Association for Information Systems

AIS Electronic Library (AISeL)

ICEB 2004 Proceedings

International Conference on Electronic Business (ICEB)

Winter 12-5-2004

Privacy-Preserving Collaborative Association Rule Mining

Justin Zhan

Stan Matwin

Nathalie Japkowicz

LiWu Chang

Privacy-Preserving Collaborative Association Rule Mining

Justin Zhan[1], Stan Matwin[1], Nathalie Japkowicz[1] , LiWu Chang[2]

[1] School of Information Technology & Engineering, University of Ottawa, Canada
[2] Center For High Assurance Computer Systems, Naval Research Laboratory, USA
{zhizhan, stan, nat}@site.uottawa.ca, lchang@itd.nrl.navy.mil

ABSTRACT

In recent times, the development of privacy technologies has promoted the speed of research on privacy-preserving collaborative data mining. People borrowed the ideas of secure multi-party computation and developed secure multi-party protocols to deal with privacy-preserving collaborative data mining problems. Random perturbation was also identified to be an efficient estimation technique to solve the problems. Both secure multi-party protocol and random perturbation technique have their advantages and shortcomings. In this paper, we develop a new approach that combines existing techniques in such a way that the new approach gains the advantages from both of them.

Keywords: Privacy, data mining, association rule mining, secure multi-party computation.

1. INTRODUCTION

Business successes are no longer the result of an individual toiling in isolation; rather successes are dependent upon collaboration, team efforts, and partnership. In the modern business world, collaboration becomes especially important because of the mutual benefits it brings. Specially, the explosion in the availability of various kinds of data has triggered tremendous opportunities for collaboration. This paper studies a very specific collaboration that becomes more and more prevalent. The problem is the collaborative data mining. The objective of our research is to develop technologies to enable multiple parties to conduct data mining collaboratively without disclosing their private data. Since its introduction [7], privacy-preserving collaborative data mining problem has attracted many researchers. The approaches applied to solve the problem can be categorized into two aspects: one is based on secure multi-party computation [3, 10]; the other is based on randomization techniques [1, 4,5]. The first approach can achieve precise results but its computation cost is usually much greater than that of the second approach. The second approach is computationally efficient but its accuracy is not as good as that of the first approach. This motivates us to find a combination in which the two approaches are integrated. The combined approach gains advantage of good accuracy that the first approach has, without losing benefit of high efficiency that the second approach has. In this paper, a secure integrated protocol is presented to deal with collaborative data mining problems. Collaborative data mining includes a number of different tasks, such as collaborative association rule mining, classification, and clustering, etc. This paper studies collaborative association rule mining problem. The goal is to discover meaningful association rules among the attributes of a large quantity of data contributed by all the collaborative parties.

The paper is organized as follows: Section 2 discusses the related work. Section 3 formally defines the privacy-preserving multi-party association rule mining problem. We develop a secure protocol in Section 4. We describe our experimental results in Section 5. Further discussion is provided in Section 6. Section 7 concludes the paper and lays out the future work.

2. RELATED WORK

2.1 Secure Multi-party Computation

Briefly, a Secure Multi-party Computation (SMC) problem deals with computing any function on any input, in a distributed network where each participant holds one of the inputs, while ensuring that no more information is revealed to a participant in the computation than can be inferred from that participant's input and output [6]. It has been proved that for any polynomial function, there is a secure multi-party computation solution [6]. The approach used is as follows: the function F to be computed is first represented as a combinatorial circuit, and then the parties run a short protocol for every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, though appealing in its generality and simplicity, is highly impractical for large datasets.

2.2 Privacy-Preserving Data Mining

In early work on such a privacy-preserving multi-party data mining problem, Lindell and Pinkas [7] propose a solution to privacy preserving classification problem using oblivious transfer protocol, a powerful tool developed by secure multi-party computation (SMC) research. Other techniques based on SMC were then proposed in [3,10]. Randomization approaches was firstly proposed to solve privacy-preserving data mining problem by Agrawal and Srikant in [1]. People proposed more random perturbation based techniques to tackle the problem in [1,4,5]. The techniques based on SMC have characteristics that mining results will be the same as original results without considering data privacy. In contrast, mining results using techniques based on randomization will be the estimation of original results but performance is usually better than SMC-based techniques. In this paper, we propose an approach which requires integrated use of two existing approaches. Through integration of two approaches, we can gain advantages of both approaches.

3. INTRODUCING THE PROBLEM

We consider the scenario where multiple parties, each having a private data set denoted by D_1, D_2, …, and D_n respectively where n is the total number of parties, want to collaboratively conduct association rule mining on the union of their data sets. Because they are concerned about their data privacy, neither party is willing to disclose its raw data set to others. Specially, we consider the scenario where data sets of collaborative parties are binary and vertically partitioned [10]. In other words, the identities of the ith (for $i \in [1, N]$) record in D_1, D_2, …, and D_n are the same, and D_1, D_2, …, and D_n contain the same number of records (we use N to denote the total number of records for each data set).

4. SECURE MULTI-PARTY ASSOCIATION RULE MINING PROTOCOL

4.1 The Commodity Server Model

The commodity server model was proposed in [2]. The collaborative parties could send requests to the commodity server and receive data from the server, but the commodities must be independent of the parties' private data. The purpose of the commodities is to help the parties to conduct their desired computations. The commodity server is semi-trusted in the following senses: (1) It should not be possible to derive private information of data from the parties; it should not learn computation result either. (2) It should not collude with all the parties. (3) It follows the protocol correctly. Because of these characteristics, we say that it is a semi-trusted party. In real world, finding such a semi-trusted party is much easier than finding a trusted party. Based on this commodity server model, we study privacy-preserving multi-party association rule mining problem. In our solution, the commodities are randomly generated permutation functions [8]. Our goal is to permute all the parties' data in such a way that a designated party (data collector) can conduct association rule mining on permuted data and obtain the same mining results. The challenge is to prevent the designated party from knowing how the other parties' data are permuted.

4.2 Notations

For convenience, we define the following intuitive notations:

1. Let V represent a permutation function that is used to permute the order of the columns in a data set. We call it the column permutation function. We use V^{-1} to represent the inverse of V.
2. Let H represent a permutation function that is used to permute the order of the rows in a data set. We call it the row permutation function. We use H^{-1} to represent the inverse of H.
3. Let VH = (V, H) represent a permutation function that contains a row permutation H and a column permutation V. VH means that we firstly apply H on the data set, V is then applied. We use $(VH)^{-1}$ to represent the inverse of VH.
4. We use $\Rightarrow$ as the permutation sign. For instance, $R_1 \Rightarrow R_3$ means that Row 1 will be permuted to Row 3 and $C_1 \Rightarrow C_4$ means that Column 1 will be permuted to Column 4.

4.3 Privacy-Preserving Multi-Party Association Rule Mining Protocol

At the beginning of our protocol, a data collector is randomly selected from all the collaborative parties. For simplicity, let's assume Party n is selected as data collector. The commodity server then generates a set of permutation functions satisfying some properties and sends them to the collaborative parties. Party 1, 2, … , n-1 permute their data by themselves and send permuted data to Party n. Since Party n also contributes its data set to the mining computation, its data must also be permuted similarly, i.e., the row permutation effects have to be the same. To avoid information leak, the permutation of Party n's data has to be handled differently. In our protocol, the permutation of Party n's data is done by itself together with another party (Party i) who is randomly selected from Party 1, 2, …, n-1. Party i and Party n will together permute the Party n's data such that the permuted data have the same row permutation as other permuted data sets. To prevent Party n from conducting possible attacks, multi-variant randomized response technique [4] is applied on the data of Party n.

Multi-Party Mining Protocol

Step I: Permutation Function Generation

1. The Commodity Server (CS) generates a set of column permutation functions V_1, V_2, V_3, …,V_{n-1} and a row permutation function H. It then sends V_1H to Party 1, V_2H to Party 2, …, and $V_{n-1}H$ to Party n-1.
2. The CS generates the other two sets of permutation functions $V_n'H'$ and $V_n''H''$. It then sends $V_n'H'$ to Party n, $V_n''H''$ to Party i where Party i is randomly selected from Party 1, Party 2 … and Party n-1.

3. The permutation functions have the following property: V_1H , V_2H , …, $V_{n-1}H$ and $V_n"H"V_n'H'$ have the same row permutation effects.

Definition 1 We say two permutation functions PF_1 and PF_2 have the same row permutation effects if $PF_1(D_1)$ and $PF_2(D_2)$ have the same row permutations.

Step II: Data Permutation

Sub-step 1: Permute D_1, D_2 …, and D_{n-1}

1. Party 1 applies V_1H on its data set D_1 to get a permuted data set PD_1, then sends PD_1 to Party n.
2. Party 2 applies V_2H on D_2 to obtain PD_2, and then sends it to Party n.
3. Repeat the above process until Party n-1 applies $V_{n-1}H$ on its data set D_{n-1} and obtain PD_{n-1}, then sends it to Party n.

Sub-step 2: Permute D_n

1. Party n applies $V_n'H'$ on D_n to obtain PD'_n, then sends it to Party i.
2. Party i applies $V_n"H"$ on PD'_n to obtain $PD"_n$, then randomizes $PD"_n$ using the multi-variant randomized response technique described in Section 4.5, finally obtains PD_n and sends it to Party n (We will explain why Party i randomizes $PD"_n$).

Sub-step 3: Combine the Permuted Data Sets

1. Party n concatenates all the data sets PD_1, PD_2 …, PD_{n-1} and PD_n such that the ith row in the PD_1, PD_2, … and PD_n becomes the ith row in the joint data set PD.

Step III: Mining

1. Party n conducts the association rule mining on the joint data sets PD and gets the permuted association rules denoted by PAR_n. Since the data collector's data are randomized, the standard association rule mining algorithm cannot be employed. Section 4.7 shows how to conduct association rule mining on data sets which contain both non-randomized data (e.g., the data of Party 1, 2… n-1) and randomized data (e.g., the data of Party n).

Step IV: Final Association Rules

Sub-step 1: Find the Final Rules Containing the Attributes from Party 1, Party 2, …, and Party n-1

1. Party n sends the attributes, which the permuted association rules contain, to the corresponding parties (from Party 1 to Party n-1) who send the name of the original attributes to Party n. For example, if the permuted rules contains a rule $PD_3(A_1) \Rightarrow PD_5(A_4)$ with $PD_3(A_1)$ from Party 3 and $PD_5(A_4)$ from Party 5, Party n then sends $PD_3(A_1)$ to Party 3 and $PD_5(A_4)$ to Party 5. Without loss of generality, assume the original attributes of $PD_3(A_1)$ and $PD_5(A_4)$ are $D_3(A_8)$ and $D_5(A_3)$ respectively, they then send the attribute names of $D_3(A_8)$ and $D_5(A_3)$ to Party n. Note that the attribute indices are used in the protocol, e.g., $PD_3(A_1)$ denotes the first attribute in the permuted data set of Party 3. However, the data collector does not know whether $PD_3(A_1)$ stands for *bread*, *butter*, or others. After the parties remove the column permutation functions and obtain the original attribute indices, e.g., Party 3 gets $D_3(A_8)$, they will not send the original attribute indices (e.g., $D_3(A_8)$) to the data collector and only sends the real attribute name that the original attribute indices denote to Party n. For instance, if the $D_3(A_8)$ stands for *bread*, it sends *bread* to the data collector. The purpose of the above process is to prevent the data collector from inferring other parties' column permutation functions based on the resultant rules.

Sub-step 2: Find the Final Rules Containing Party n's Attributes

1. Party n sends its permuted attributes that the permuted association rules contain, to Party i who applies the inverse of $V"_n$ on the permuted attributes and obtains some attributes. We call them the middle permuted attributes since they are still in permuted form (e.g., permuted by V'_n).
2. Party i sends the middle permuted attributes to Party n who applies the inverse of V'_n on them and obtains the original Party n's attributes.

Sub-step 3: Find the Final Rules

After obtaining all the original attributes of the permuted attributes that the permuted association rules contain, Party n gets the final association rules and shares them with all the other parties.

4.4 Why Party i Randomizes the Party n's Data

Assume that after Party i applies $V"_n H"_n$ on the data $V_n'H_n'(D_n)$ obtained from Party n, Party i directly sends $V"_n H"_n V'_n H'_n(D_n)$, without randomization, back to Party n. Since the total number of records N is usually much larger than the total number of attributes M in a large-scale data set, the probability of the sum of each column being unique is high. Thus, Party n is likely to find out the column permutation function $V"_n$ by comparing $V'_n H'_n(D_n)$ with $V"_n H"_n V'_n H'_n(D_n)$ using the sum of each column. After finding $V"_n$, she can remove V''_n from $V''_n H''_n V'_n H'_n(D_n)$. If Party n can somehow find out H''_n, Party n then knows $V''_n H''_n$. In other words, she knows $V''_n H''_n V'_n H'_n(D_n)$. Since all the parties' data are permuted in a way with the same row permutation effects and Party n also obtains other parties' permuted data, Party n can find out other parties' row permutation function *H*. H''_n is likely to be disclosed for any row of D_n that has an unique pattern. To avoid this problem, we let Party i randomize $V''_n H''_n V'_n H'_n(D_n)$ before sending it to Party n. Even if the Party n's data has the above special characteristics, Party n cannot exactly identify the Party i's permutation function $V''_n H''_n$ because of randomization. The challenge is how Party n conducts association rule mining after obtaining data set *PD*. Obviously, Party n cannot apply the traditional association rule mining algorithm [9], since part of the data set are not randomized and part of them are randomized. We will show how Party n conducts the association rule mining on the hybrid data set *PD* in Section 4.7.

4.5 How Party i Randomizes The Party n's Data

We propose to use randomization technique [4] to disguise $V''_n H''_n V'_n H'_n(D_n)$. The basic idea is that, for each record, we keep the same values with a predefined probability q; we reverse the values with the probability of $1-q$. For example, assume there are 3 attributes for a record and the values for the record is $(A_1=1, A_2=1, A_3=0)$. We generate a random number between 0 and 1. If the number is less than q, the record is kept the same, that is $(A_1=1, A_2=1, A_3=0)$. If the number is greater than q, the record becomes $(A_1=0, A_2=0, A_3=1)$. For each record, we generate a random number to decide whether the record keeps the same values or being reversed. Finally, Party i obtains the data set PD_n and sends it to Party n. To conduct association rule mining on the randomized data set, we need to compute $P(A_1=1, A_2=1, A_3=0)$. To simplify our presentation, we use P(110) to represent $P(A_1=1, A_2=1, A_3=0)$, P(001) to represent $P(A_1=0, A_2=0, A_3=1)$. Because the contributions to $P*(110)$ and $P^*(001)$ partially come from P(110), and partially come from P(001), we can derive the following equations:

$$P*(110) = P(110)\cdot q + P(001)\cdot(1-q)$$

$$P^*(001) = P(001)\cdot q + P(110)\cdot(1-q)$$

By solving the above equations, we can get P(110). The general model is described in the following:

$$P*(E) = P(E)\cdot q + P(\bar{E})\cdot(1-q)$$

$$P^*(\bar{E}) = P(\bar{E})\cdot q + P(E)\cdot(1-q)$$

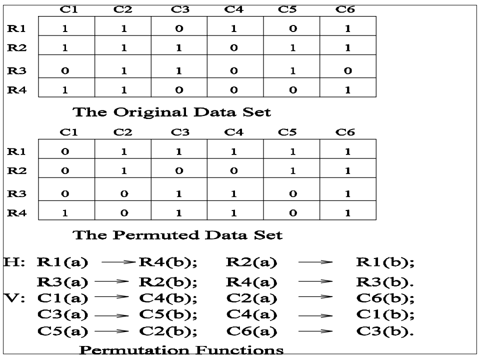
where *E* is a logical expression. In the above example, $E=(A_1=1, A_2=1, A_3=0)$ and $\bar{E}=(A_1=0, A_2=0, A_3=1)$. To evaluate the proposed randomization approach, we conduct extensive experiments in Section 5.

4.6 How Can We Obtain the Final Association Rules from The Permuted Association Rules

In the mining step, Party n obtains the permuted association rules PAR_n. To obtain the final association rules from PAR_n, the parties need to apply the inverse of their column permutation functions on the permuted association rules. In the following, we use a simple example to illustrate the computation. Without loss of generality, we assume the original data, permuted data and permutation functions are shown in Fig. 1. Based on the permuted data, suppose that we obtain the permuted association rules as shown in Fig. 2. We can then apply the inverse of the column permutation functions and get the final association rules (Fig. 2).

4.7 Multi-Party Association Rules Mining On Randomized Data

The following is the procedure for mining association rules on $PD_1 \cup PD_2 \cup \cdots \cup PD_n$. It differs from traditional association rule mining procedure [9] in the computation of the frequency count (denoted as *c.count*). When the data are not randomized, we can easily compute the *c.count*, but when part of the data are randomized using the multi-variant randomized response techniques, computing it becomes non-trivial.



	C1	C2	C3	C4	C5	C6
R1	1	1	0	1	0	1
R2	1	1	1	0	1	1
R3	0	1	1	0	1	0
R4	1	1	0	0	0	1

The Original Data Set

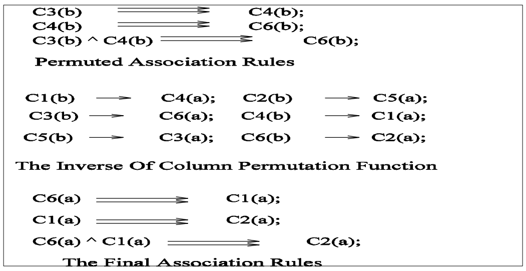
	C1	C2	C3	C4	C5	C6
R1	0	1	1	1	1	1
R2	0	1	0	0	1	1
R3	0	0	1	1	0	1
R4	1	0	1	1	0	1

The Permuted Data Set

H: R1(a) → R4(b); R2(a) → R1(b);
R3(a) → R2(b); R4(a) → R3(b).
V: C1(a) → C4(b); C2(a) → C6(b);
C3(a) → C5(b); C4(a) → C1(b);
C5(a) → C2(b); C6(a) → C3(b).

Permutation Functions

FIGURE 1



C3(b) ⇒ C4(b);
C4(b) ⇒ C6(b);
C3(b) ^ C4(b) ⇒ C6(b);

Permuted Association Rules

C1(b) → C4(a); C2(b) → C5(a);
C3(b) → C6(a); C4(b) → C1(a);
C5(b) → C3(a); C6(b) → C2(a);

The Inverse Of Column Permutation Function

C6(a) ⇒ C1(a);
C1(a) ⇒ C2(a);
C6(a) ^ C1(a) ⇒ C2(a);

The Final Association Rules

FIGURE 2

For the part which is randomized, we do not know whether a record in the data set is true information or false information. Therefore, we cannot directly compute *c.count* on the randomized data. In the following, we will show how to compute *c.count* when some attributes are randomized. For example, we want to compute *c.count* for A_1, A_2, and A_3. Let's assume A_1 and A_2 are not randomized and A_3 is randomized. We can use the following estimation model.

$$P^*(A_1A_2A_3) = P(A_1A_2A_3)\cdot q + P(A_1A_2\bar{A}_3)\cdot(1-q)$$

$$P^*(A_1A_2\bar{A}_3) = P(A_1A_2\bar{A}_3)\cdot q + P(A_1A_2A_3)\cdot(1-q)$$

In general, when part of the attributes (denoted by RE) are randomized and the other part (denoted by NE) is not randomized, then the estimation model becomes:

$$P^*(RE \wedge NE) = P(RE \wedge NE)\cdot q + P(\bar{RE} \wedge NE)\cdot(1-q)$$

$$P^*(\bar{RE} \wedge NE) = P(\bar{RE} \wedge NE)\cdot q + P(RE \wedge NE)\cdot(1-q)$$

where ^ is logical *and* operator. $P^*(RE \wedge NE)$ and $P^*(\bar{RE} \wedge NE)$ can be obtained from the randomized data set and q is known (With the knowledge of q, Party n only knows the probability of each value of PD_n comes from the values of $V''_n H''_n V'_n H'_n (D_n)$). We can solve the above equation and get $P(RE \wedge NE)$. We can then compute $c.count = P(RE \wedge NE) * N$.

5. EXPERIMENTAL RESULTS

To evaluate the effectiveness of our proposed scheme, we conduct association rule mining on the original data using the traditional association rule mining algorithm [18] and get a benchmark rule sets T_D. We then induce our rule set T_G from PD using our modified algorithm. We test the difference of these two sets. Since Party n's data are randomized by the multi-variant randomized response technique, we modified the association rule mining algorithm to handle the randomized data based on our proposed methods in Section 4.7. We run this modified algorithm on the randomized data, and find a set of association rules (e.g., our rule set T_G). We also apply the original association rule mining algorithm to the original data set and find the other set of association rules. We then compute the error which is defined as the number of rules that the benchmark rule set T_D contains but our rule set T_G does not contain. The error rate is defined as the error divides the total number of rules in the benchmark rule set T_D. In this section, we will show the experimental results on two real-life data sets.

5.1 Data Setup

We conducted experiments on two real life data sets. We obtain the data sets from the UCI Machine Learning Repository. The first data set is *Adult*. The second data set is *Breast-Cancer*.

5.2 Experimental Steps

Our experiments consist of the following steps:

Step I: Since we assume that the data set contains only binary data, we firstly transformed the original non-binary data to the binary (D). We split the value of each attribute from the median point of the range of the attribute.

Step II: We use D and the original association rule mining algorithm to build a rule set T_D.

Step III: We randomly select 10% number of attributes from D. For $q = 0$, 0.1, 0.2, 0.3, 0.4, 0.51 (*We use 0.51 instead of 0.5. Because when* $q = 0.5$, *estimation equation doesn't have a unique solution. We call this point an exception point in our approach.*}, 0.6, 0.7, 0.8, 0.9, 1, we conduct the following 4 steps:

1. Randomization: We create a randomized data set G. For each record in the training data set D, we generate a random number *r* from 0 to 1 using uniform distribution. If $r \le q$, we copy the record to G without any change; if $r \succ q$, we copy the *opposite* values of selected attributes in the current record to G and we copy the *original* values of non-selected attributes in the current record to G. For instance, suppose there are three attributes (e.g., A_1, A_2, and A_3) in D and the original values for them are $A_1 = 0$, $A_2 = 0$ and $A_3 = 0$. If the selected attributes are A_1 and A_2. When $r \succ q$, the values for these two attributes will be $A_1 = 1$ and $A_2 = 1$ in the current record of G, but the value for A_3 is still 1. We perform this randomization step for all the records in the training data set D and generate the new data set G.
2. Build a rule set: We use the data set G and our modified association rule mining algorithm to build our rule set T_G.
3. Compute the error: We compare T_D and T_G, and compute the error rate.
4. Repeating: We repeat the above steps for 50 times, and get E_1, E_2, …, E_{50}. We then compute the mean of these 50 error rate scores.

Step IV: We randomly select 20% and 50% number of attributes respectively from D. We repeat Step III (1-4) . Since there are at least two parties in the multi-party data mining, Party n has at most 50% number of attributes in the combined data set PD if we assume that each party's data have the same number of attributes. Therefore, we didn't conduct experiments when more than 50% attributes are randomized.

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0	0	0	0.023	0.256	0.024	0	0	0	0

(a) 10% Attributes Are Randomized

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0.12	0.12	0.2	0.352	0.336	0.24	0.24	0.12	0.08	0

(b) 20% Attributes Are Randomized

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0.16	0.16	0.32	0.326	0.376	0.247	0.24	0.168	0.12	0

(c) 50% Attributes Are Randomized

FIGURE 3

5.3 Result Analysis

Fig. 3 shows the mean of error for *Adult* data set with the $\text{minsup} = 0.6$. Fig. 4 depicts the mean of error for *Breast-Cancer* data sets with the $\text{minsup} = 0.45$. We can see when $q = 1$ and $q = 0$, the results are exactly the same as the results when the original association rule mining algorithm is applied. This is because when $q = 1$, the randomized data sets are exactly the same as the original data set D; when $q = 0$, the randomized data sets are exactly the opposite of the original data set D. In both cases, our algorithm produces the accurate results comparing to the original algorithm. However, the privacy is not preserved in either case because an adversary can know the real values of all the records provided that she knows the q value. When q moves from 1 and 0 towards 0.5, the mean of error has the trend of increasing, and the probability of disclosing the original data is decreasing. Therefore the privacy level of the data increases. When more and more data are randomized, the mean of error will increase.

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0	0	0	0.04	0.044	0.03	0	0	0	0

(a) 10% Attributes Are Randomized

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0.043	0.043	0.044	0.047	0.066	0.042	0.041	0.039	0.024	0

(b) 20% Attributes Are Randomized

θ	0	0.1	0.2	0.3	0.4	0.51	0.6	0.7	0.8	0.9	1
Error Rate	0	0.088	0.107	0.113	0.116	0.183	0.109	0.104	0.077	0.05	0

(c) 50% Attributes Are Randomized

FIGURE 4

6. DISCUSSION

In this paper, we combine the secure multi-party computation approach with the random perturbation approach to tackle the problem of privacy-preserving collaborative association rule mining. As shown in previous sections, all parties' data are permuted, but Party n's data are also randomized. As a result, the mined rules containing Party n's attributes are degraded with a certain level of inaccuracy. On the other hand, the rules that do not contain Party n attributes will be 100% accurate comparing to the results obtained by applying the traditional association rule mining algorithm [9] on the non-randomized and non-permuted data. Therefore, the results will be the tradeoff between the SMC-based approach and the randomization approach. Our secure multi-party mining protocol gains the advantage of good accuracy that the SMC-based approach has, without losing the benefit of high efficiency that the randomization approach has. Although we deal with the association rule mining problem in this paper, the proposed approach can be extended to solve other data mining problems, e.g., decision tree, Bayesian network classification, etc. To improve privacy level, we can also use multi-group randomized response [11] to randomize the Party n's data. We need also point out that, Party n can apply maximum likelihood estimation method to attack H'' when the distribution of rows of D_n is highly skewed, and the number of records of D_n is small. Therefore, the data distribution should be considered when selecting the data collector.

7. CONCLUSION

We have presented a secure protocol to let multiple parties to find association rules on the joint data sets without comprising their data privacy. In the protocol, the data collector is firstly selected. All the parties except for the data collector permute their data using the permutation functions obtained from the commodity server. To prevent the data collector from conducting possible attacks on other parties' permutation functions, the multi-variant randomized response technique is utilized to randomize data collector's data. Finally, we conduct a set of experiments to evaluate the effectiveness of our proposed scheme.

In this paper, we only consider the semi-trust case in which all the collaborative parties honestly follow the protocol. The more challenging scenario is the malicious case where the collaborative parties may cheat with each other. As future work, we will develop the protocol to deal with the malicious case.

REFERENCES

[1] R. Agrawal and R. Srikant, Privacy-preserving data mining, In proceedings of the ACM SIGMOD conference on management of data, Dallas, Texas, USA, 2000.

[2] D. Beaver, Commodity-based cryptography (extended abstract), In proceedings of the twenty-ninth annual ACM symposium on theory of computing, EL Paso, TX USA, May 4-6, 1997.

[3] W. Du and Z. Zhan, Building decision tree classifier on private data, In workshop on privacy, security and data mining at the 2002 IEEE international

conference on data mining, Maebashi City, Japan, December 9, 2002.

[4] W. Du and Z. Zhan, Using randomized response techniques for privacy-preserving data mining, In proceedings of the 9th ACM SIGKDD international conference on knowledge discovery and data mining, Washington, DC, USA, August 24-27, 2003.

[5] A. Evfimievski, J. Gehrke, and R. Srikant, Limiting privacy breaches in privacy-preserving data mining, In the proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems, San Diego, California, 2003.

[6] O. Goldreich, Secure multi-party computation (working draft), http://www.wisdom.weizmann.ac.il /home/oded/public_html/foc.html, 1998.

[7] Y. Lindell and B. Pinkas, Privacy-preserving data mining, Advantages in cryptology-CRYPTO’00, 1880 of lecture notes in computer science, Spinger-Verlag: 36-54, 2000.

[8] M. Luby, Pseudorandomness and cryptographic applications, Princeton University Press, Jan., 1996.

[9] R. Agrawal, T. Imielinshi and A. Wigderson, mining association rules between sets of items in large databases, In proceedings of ACM SIGMOD conference on management of data, pages:207-216, May, 1993.

[10] J. Vaidya and C. Clifton, Privacy-preserving association rule mining in vertically partitioned data, In proceedings of the 8th ACM SIGKDD international conference on knowledge discovery and data mining, July 23-26, 2002.

[11] Z. Zhan, L. Chang and S. Matwin, Privacy-preserving multi-party decision tree induction, In the 18th annual IFIP WG 11.3 working conference on data and application security, Sitges, Catalonia, Spain, 2004.