hiding problem from a view of multi-objective optimization. We formulated the side effects, including missing non-sensitive itemsets and data distortion, into optimization goals, and adopted evolutionary multi-objective optimization (EMO) to minimize them. Actually, a trade-off exists when minimizing them simultaneously. Improvement on one dimension often leads to degradation on the other. This fact has been neglected in the past research work. The great advantage of using EMO is that it can deal with the tradeoff and find optimal or nearly optimal solutions in the context of multiple conflicting optimization goals. Through a suite of experiments, we demonstrate the EMO-based solution can achieve satisfactory results with less damage to knowledge and data. In addition, the fact that a tradeoff exists within side effects has been corroborated.

Manuscript received December 20, 2014.

Manuscript revised May 25, 2015.

Manuscript publicized July 14, 2015.

[†]The authors are with Shenzhen Graduate School, Harbin Institute of Technology, P.R. China.

^{††}The author is with School of Computer and Information Science, University of South Australia, Australia.

a) E-mail: chengp.mail@gmail.com

DOI: 10.1587/transinf.2014EDL8250

2. Definitions and Problem Description

Let $I = \{I_1, I_2, \dots, I_m\}$ be a set of items available. An itemset X is a subset of I . A transactional database D is a relation consisting of a set of transactions. $D = \{T_1, T_2, \dots, T_n\}$, where each transaction T_i ($i \in [1, \dots, n]$) is characterized by an ordered pair, denoted as $\langle ID, X \rangle$. ID is a unique transaction identifier number which indicates its location in the database, and X is an itemset. A transaction T supports or contains an itemset X if $X \subseteq T$.

The absolute support of an itemset X is the number of transactions in D that contain X , which is denoted as $supp_count(X)$. Likewise, the relative support of X is the fraction (or percentage) of the transactions in D that contain X , denoted as $supp(X)$. $supp(X) = supp_count(X)/|D|$.

An itemset X is called frequent or large if $supp(X)$ is at least equal to a minimum relative support threshold (denoted as MST) specified by the user. The goal of frequent itemset mining is to find all frequent itemsets in the database D .

Assume that I denotes frequent itemsets mined from D , and I_S is the sensitive itemsets to be concealed. $I_S \subset I$. The itemset hiding process is to transform the source database D into a sanitized database D' , so that the sensitive itemsets I_S can not be mined out in D' with the same or higher MST s. Let I' denote the frequent itemsets mined from D' .

An example is introduced to clarify the concepts. Figure 1 shows a transactional database and frequent itemsets extracted from it. There are six transactions in the database. $MST = 50\%$. Suppose that the data owner decides to hide the itemset $\{d, c\}$, i.e., to reduce its support below MST . This can be done in a variety of ways. For instance, we may choose the 3rd transaction to sanitize by removing the item c or item d . Optionally, the 4th or 5th transaction also can be selected for sanitization in a similar way. However, choosing different ones to modify will bring different side effects. In order to minimize the side effects generated in the sanitization process, we transformed the hiding problem into a multi-objective optimization problem.

3. Hiding Methodology

A sensitive itemset can be hidden by reducing its support below MST . In other words, after sanitizing, the sensitive itemset becomes uninteresting from the frequent pattern

ID	Trans.		Freq. itemset	Supp	Freq. itemset	Supp
1	a c e f g	$\Rightarrow$	d f	0.5	e c	0.83
2	a c e f		d c	0.5	f c	0.83
3	b c d f g		g e	0.5	d f c	0.5
4	c d e f		g f	0.5	g e c	0.5
5	c d e f g		g c	0.67	g f c	0.5
6	b c e g		e f	0.67	e f c	0.67

Fig. 1 A transactional database and frequent itemsets extracted from it ($MST = 50\%$).

mining view. The difficulty lies in how to minimize the side effects in the hiding process.

3.1 Hide Sensitive Itemsets by Removing Items

The support of an itemset can be reduced by removing items in the transactions that contain it. The item with the highest support is selected to be removed. To reduce the support of a sensitive itemset X below MST , the minimum number of transactions that have to be sanitized is:

$$N_{mod} = |D| * supp(X) - |D| * MST + 1. \quad (1)$$

Where, $|D| * supp(X)$ is the absolute support of the itemset X , i.e., $|D| * supp(X) = supp_count(X)$. $|D| * MST$ is the minimum absolute support threshold. Although the minimum number of sanitized transaction to hide one sensitive itemset can be determined in this way, the overall number of sanitized transaction to hide all sensitive itemsets is indeterminate. It is common that many transactions contain several sensitive itemsets simultaneously.

3.2 Use EMO to Minimize Side Effects

3.2.1 Problem Formulation

From the view of multi-objective optimization, the hiding process is to choose a subset of transactions for sanitization, so that all sensitive itemsets can be concealed, and the accompanying side effects can be minimized.

We formulated the problem as follows. In this formulation, the side effect of missing non-sensitive itemsets and the side effect of data distortion act as two optimization goals. Both are normalized into $[0, 1]$. The constraint requires all sensitive itemsets to be concealed completely.

$$\text{Minimize: } F = [f_1, f_2] \quad (2)$$

$$f_1 = |I_{lost}| / (|I| - |I_S|)$$

$$f_2 = |\Sigma| / |D|$$

$$\text{Constraint: } I_S \cap I' = \emptyset$$

Where, I_{lost} denotes the lost non-sensitive itemsets after sanitization, and Σ denotes the sanitized transactions. Theoretically, the size of search space is $\{1, 2, \dots, |D|\}^{\Sigma}$. A sanitization approach needs to find the optimal solution(s) in the search space to minimize the different side effects simultaneously.

3.2.2 Evolutionary Multi-Objective Optimization (EMO)

EMO refers to use evolutionary algorithms to solve the multi-objective optimization problem. The use of population in EMO to conduct the search presents the advantage of generating several solutions in a single run. Generally, evolutionary algorithms are robust and less susceptible to the specific features of problems.

A platform and programming language independent EMO framework – PISA, has been developed [8]. In the

PISA framework, the components of an EMO algorithm are divided into two parts: the variation part and the selector part. The variation part is problem-related. While the selector part deals with the evolutionary selection. The two parts communicate via the file system. Our implementation was based on PISA. The algorithm NSGA-II [9] and HypE [10] were tried respectively in the selector part to drive the evolution forward. For the variation part, we devised problem-specific encoding scheme and efficient variation operators to search solutions.

3.2.3 Encoding Scheme and Variation Operators

The chromosome encoding used in the proposed approach is binary-based. Every bit in the chromosome corresponds to a transaction. Bit 1 denotes that the corresponding transaction is selected for modification and bit 0 otherwise. Only the transactions which support sensitive itemsets need to be considered. These supporting transactions can be retrieved in advance.

The supporting transactions are mapped to binary bits in the following way. Assume that there are n sensitive itemsets. The chromosome is divided into n segments accordingly. Each segment in the chromosome corresponds to a distinct sensitive itemset. The length of the i^{th} segment is the number of transactions which support the i^{th} sensitive itemset. The j^{th} bit in the i^{th} segment corresponds to the j^{th} supporting transaction. Here, $1 \leq i \leq |I_S|$ and $1 \leq j \leq N_i$. N_i is the number of transactions which support the i^{th} sensitive itemset. A transaction could correlate two or more bits in different chromosome segments since it may contain several sensitive itemsets simultaneously.

Figure 2 shows an example to illustrate the encoding scheme. The same database is used as in Fig. 1. $MST = 0.5$. Suppose that the sensitive itemsets are (d c) and (g c). The chromosome is divided into two segments, named as S_1 and S_2 , corresponding to two sensitive itemsets. The IDs of supporting transactions for (d c) and (g c) are {3, 4, 5} and {1, 3, 5, 6} respectively. According to Eq. (1), The minimum number of transactions which have to be sanitized to hide (d c) and (g c) is ($supp_count(d\ c) - 3 + 1 = 1$) and ($supp_count(g\ c) - 3 + 1 = 2$) respectively. Accordingly, there are one 1-bit in S_1 and two 1-bits in S_2 .

As Fig. 2 indicates, the 3rd transaction is selected for modification to hide (d c); the 3rd and 6th transactions are

selected for modification to hide (g c). If the victim item is determined to be the one with the highest support in the candidate transaction, then the item c in the 3rd transaction and the item c in the 6th transaction need to be removed respectively.

Ordinary variation operators, such as uniform or single point crossovers, may bring less (or more) 1-bits in each segment of the offspring chromosome. They could generate infeasible offspring solutions (an infeasible solution result in that the constraint in Eq. (2) is violated). For each segment, if the number of 1-bits is fewer than the value specified by Eq. (1), the corresponding sensitive itemset will not be concealed. So it is necessary that the variation operators should generate feasible offspring solutions. We devised special crossover and mutation operators to satisfy the constraint.

4. Performance Evaluation

We tested the proposed approach on real datasets: bms-1, bms-2, mushroom and chess. They are publicly available through the FIMI repository (<http://fimi.cs.helsinki.fi/>). These datasets exhibit varying characteristics, as summarized in Table 1.

The threshold MST was appropriately set for each dataset to ensure that sufficient frequent itemsets can be generated. A higher MST level was used for a denser dataset (such as mushroom or chess), and a lower MST level was used for a sparser dataset (such as bms-1 or bms-2). The density of a database is measured as the average transaction length divided by the number of different items.

The population size for EMO was 40 and the maximum generation was 200. For each test instance, we ran the proposed approach for twenty times to get the average. EMO may produce multiple solutions in a single run. The user often needs to choose a preferred one by using high-level domain experience. Generally, the receiver of shared data would likely be interested in a sharing agreement only when the data accuracy is reasonable [5]. In other words, the degree of data distortion cannot exceed a maximum level. Thus, a preferred solution can be selected in the following way. Firstly, a maximum level for data distortion is specified for each test instance. The solutions which can satisfy the constraint are considered as effective solutions. Then, from the effective solutions, the one which incurred least missing non-sensitive itemsets is chosen as the preferred solution.

We compared the proposed approach with the “Hybrid” method in [3] and the “SIF-IDF” method in [4]. NSGA-II [9] was used as the selector of EMO. The side effects, i.e.,

Fig. 2 The encoding scheme.

Table 1 The characteristics of datasets.

Dataset	Count of Tran.	Count of Items	Avg. Tran. Len.	MST
Bms-1	59,602	497	2.5	0.001
Bms-2	77,512	3,340	5.0	0.002
Mushroom	8,124	119	23	0.05
Chess	3197	75	40.2	0.5

Fig. 3 Obtained solutions by NSGA-II and HypE.

Table 2 Comparison of sanitization methods.

Dataset	$ I_S $	Performance (%)	EMO	Hybrid	SIF-IDF
Bms-1	10	Missing	1.110	1.280	1.919
		Distortion	1.967	1.971	1.992
	20	Missing	10.438	10.496	20.541
		Distortion	7.129	7.203	7.189
Bms-2	10	Missing	2.320	3.602	5.932
		Distortion	3.585	3.648	3.658
	20	Missing	6.710	8.875	17.533
		Distortion	6.453	6.590	6.540
Mushroom	10	Missing	1.137	9.856	2.957
		Distortion	25.920	27.028	26.105
	20	Missing	5.730	23.377	12.681
		Distortion	66.745	66.917	76.542
Chess	10	Missing	3.042	13.820	5.374
		Distortion	25.749	29.277	31.373
	20	Missing	24.051	30.528	27.397
		Distortion	40.696	60.307	71.786

the proportion of missing non-sensitive itemsets (denoted as *Missing*) and the proportion of sanitized transactions (denoted as *Distortion*), were used as the metrics to evaluate the performance. For each dataset, 10 and 20 sensitive itemsets were randomly selected respectively. The comparative results are indicated in Table 2. All three methods can completely hide sensitive itemsets.

From Table 2, we may notice that the EMO-based solution is effective on reducing side effects. For instance, on average, EMO, Hybrid and SIF-IDF sanitize the datasets with 6.816%, 12.729%, 11.792% missing non-sensitive itemsets, and with 22.277%, 25.368% and 28.148% data distortion degrees. It demonstrates that EMO may find more suitable transactions for sanitization to minimize side effects. On each dataset, increasing the number of sensitive itemsets may bring more side effects. This is expected since it requires to sanitize more supporting transactions to hide a larger number of sensitive itemsets. The density of a dataset may have an impact on side effects. For instance, the data distortion degree tends to be higher on a denser dataset (like mushroom or chess). Usually, the support of a sensitive itemset in a denser dataset is, on average, higher than in a sparser one. To reduce its support below *MST*, more transactions have to be sanitized.

A tradeoff relation exists within side effects. Figure 3 shows obtained solutions in a single run by using NSGA-II and HypE respectively to hide ten itemsets on each dataset. In most cases, the optimal solution for minimal side effects is not only one. This reflects the multi-objective property

of the sanitization problem. As discussed previously, a user needs to analyze alternative solutions and choose the preferred one. In addition, it indicates that, HypE [10], a recently proposed popular algorithm for many-objective optimization has not showed a superior performance on this two-objective optimization problem. It is not strange since HypE was devised for solving high-dimensional optimization problem (i.e., the goal number is greater than three). Its efficiency is achieved at the cost of accuracy.

5. Effectiveness of the Hiding Strategy

The proposed methodology preserves the sensitive itemsets by suppressing their supports below *MST*. In this way, the sensitive itemsets become infrequent and they cannot be mined out with the same or higher *MST* values in the shared data. However, how can we ensure that malicious attackers will not infer the hidden itemsets from the shared data? For instance, they may use relatively lower *MST* values to mine the shared data and attempt to expose hidden sensitive itemsets.

Let us make the following assumptions on what potential attackers may know:

- The sanitized database.
- The sanitization algorithm we devised.
- *MST* used in the sanitization process.

The attackers might expose the sensitive itemsets through mining the shared data with smaller values than *MST*. Even so, the sensitive itemsets still can be protected. If the number of potential itemsets with similar supports to sensitive ones is great, the attackers are not easy to distinguish which are sensitive from them. This situation often occurs when *MST* is relatively low and quite a lot of frequent itemsets can be generated. So when using the proposed methodology to sanitize a data set, a user is suggested to adopt a low *MST* value.

A more secure way is to use the Safety Margin threshold (*SM*). The safety margin is a value between 0 and *MST*. By defining the safety margin, the hiding approach needs to suppress the supports of sensitive itemsets below *MST* - *SM* instead of *MST*. This may cause the support levels of sensitive itemsets to become very low after data sanitization. If the safety margin is kept secret and not known to the adversaries, it is difficult for them to infer sensitive itemsets from the sanitized dataset.

Let us make a change to the assumptions. Suppose that the attackers are not aware of the exact *MST* value applied in data sanitization, they might try several *MST*s and receive outputs associated with those different *MST*s. Apparently, it is a more difficult situation for attackers. They cannot determine the updated supports of the hidden itemsets in the sanitized database, and have to try different *MST*s to search from a much wider range of patterns. It is almost impossible to find the hidden itemsets under such a situation.

6. Conclusions

In this study, we proposed a new solution to address the itemset hiding problem based on evolutionary multi-objective optimization. The EMO-based solution presents a variety of benefits. Firstly, it may easily manage the trade-off existing within side effects, and the side effects can be reduced to as few as possible. Secondly, its performance is robust to the dataset sanitized. Thirdly, EMO may produce multiple tradeoff solutions in a single run. This provides the opportunity for a user to choose his or her preferred solution by utilizing high-level information.

References

- [1] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. Verykios, "Disclosure limitation of sensitive rules," Proc. Workshop on Knowledge and Data Engineering Exchange (KDEX), pp.45–52, IEEE, 1999.
- [2] V.S. Verykios, A.K. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, "Association rule hiding," IEEE Trans. Knowl. Data Eng., vol.16, no.4, pp.434–447, 2004.
- [3] A. Amiri, "Dare to share: Protecting sensitive knowledge with data sanitization," Decision Support Systems, vol.43, no.1, pp.181–191, 2007.
- [4] T.-P. Hong, C.-W. Lin, K.-T. Yang, and S.-L. Wang, "Using TF-IDF to hide sensitive itemsets," Applied Intelligence, vol.38, no.4, pp.502–510, 2013.
- [5] S. Menon, S. Sarkar, and S. Mukherjee, "Maximizing accuracy of shared databases when concealing sensitive patterns," Information Systems Research, vol.16, no.3, pp.256–270, 2005.
- [6] A. Gkoulalas-Divanis and V.S. Verykios, "An integer programming approach for frequent itemset hiding," Proc. 15th ACM International Conference on Information and Knowledge Management, pp.748–757, ACM, 2006.
- [7] A. Gkoulalas-Divanis and V.S. Verykios, Association rule hiding for data mining, Advances in Database Systems, vol.41, Springer, 2010.
- [8] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler, "PISA — A platform and programming language independent interface for search algorithms," Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol.2632, pp.494–508, Springer, Berlin, Heidelberg, 2003.
- [9] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput., vol.6, no.2, pp.182–197, 2002.
- [10] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," Evolutionary Computation, vol.19, no.1, pp.45–76, 2011.