Privacy Preserving Classification in Two-Dimension Distributed Data

Luong The Dung*, Ho Tu Bao†, Nguyen The Binh* and Tuan-Hao Hoang‡

*Information Technology Center
VietNam Government Information Security Commission

†Japan Advanced Institute of Science and Technology
Nomishi, Ishikawa, Japan

‡Faculty of Information Technology,
Le Quy Don Technical University, Hanoi, Vietnam

***Abstract*—Within the context of privacy preserving data mining, several solutions for privacy-preserving classification rules learning such as association rules mining have been proposed. Each solution was provided for horizontally or vertically distributed scenario.**

The aim of this work is to study privacy-preserving classification rules learning in two-dimension distributed data, which is a generalisation of both horizontally and vertically distributed data. In this paper, we develop a cryptographic solution for classification rules learning methods. The crucial step in the proposed solution is the privacy-preserving computation of frequencies of a tuple of values, which can ensure each participant's privacy without loss of accuracy. We illustrate the applicability of the method by using it to build the privacy preserving protocol for association rules mining and ID3 decision tree learning.

I. Introduction

Data mining has emerged as a significant technology for gaining knowledge from vast quantities of data. Data mining allows us to analyse personal data or organisational data, such as customer records, criminal records, medical history, credit records, etc. However, analyzing such data creates threats to privacy and thus, might prevent data mining works. The challenge then is whether we can obtain results of mining while still preserve the data secrecy. Privacy preserving data mining (PPDM) techniques have been proposed to address this type of problem [2].

Generally, there are mainly two kinds of privacy preserving data mining approaches: the perturbation-based approach and the cryptography-based approach. The methods based on perturbation (e.g., [1], [3], [23]) have been proved to be efficient, but have a tradeoff between privacy and accuracy. The methods based on cryptography (e.g., [22], [20], [11]) can safely preserve privacy without loss of accuracy, but have high complexity and high communication cost. These privacy preserving data mining methods have been presented for various scenarios in which the general idea is to allow mining datasets distributed across multiple parties, without disclosing each party's private data [2].

Classification is an important data mining technique that has found applications in various areas, such as business, education, and defense. Within the context of privacy preserving data mining, several privacy-preserving classification solutions have been proposed, e.g., [9], [10] and [24]. The goal of these works is for one of the participants to obtain the global classification model from the joint data set of all parties. The requirement of these works is that no information about private data, except in the classification model, will be disclosed.

Each privacy preserving classification solution can be applied in a particular privacy preserving data mining scenario. In [9], [10] and [24], they developed a privacy preserving classification protocols from the vertically distributed data based on a secure scalar product method. These privacy preserving protocols have been applied for learning naive Bayes classification, association rules and decision trees. In [12], the privacy preserving classification was addressed for horizontally distributed data by computing the secure sum of all local frequencies of participating parties. Much more complicated solutions have been proposed for the fully distributed setting [5], [22]. These works aimed to allow a miner to learn classification rules from a data set distributed across a large number of users, while preserving privacy of each user's private data.

Some randomization-based solutions proposed in [1], [3], [15], [16] can be used in various distributed data scenarios. The basic idea of these solutions is that every user perturbs its data, before sending it to the miner. The miner then can reconstruct the original data to obtain the mining results with some bounded error. These solutions allow each user to operate independently, and the perturbed value of a data element does not depend on those of the other data elements, but only on its initial value. Although these solutions are high efficient, their use generally involves a tradeoff between privacy and accuracy, i.e., if we require more privacy, the miner loses more accuracy in the data mining results, and vice-versa.

In this paper, we study a privacy preserving classification rules learning in two-dimension distributed data, which is a generalization of both horizontally and vertically distributed

978-0-7695-4213-3/10 $25.00 © 2010 IEEE
DOI 10.1109/KSE.2010.38

data. The generality of this model would be experimental proved to be better suited at practical settings in which data may be mostly, but not completely, vertically or horizontally distributed. The contributions of the work include:

- *The development of a cryptographic approach to privacy preserving classification learning in two-dimension distributed data*. We proposed a protocol for privacy preserving frequency computation. The protocol ensures each parties's privacy without loss of accuracy. In addition, it is efficient and based on centered management model.
- *The applicability of the approach*. To illustrate it we present the design and analysis of the privacy-preserving association rule mining and ID3 learning protocols.

The rest of this paper is organised as follows: Section 2 presents the preliminaries. In Section 3, we introduce a privacy preserving protocol for frequency mining. We also prove the correctness and privacy properties of the protocol and present experimental results of the efficiency. In Section 4, we use the protocol for frequency mining in Section 3 as a building block to design a privacy preserving protocols for association rule mining and ID3 learning. We also evaluate the efficiency of these protocols in Section 4. The last section concludes our research work in this paper.

II. Preliminaries

A. Privacy preserving frequency mining problem

Definition 1. *(Two-dimension distributed data.) A data set DB includes d normal attributes $\{A_1, ..., A_d\}$ with N records. DB is distributed into $n \times m$ blocks, where $2 \leq n < d$ and $2 \leq m < N$.*

The parties P_{ik} with $i = 1, ..., n$ and $k = 1, ..., m$. Each party holds a block DB_{ik} containing information about certain attributes set $A^{(k)}$ and certain records. The DB_{ik} are such that:

- *DB_{ik} and $DB_{i'k}$ $(i \neq i')$ have the same attributes set $A^{(k)}$ but (parts of) different records of DB;*
- *DB_{ik} and $DB_{ik'}$ $(k \neq k')$ have disjoint attributes sets $A^{(k)}$ and $A^{(k')}$ but contain information about the same records of DB;*

We call DB be a two-dimension distributed data set.

Privacy preserving frequency mining problem: In this section, we consider a scenario in which a party plays a role as a miner, which mines the frequency of a tuples of attribute values from DB. Where, DB is called the joint data set of all parties. Our aim is to design the distributed protocols to obtain the frequency while preserving privacy of each party's data. We consider privacy as protecting individual data records as well as protecting the frequency of the local tuples of each party.

Let $S \subset \{1, ..., m\}$ and $1 \leq |S| \leq m$. In general case, we assume that the miner needs to compute the frequency of a data tuple (T) that consists $|S|$ parts indexed in S. Each part T_k $(k \in S)$ consists of some values for some attributes $\in A^{(k)}$. Note that each record RC_j $(j = 1, ..., N)$ is partitioned into m parts; where each part RC_{jk} consists of values for the attributes in set $A^{(k)}$ and is owned by a party P_{ik}. We consider the map of each record j to $|S|$ binary numbers as follows. For each $k \in S$

$$u_{jk} = \begin{cases} 1, & \text{if } T_k \subseteq RC_{jk}; \\ 0, & \text{otherwise.} \end{cases}$$

Thus, DB is mapped to a binary matrix (U) that consists of $|S|$ columns and N rows. Note that if $T \subseteq RC_j$, all elements of the j^{th} row of the binary matrix have 1 that means $\lambda_j = \sum_{k \in S} u_{jk} - |S| = 0$, where each u_{jk} is owned by a P_{ik}.

Therefore, our purpose is to design a protocol that allows the miner learning the number of records j to satisfy $\lambda_j = 0$ while the party does not have to disclose the values u_{jk}.

B. Encryption scheme selection

Our protocol requires each party to send private data to the miner. In order to preserve privacy of each party's data, each party encrypts its data and then send the encrypted data to the miner. The miner executes the protocol to obtain the frequency from the encrypted data. We use the ElGamal cryptosystem [21], a public key cryptosystem, consists of three algorithms: the key generation algorithm, the encryption algorithm, and the decryption algorithm. In the ElGamal cryptosystem, the key generation algorithm generates the parameters (G, q, g, x), where g be a generator for a cyclic group G of order q. x be a private key that is uniformly chosen from $\{0, 1, ..., q-2\}$. The public key is $y = g^x$ that are publicly known.

The encryption of message $m \in$ the clear-text space M defined as:

$$E(m, r) = (my^r \mod q, g^r \mod q) = (C^{(1)}, C^{(2)}) = C$$

where r is uniformly chosen from $\{0, 1, ..., q-2\}$.

Decryption of the cipher-text C with the private key x can be executed by computing $m = C_1(C_2^x)^{-1}$.

The encryption scheme is based on the computational difficulty of computing the discrete logarithm of group G. Given a prime number q, a generator g and an expression $g^a \mod q$, it is computationally difficult to find the value a. The ElGamal encryption is semantically secure under the Decisional Diffie-Hellman (DDH) Assumption[4]. Thus, the encryption scheme has the indistinguishability property that any two different messages will have different cipher-texts since the random number r can take many different values. The ElGamal encryption has a randomisation property to allow computing a different encryption from a given encryption, M.

C. Computation model

Privacy preserving mining protocols implement among a set of semi-honest parities. One of these parties play the special role, which is called the miner to collect messages from all parties and compute the final result.

We assume that all parties are online, where each party has a communication channel with the miner. To be applicable, we require that the protocol can ensure users' privacy in an environment that doesn't have any secure channels between the party and the miner. In addition, it should not require any communication among the parties.

In our solution, k of the parties act as commodity parties, that is called a moderator. Moderators have the special duty of randomising the data for protecting privacy of each party's data. In order to make the protocol practical, we divide all parties' joint data sets into groups; where each group has m records which is randomly chosen by all parties from their database. The miner computes the frequency from one group at a time. Our requirement is that each party should has its data at a time. In other words, the miner should compute the frequency of the data tuple from a group. In contrast, it should not know which frequency generated from which group record.

We assume that prior to the protocol, each party has obtained the key pairs for the Elgamal encryption scheme, and each party's public key and the miner has known by members in the system.

D. Definition of privacy

The privacy preservation of the proposed protocol is based on the semi-honest security model. In this model, each party participating in the protocol has to follow rules using correct input, and cannot use what it sees during the execution of the protocol to compromise security. A general definition of the secure multi-party computation in the semi-honest model is stated in [6]. This definition was derived to make a simplified definition in the semi-honest model for privacy-preserving data mining in the fully distributed setting scenario [22], [5]. This scenario is similar to our computation model. In this paper, we consider the possibility that some corrupted parties share their data with the miner to derive the private data of the honest parties. One requirement is that no other private information about the honest parties be revealed, except the final result. In our model, information known by parties is no more than information known by the miner. Thus, we do not have to consider the problem in which parties share information with each other.

A distribution ensemble $X = \{X_n\}n \in S$ is a family of probability distributions indexed by some infinite set S of binary strings. We sometimes take $S = \{1^n : n \in \mathbb{N}\}$, in which case the indices in S are viewed as natural numbers.

Definition 2. *(Computational indistinguishability). Two ensembles, $X \stackrel{def}{\equiv} \{X_n\}_{n\in S}$ and $Y \stackrel{def}{\equiv} \{Y_n\}_{n\in S}$ are computational indistinguishable, denoted $X \stackrel{c}{\equiv} Y$ if the following holds: For every polynomial time algorithm, A, and every $c > 0$, there exists an integer N such that for all $n \geq N$.*

$$|Pr(A(X_n) = 1) - Pr(A(Y_n) = 1)| < \frac{1}{n^c}$$

For more general, we assume that there are K parties involved in the frequency mining protocol in which a party plays the role as a miner. Each party P_i has the input d_i, where d_i can be an integer number or the set of the binary numbers. The definition of privacy preserving in semi-honest model is presented as follows:

Definition 3. *Assume that each party P_i has a private set of keys D_i and a public set of keys E_i. A protocol for the above defined frequency mining problem protects each user's privacy against the miner along with t corrupted parties in the semi-honest model if, for all $I \subset \{1, ..., K\}$ such that $|I| = t$, there exists a probabilistic polynomial-time algorithm M such that*

$$\{M(f, [d_i, P_i]_{i\in I}, E_i\} \stackrel{c}{\equiv} \{View_{miner,\{P_i\}_{i\in I}}[d_i, D_i]_{i=1}^{K}\}$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability.

Basically, the definition 2 states that the computation is secure if the joint view of the miner and the corrupted parties during the execution of the protocol can be effectively simulated by a simulator. The miner and the corrupted parties have observed in the protocol using only the result f, the corrupted parties' knowledge, and the public keys. Therefore, the miner and the corrupted parties can not learn anything from f. By the definition, it indicates that there exists a simulator satisfied the above equation.

III. Privacy preserving frequency mining in two-dimension distributed data

A. Privacy preserving frequency mining protocol

Our goal is that the miner should obtain a random permutation of the set ($g^{r_1\lambda_1}$, .., $g^{r_N\lambda_N}$), without knowing each λ_j coming from which parties. Where, r_i is randomly chosen from $[1, p-2]$ by all other parties. Note that if we obtain this result, the miner can be counted the number $\lambda_j = 0$ that is equal with $g^{r_j\lambda_j} = g^0 = 1$. Clearly, when $\lambda_j \neq 0$, $g^{r_j\lambda_j}$ is the random number, the protocol does not leak any other information except the frequency.

To achieve this goal, we use variants of the ElGamal encryption in which it replaces m with g^m [22]. As the result, there are the following properties:

Property 1. The decryption algorithm is not efficient when m is large, however, there exists an efficient algorithm that uses the private key to decide whether a ciphertext decrypts to 1 or (g^0). That is suitable to our algorithms.

Property 2. It has the additive homomorphic property that can be used to perform computation on ciphertexts from different parts.

Property 3. It allows homomorphic computing of constant multiplication that can be performed without decrypting.

In our solution, we assume that each party has a key pair $(x_i, y_i = g^{x_i})$ and there t of the nm parties play the role as moderators. Without loss of generality, we assume in the sequel that the moderators are numbered from 1 to t. In practice, the choice of moderators can be arbitrary that depends on the requirement of privacy and the efficiency of the application.

Define

$$\begin{aligned} x &= \sum_{i=1}^{t} x_i \\ y &= \prod_{i=1}^{t} y_i = g^x \end{aligned}$$

In our protocol, the parties use the public value y as a public key to encrypt their data. Therefore, decrypting these

encryptions needs the private key x, which is not known to any individual party. We call (x, y) be the joint key pair. Basically, the idea of our protocol as follows:

(1) For each value u_{jk} of the binary matrix, the party holding the value u_{jk} encrypts this value using the joint public key of moderators and sends this encryption to the miner. Note that, without the help of all moderators, nobody can decrypt any of this encryption. By additional monomorphic property of Elgamal scheme, the miner connects all encryptions of the binary values u_{jk} corresponding to the row j to obtain the encryption of $\sum_{k\in S} u_{jk}$. It then adds $g^{-|S|}$ to this encryption to obtain the encryption of $\lambda_j = \sum_{k\in S} u_{jk} - |S|$. At the end of this step, the miner obtains the N encryptions of λ_1,..., λ_N.

(2) The miner sends the set of encryptions of λ_j ($j = 1, ..., N$) to moderators. Each moderator i ($i = 1, ..., t$) computes the encryption of each $\lambda_{ij} = r_{ij}\lambda_j$ and sends back to the miner. Where, r_{ij} are non-zero elements of G chosen independently and uniformly at random. The miner connects encryptions λ_{ij} ($i = 1, .., t$) to obtain an encryption of $\lambda'_j = \lambda_j \sum_{i=1}^{t} r_{ij}$. Note that λ'_j has a 0 if and only if the original value λ_j has a 0. On the other hand, all non-zero elements in the original values have been changed to the randomized element. As the result, no extra information is leaked.

(3) The miner together with t moderators execute the t round to permute and randomise the set of the encryptions of $\lambda'_1, ..., \lambda'_N$. Note that ElGamal supports re-randomisation, which means computing a different encryption of M from a given encryption of M. A related operation is a permutation of the order of items. Therefore, it randomly rearrange the order of items. If we re-randomise and permute a sequence of cipher-texts, then we get another sequence of cipher-texts with the same multiset of clear-texts but in a different order. Looking at these two sequences of cipher-texts, the adversary cannot determine any information about the correspondence between the new cipher-text corresponding and the old cipher-text.

(4) Finally, the moderators jointly help the miner to decrypt the received new encryptions, which are in an independent order of the original encryptions. The miner counts how many of the decryptions are equal to 1 (g^0). This number is equal to the frequency of the tuple.

The detailed protocol is implemented in phases as follows:

- Phase 1. Data submission. For $j = 1, ..., N$
  - For each $k \in S$, the party holding u_{jk} encrypts u_{jk} using the public key y :

    $$R_{jk} = (R_{jk}^{(1)}, R_{jk}^{(2)}) = (g^{u_{jk}} y^{\alpha_{jk}}, g^{\alpha_{jk}})$$

    where α_{jk} is chosen randomly from $[0, q-1]$ and after that R_{jk} is sent to the miner.
  - The miner computes:

    $$C_j = (C_j^{(1)}, C_j^{(2)}) = (g^{-|S|} \prod_{k\in S} R_{jk}^{(1)}, \prod_{k\in S} R_{jk}^{(2)})$$

- Phase 2. Encryption randomisation.
  - For $i = 1, ..., t$, the miner sends (C_1,..., C_N) to the moderator i
  - Each moderator i re-randomises the received encryptions: for $j = 1, ..., N$,

    $$R_{ij} = (R_{ij}^{(1)}, R_{ij}^{(2)}) = ((C_j^{(1)})^{r_{ij}}, (C_j^{(2)})^{r_{ij}})$$

    where r_{ij} is chosen uniformly from $[0, q-1]$.
  - For $j = 1, ..., N$, each moderator i sets $C_{ij} = R_{ij}$ and sends back to the miner.
  - The miner computes: For $j = 1, ..., N$,

    $$C_j = (C_j^{(1)}, C_j^{(2)}) = (\sum_{i=1}^{t} C_{ij}^{(1)}, \sum_{i=1}^{t} C_{ij}^{(2)})$$

- Phase 3. Permutation. For $j = 1, ..., t$,
  - At the beginning of this step, the miner sends (C_1,..., C_N) to the moderator i.
  - Each moderator i permutes encryptions: For $j = 1, ..., N$,

    $$R_j = (R_j^{(1)}, R_j^{(2)}) = (C_{\pi_i(j)}^{(1)} y^{\delta_{\pi_i(j)}}, C_{\pi_i(j)}^{(2)} g^{\delta\pi_i(j)})$$

    where δ_{π_i} is an permutation on $\{1, .., N\}$ and δ_j is chosen independently and uniformly from $[0, q-1]$.
  - For $j = 1, ..., N$, each moderator i sets $C_j = R_j$ sends (C_1,..., C_N) back to the miner.

- Phase 4. Frequency computation
  - The miner sends ($C_1^{(2)}$,..., $C_N^{(2)}$) to all moderators.
  - Each moderator i computes: for $j = 1, ..., N$,

    $$C_{j(i)}^{(2)'} = (C_j^{(2)})^{x_i}$$

    and sends $C_{j(i)}^{(2)'}$ to the miner
  - The miner sets $f = 0$ and works as follows: for $j = 1, ..., N$,

    $$d_j = C_j^{(1)} / \prod_{i=1}^{t} C_{j(i)}^{(2)'}$$
    $$if\ d_j = 1\ then\ f = f + 1;$$

  - The miner outputs f

B. The correctness of the protocol

Theorem 1. *If all participants follow the protocol, then the miner's result is a frequency f as description in Section 2.A.*

Proof:

- At the end of the first step, the miner obtains:

$$C_j = (C_j^{(1)}, C_j^{(2)}) = (g^{\sum_{k\in S} u_{jk} - |S|} y^{\sum_{k\in S} \alpha_{jk}}, g^{\sum_{k\in S} \alpha_{jk}}) = (g^{\lambda_j} y^{\theta_j}, g^{\theta_j})$$

Where, denote $\lambda_j = \sum_{k\in S} u_{jk} - |S|$ and $\theta_j = \sum_{k\in S} \alpha_{jk}$

- In Phase 2, the miner receives ($C_1^{(2)}$,..., $C_N^{(2)}$) from Phase 1 and randomises each C_j. It obtains:

$$\begin{aligned} C_j &= (C_j^{(1)}, C_j^{(2)}) = (\sum_{i=1}^{t} C_{ij}^{(1)}, \sum_{i=1}^{t} C_{ij}^{(2)}) \\ &= (\sum_{i=1}^{t} g^{\lambda_j r_{ij}} y^{\theta_j r_{ij}}, \sum_{i=1}^{t} g^{\theta_j r_{ij}}) \\ &= (g^{\lambda_j \sum_{i=1}^{t} r_{ij}} y^{\sum_{i=1}^{t} \theta_j) r_{ij}}, g^{\theta_j \sum_{i=1}^{t} r_{ij}}) \end{aligned}$$

- In Phase 3, the protocol receives the set ($C_1^{(2)}$,..., $C_N^{(2)}$) and permutes this set k times. Thus, the miner obtains: $C_j = (C_{\pi(j)}^{(1)}, C_{\pi(j)}^{(2)})$, where, $\pi(j)$ is the final result of the permutation of the k times.

$$\begin{aligned} C_{\pi(j)}^{(1)} &= g^{\lambda_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)}} y^{\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)}}; \\ C_{\pi(j)}^{(2)} &= g^{\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)}} \end{aligned}$$

Therefore, in Phase 4, the miner obtains

$$\begin{aligned} d_j &= C_j^{(1)} / \prod_{i=1}^{t} C_i^{x_i} \\ &= \frac{g^{\lambda_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)}} y^{(\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)})}}{g^{(\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)}) \sum_{i=1}^{t} x_i}} \\ &= \frac{g^{\lambda_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)}} g^{\sum_{i=1}^{t} x_i (\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)})}}{g^{(\theta_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)} + \delta_{\pi_i(j)}) \sum_{i=1}^{t} x_i}} \\ &= g^{\lambda_{\pi(j)} \sum_{i=1}^{t} r_{i\pi(j)}} \end{aligned}$$

Thus, if $d_j = 1$, C_j is the encryption of g^0 that means $\lambda_j = 0$. It further follows that the tuple occurs at a record of the data set and so the frequency f is increased to 1.

C. The privacy of the protocol

Theorem 2. *The protocol in Subsection 3.A preserves the privacy of the honest parties against the miner and up to $nm - 2$ corrupted parties as long as at least one of honest parties is a moderator.*

Proof: To prove this theorem, we will design a simulator M that simulates the joint view of the miner and the corrupted users by a probabilistic polynomial-time algorithm. Subsequently, this simulator is combined with a simulator for the ElGamal cipher-texts to obtain a completed simulator. To do so, basically we show a polynomial-time algorithm for computing the joint view of the miner and the corrupted parties. The computation of the algorithm is based on what the miner and the corrupted parties have observed in the protocol using only the result f, the corrupted parties information, and the public keys. The algorithm outputs the simulated values for the encryptions generated by a simulator of ElGamal encryptions. Clearly, it suffices to consider the case in which only one moderator (h_1) and one regular party (h_2) are honest. Without loss of generality, we assume that h_1 is the moderator 1. We will show the simulator that its steps follow the protocol's steps.

- Phase 1, for each $R_j \in h_1$ or h_2, M computes

$$\begin{aligned} W_j = (W_j^{(1)}, W_j^{(2)}) &= (\frac{R_j^{(1)}}{(R_j^{(2)})^{\sum_{i=2}^{k}}}, R_j^{(2)}) \\ &= (g^{w_j} g^{\gamma_j}, g^{\gamma_j}) \end{aligned}$$

where w_j denotes the binary value in the row j of the binary matrix owned by h_1 or h_2, and γ_j denotes a random number in $[0, .., p-1]$. Note that Elgamal encryption is semantically secure under the DDH; thus, M can simulates W_j by a random ciphertext of the Elgamal encryption.

- Phase 2, M has to simulate all $C_{1j} = (C_{1j}^{(1)}, C_{1j}^{(2)})$. Note that given $(g^{e_1}, g^{e_2}, g^{re_1}, g^{re_2})$, since the computational indistinguishability follows from the semantic security of ElGamal encryption, which is well known to hold under DDH, $(g^{e_1}, g^{e_2}, g^{re_1}, g^{re_2})$ and $(g^{e_1}, g^{e_2}, g^{e_3}, g^{e_4})$ are not computationally distinguishable. Therefore, M can simulate each C_{1j} by $C'_{1j} = (g^{e_3}, g^{e_4})$, where e_3 and e_4 are randomly and independently chosen in $[0, p-1]$.

- Phase 3, M has to simulate all $R_j = (R_j^{(1)}, R_j^{(2)}) = (C_{\pi_1(j)}^{(1)} y^{\delta_{\pi_1(j)}}, C_{\pi_1(j)}^{(2)} g^{\delta\pi_1(j)})$. M simulates this using a randomly permuted vector of m ElGamal cipher-texts, among these m cipher-texts, the number of encryptions of g^0 should be equal to the output of the protocol, all the remaining cipher-texts should be the random number.

- Phase 4, each message $C_{j(1)}^{(2)'}$ in the protocol can be simulated using:

$$C_{j(1)}^{(2)'} = \frac{C_j^{(1)}}{d_j \prod_{i=2}^{k} C_{j(i)}^{(2)'}}$$

where it is similar to the choosing a uniformly random element of G. This finishes the simulation algorithm.

IV. Complexity analysis

A. Communication analysis

Let us assume that the size of the moderator's key is b bits. Phase 1 requires the transmission of $2|S|N$ encryptions, for a total of $2|S|Nb$ bits. In phase 2, $2tNb$ bits cipher-texts are transmitted. In each iteration of phase 3 , $2Nb$ bits are transmitted, for a total of $2tNb$ bits. Phase 4 transmits tNb bits. Since, the total is $(5t + 2|S|)Nb$.

B. Computational complexity evaluation

In this section, we show the results of the complexity estimation of the protocol and the efficiency measurement of the protocol in practice. In the proposed protocol, the computational cost of all parties in the first phase is $2|S|N$ modular exponentiations. The computational cost of the miner is at most of all $(|S| + t)N$ modular multiplications for the first phase and the second phase, and N modular multiplication inversions for the last phase. Each moderator uses $5N$ modular exponentiations. We conclude that the total computational complexity is $O((|S| + t)N)$ modular exponentiations. Note

$\|S\|$	N				
	200	400	600	800	1000
2	0.22	0.44	0.67	0.88	1.11
3	0.33	0.66	0.97	1.32	1.65
5	0.55	1.08	1.64	2.14	2.75
10	1.2	2.2	3.2	4.3	5.5
20	2.1	4.3	6.5	8.7	10.8

TABLE I
TIME (SECONDS) USED BY ALL PARTIES FOR DATA ENCRYPTION

N	200	400	600	800	1000
Time	0.15	0.32	0.48	0.63	0.8

TABLE II
TIME (SECONDS) USED BY EACH MODERATOR

that these computational costs do not include the overhead of key generation and computing two parameters x and y. However, generating these parameters belongs to the preparation period of the mining process. Therefore, it can be implemented before the protocol is executed without affecting the computation time of the protocol.

For evaluating the efficiency of the protocol in practice, we build an experiment on the privacy preserving frequency mining in C environment, which runs on a laptop with CPU Pentium M 1.8 GHz and 1GB memory. The used cryptographic functions are derived from Open SSL Library.

To measure the computation cost of the frequency mining protocol in worst case, we assume that all parties involve in the protocol except the miner are the moderators. We measure the computation cost of the frequency mining protocol for 10 parties. Before executing the protocol, we generate a pair of keys for each party, with the size of public keys set at 512 bits.

Table 1 illustrates our measurements of all parties's computation time in the submission phase: it is in regard to N and $|S|$, for a typical scenario, where $N = 1000$, $|S| = 20$. The computation time of all parties is about 10.8 seconds. Table 2 illustrates our measurements of a moderator's computation time: it is linear in N and does not depend on t and $|S|$. For a typical scenario where $N = 1000$, the computation time of a moderator is about 0.8 seconds. The miner's computation time: it is linear in N, $|S|$, and t. However, it is very small, it only is $65ms$ when $N = 10000$, $|S| = 20$, and $t = 10$.

V. PRIVACY PRESERVING FOR CLASSIFICATION RULES LEARNING IN TWO-DIMENSION DISTRIBUTED SETTING

A. Privacy preserving association rules mining

1) Association rules and frequent itemset: The association rules mining problem can be formally stated in [17]. Let $I = I_1, I_2, ..., I_d\}$ be the set of all items. Let DB a transaction database, where each transaction T is a set of items such that $T \subseteq I$. Associated with each transaction is a unique identifier, denoted by TID. We say that a transaction T contains X, a set of some items in I, if $X \subseteq T$. The problem is to find the association rules that have an implication of the form $X \Rightarrow Y[s, c]$, where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \phi$. The support s and the confidence c of the rule $X \Rightarrow Y$ are defined as:

$$s = P(X \cup Y) = \frac{T(X \cup Y)}{|DB|}$$

$$c = P(X|Y) = \frac{T(X \cup Y)}{T(Y)}$$

Where $T(X)$ stands for the number of transactions containing the set X in DB and $|DB|$ denotes the total number of transactions in DB. The strong association rules are required to meet a minimum support (s_{min}) and a minimum confidence (c_{min}) defined by the miner.

A set of items is referred as an itemset. An itemset that contains k items is a k-itemset. The support count of an itemset is the number of transactions containing the itemset. The minimum support count is defined as $s_{min}|DB|$. An itemset is frequent if its support count is not less than the minimum support count. Association rule mining is a two-step process: (1) Finding all frequent itemsets; (2) Generating strong association rules from the frequent itemsets.

Agrawal et al. [17], [2] presented the Apriori algorithm to efficiently identify frequent itemsets for boolean association rules. The name of the algorithm is based on the fact that the algorithm uses the Apriori property, i.e., all nonempty subsets of a frequent itemset must also be frequent.

2) Finding a frequent itemset: Assume that the transactions set DB is two-dimension distributed into nm parties as in Section 2.A. Each party $P_{ik}(i = 1, ..n, k = 1, .., m)$ owns DB_{ik} that contains information about certain attribute set $I^{(k)}$ and certain records. Given a candidate set c of the k items, the parties wish to cooperatively find whether or not the candidate set is frequent from the joint transaction set (DB), without disclosing each party's individual transactions and even the local frequent itemsets.

Assume that c is partitioned into parts c_k, where $k \in S$, S$\subseteq \{1, .., m\}$. Each c_k consists of items $\in I^{(k)}$. Note that if c is frequent in DB, it is frequent in at least one horizontal partition DB_i, where $DB_i = DB_{i1} \cup ... \cup DB_{im}$. In addition, if c_k is frequent in DB_i, every c_k ($k \in S$) is frequent in DB_{ik}. Considering a map from each DB_{ik} to a binary number that is done by each P_{ik} as follows:

$$v_{ik} = \begin{cases} 1, & \text{if } c_k \text{ is frequent in } DB_{ik}; \\ 0, & \text{otherwise.} \end{cases}$$

Thus, DB is mapped to the binary matrix $n \times m$ (V). Hence, c is frequent in at least one horizontal partition DH_i as long as at least a row i in V with all elements are 1. As the result, $v_i = \sum_{k \in S} v_{ik} - |S| = 0$. Clearly, using frequency mining can allow the miner to find a random permutation of (g^{v_1}, ..., g^{v_m}). Therefore, the miner can identify whether c is frequent or not, without knowing c be frequent in which DB_i.

3) Finding all frequent itemsets and their support counts: In the classic Apriori algorithm [17]. The key issue is computing the support of an itemset. To find out if a particular itemset is frequent, we count the number of records where the values for all the attributes in the itemset are 1. Thus, the

problem is to compute the frequency of values' tuples that all values in the tuple are 1. The privacy preserving protocol for finding frequent itemsets and support counts follows Apriori algorithm as below:

1: {Finding an item 1-Itemsets is frequent}
2: $C_1 = I$
3: The miner sets $L_1 = \emptyset$
4: **for** each $c \in C_1$ **do**
5: The miner uses the frequency mining protocol to identify whether or not c is frequent
6: **if** c is frequent **then**
7: The miner does: $L_1 = L_1 \cup c$
8: Let $c \in I^{(k)}$, the miner broadcasts the requirement for computing $Support(c)$ to all P_{ik} ($i \in \{1, ..., n\}$).
9: All parties involve in frequency mining protocol to compute $Support(c)$
10: **end if**
11: **end for**
12: **for** $\{p = 2;\ L_{p-1} \neq \emptyset\ p++\}$ **do**
13: The miner does: C_p=Approri-gen(L_{p-1})
14: **for** each $c \in C_p$ **do**
15: The miner uses the frequency mining protocol to identify whether or not c is frequent
16: **if** c is frequent **then**
17: The miner does: $L_p = L_p \cup c$
18: Let c consists of items partitioned into the sets c_k, where $k \in S$, $S \subset \{1, ..., m\}$, the miner broadcasts the requirement for computing $Support(c)$ to all P_{ik} ($k \in S$).
19: The miner and the parties involve in frequency mining protocol to compute $Support(c)$
20: **end if**
21: **end for**
22: **end for**

4) Analysis of protocol:

Statement 1. *(Correctness). If all participants follow the protocol, then the miner's result is the frequent itemsets and the support count of each frequent itemset*

*Proof:*Candidate itemsets are generated by the Apriorigen procedure. The correctness of that procedure has proved [17]. The C_p sets are generated correctly as long as the input to the procedure is correct. We show by induction that the L_p sets are generated correctly. At steps $1 - 9$ with $p = 1$, L_1 is correctly generated by the frequency mining protocol. Assume that L_{p-1} has been correctly generated, then C_p is correctly generated by Apriorigen procedure. Since frequency ming protocol is correct, the support count of each $c \in C_{k-1}$ is computed correctly. Hence, L_p is generated correctly from L_{p-1}. The entire frequent itemsets and the support counts gives correct results.

Statement 2. *(Privacy.) The protocol preserves the privacy of the honest users against the miner and up to $nm-2$ corrupted parties as long as there is at least an honest be the moderator.*

Proof: Since all support count computations and frequent itemsets identification are done independently using frequency mining. This statement follows immediately from Theorem 2.

5) Evaluation of complexity: The communication analysis critically depends on the number of frequency computations called. We incur the cost of privacy preserving frequency mining for each call. Let r be the maximum size of a frequent itemset, and let $C_i(i = 1, ..., r)$ and $L_i(i = 1, ..., r)$ represent the number of candidate itemsets and the found number of frequent itemsets at each round, the total communication consists of cost of finding frequent and the cost of the support counts computation that is: $C = \sum_{i=1}^{r}(5t + 2C_i)nb + (5t + 2L_i)Nb$ bits. Similarity, the computational complexity is $O(\sum_{i=1}^{r}((t + C_i)n + (t + L_i)N)$ modular exponentiations.

B. Privacy preserving learning of ID3 tree in two-dimension distributed data

Using the primitive of proposed privacy-preserving frequency mining, we can learn ID3 trees in two-dimension distributed data without loss of accuracy. The miner's algorithm has the same complexity as the original ID3 tree algorithm, except for an additional linear overhead factor. Which has a value determined by the number of times frequency mining protocol using to compute gain.

1) ID3 decision tree learning: we firstly present a brief review of ID3 decision trees. An ID3 tree is a rooted tree containing nodes and edges. Each internal node is a test node and corresponds to an attribute. The edges going out of a node correspond to the possible values of that attribute. The ID3 algorithm works as follows. The tree is constructed top-down in a recursive fashion. At the root, each attribute is tested to determine how well it alone classifies the samples. The best attribute is then chosen and the samples are partitioned according to this attribute. The ID3 algorithm is then recursively called for each child of this node, using the corresponding subset of data.

Thus, major problem of the algorithm is choosing the best attribute that can achieve the maximum information gain at each node. Clearly, the problem of choosing the best attribute can be reduced to computing entropies that require computation of the frequency of tuples of values[24].

2) Protocol of privacy-preserving ID3 tree learning: Let DB be a data set that has the set of (non-class) attributes $A = \{A_1, .., A_l\}$ and V the class attribute. Without loss of generality, we assume that all attributes have the same domain size d: $\{a_1, ..., a_d\}$. DB is two-dimension distributed into nm parties as in Section 2.A. Each party $P_{ik}(i = 1, ..n, k = 1, .., m)$ owns DB_{ik}. There are n parties P_{im} $(i = 1, ..n)$ holding the classification attribute V. The parties wish to cooperatively build the $ID3$ decision tree classifier from the joint data set of all parties, without disclosing each party's individual transactions and even the number of the local records. Assume that parties have set a system as the computation model described in Section 3. In this section, we use frequency protocol as the primitive to design the privacy protocol for building decision tree following the ID3

Algorithm:

PrivacyPreservingID3(A, V, DB)

1. If A is empty, return a leaf-node with the class value assigned to the most of all transactions in DB.
2. Use the privacy preserving method to count the number of records with each class label. If DB consists of records which have the same class label v, return a leaf node with v.
3. Otherwise:
   - Determine the best attribute A_i for DB using the privacy-preserving method.
   - For $A_i = \{a_1, ..., a_d\}$, let $DB(a_1)$,..., $DB(a_d)$ be a partition of DB that every record in $DB(a_j)$ has attribute value a_j .
   - Return a tree whose root is labeled A_i, the root has outgoing edges labeled a_1,..., a_d s.t. each edge a_j goes to the tree $PrivacyPreservingID3(A - A_i, V, DB(a_j))$.

3) Analysis of protocol: The communication/computation depends on the number of records, number of vertically partition, number of attributes, number of attribute values per attribute, number of classes and complexity of the tree. For a rough analysis, the cost of computation involves in terms of the time number of frequency mining protocol called to build the tree. Assume that there are r nodes in final classification tree. In total, each node needs $p(1 + dl)$ the calls of frequency mining protocol. All node of the tree need $pr(1 + dl)$ the frequency computation. Therefore, in total the entire classification process will require O($prdlN(t + |S|)$) encryptions and O($prdlN(t + |S|)b$) bits communication.

VI. Conclusion

In this paper, we proposed a method for privacy-preserving classification learning in two-dimension distributed data, which has not been investigated previously. Basically, the proposed method is based on the ElGamal encryption scheme and it ensures strong privacy without loss of accuracy. We illustrated the applicability of the method by applying it to design the privacy preserving protocol for some learning methods such as association rules mining, decision tree learning. We conducted experiments to evaluate the complexity of the protocols. The experimental results showed that the protocols are efficient and practical.

References

[1] A. Evmievski, R. Srikant, R. Agrawal and J. Gehrke, Privacy preserving mining of association rules, In Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, pp. 217-228, 2002.

[2] C. C. Aggarwal, P.S. Yu (Eds.), Privacy-Preserving Data Mining: Models and Algorithms, Series: Advances in Database Systems, Springer, Vol. 34, 2008.

[3] D. Agrawal and C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, In Proc. ACM SIGMOD, pp. 247-255, 2001.

[4] D. Boneh, The decision Diffe-Hellman problem, In ANTS-III, Vol. 1423 of LNCS, pp. 48-63, 1998.

[5] F. Wu, J. Liu, and S. Zhong, An efficient protocol for private and accurate mining of support counts, Pattern Recognition Letters, Vol. 30, Issue 1, 1, pp. 80-86, 2009.

[6] O. Goldreich, Foundations of Cryptography. Basic Tools, Vol. 1. Cambridge University Press, 2001.

[7] H. Martin and S. Kazue, Efficient receipt-free voting based on homomorphic encryption, In Proc. of Advances in Cryptology-Eurocrypt, 2000.

[8] J. Benaloh and D. Tuinstra, Receipt-free secret- ballot elections (extended abstract), In Proc. of the 26th Annual ACM Symposium on Theory of Computing, ACM Press, pp. 544-553, 1994.

[9] J. Vaidya and C. Clifton, Privacy preserving naive Bayes classifier for vertically partitioned data, In Proc. of the 2004 SIAM Conference on Data Mining, 2004.

[10] J. Vaidya and C. Clifton, Privacy preserving association rule mining in vertically partitioned data, In Proc. of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 639-644, 2002.

[11] Luong, T.D., Ho, T.B. (2010). Privacy Preserving Frequency Mining in 2-Part Fully Distributed Setting, IEICE Trans. Information Systems (to appear).

[12] M. Kantarcoglu and J. Vaidya, Privacy preserving naive Bayes classifier for horizontally partitioned data, In IEEE ICDM Workshop on Privacy Preserving Data Mining, pp. 3-9, 2003.

[13] M.Freedman, K.Nissim, and B.Pinkas. Efficient private matching and set intersection. In Proc. of Eurocrypt, Vol. 3027 of LNCS, Springer-Verlag, pp. 1-19, 2004.

[14] R. Agrawal and R. Srikant, Privacy preserving data mining, In Proc. of ACM SIGMOD Conference on Management of Data, pp. 439-450, 2000.

[15] R. Agrawal, R. Srikant and D. Thomas, Privacy preserving OLAP, In Proc. of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD '05. ACM, pp. 251-262, 2005.

[16] R. Agrawal and R. Srikant, Privacy-preserving data mining, In Proc. of the ACM SIGMOD Conference on Management of Data, ACM Press, pp. 439-450, 2000.

[17] R.Agrawal, T.Imielinski, and A.Swami. Mining association rules between sets of items in large databases. In Proc. of the 1993 ACM SIGMOD international Conference on Management of Data, 207-216, 1993.

[18] S. Zhong, Z. Yang, and T. Chen, k-Anonymous data collection, Journal of Information Sciences, Vol. 179, Issue 17, pp. 2948-2963, 2009.

[19] V.S. Verykios, E. Bertino, I.N. Fovino, L.P. Provenza, Y. Saygin and Y. Theodoridis, State-of-the-art in privacy preserving data mining, ACM SIGMOD Record, Vol. 3, No. 1, pp. 50-57, 2004.

[20] Y. Lindell, B. Pinkas, Privacy preserving data mining, In: Advances in Cryptology Crypto2000, Vol. 1880 of LNCS, Springer-Verlag, pp. 36-53, 2000.

[21] Y. Tsiounis and M. Yung, On the security of ElGamal-based encryption, In Public Key Cryptography'98, Vol. 1431 of LNCS, pp. 117-134, 1998.

[22] Z. Yang, S. Zhong, R.N. Wright, Privacy-preserving classification of customer data without loss of accuracy, In Proc. of the 2005 SIAM International Conference on Data Mining (SDM), pp. 21-23, 2005.

[23] W. Du and Z. Zhan, Using randomized response techniques for privacy preserving data mining, In Proc. of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, pp. 505510, 2003.

[24] W. Du and Z. Zhan, Building decision tree classifier on private data, In Proc. of IEEE International Confonference on Privacy, Security, and Data Mining, pp. 1-8, 2002