3L3-OS-06b-1

Collusion-Resistant Privacy-Preserving Data Mining

Bin Yang*1 Hiroshi Nakagawa*2

*1 Graduate School of Information Science and Technology, The University of Tokyo
*2 Information Technology Center, The University of Tokyo

Numerous methods have been proposed for privacy-preserving data mining (PPDM). We considered a general problem in this issue – multi-party secure computation of functions on secure summations of data spreading around multiple parties. Most of the related works are based on an assumption that semi-honest is and collusion is not present. In other words, some parties may collude and share their record to deduce the private information of other parties. In order to solve above collusion problem, a secure computation method that entails a high level of collusion-resistance have been proposed. Unfortunately, the private inputs of some parties may be inferred because unnecessary information is disclosed in the process of this method. In this paper, we will improve this method, so that the final result is directly computed without any intermediate information being revealed. Moreover, this method can be used to securely compute almost any kind of function on secure summations.

1. Introduction

In recent years, increased concern over personal information and privacy protection has led to the development of a number of techniques such as randomization and homomorphic encryption. Such techniques have been suggested to ensure that data mining can be performed while maintaining the preservation of private information protection. We consider privacy-preserving data mining in which m parties collaborate to compute some function of their input, and no party in the system will learn any additional information separate from the functions and the parties' own input.

Large numbers of conclusions in privacy-preserving data mining have been obtained by researchers, most of which are based on an assumption that each party is semi-honest. In particular, a party is deemed semi-honest when the party follows the protocol properly with the exception that it keeps a record of all its intermediate computation results and then tries to deduce further information in addition to the protocol result. Moreover, researchers also assume that every party does not collude or share its record with any other party.

Consider an example in which m parties collaborate to securely compute the average of all data on them. Each party has the inputs ${}^{i}s$ and ${}^{i}n$, the summation of all data on that party and the number of those data respectively. The goal is to compute $r = ({}^{1}s + {}^{2}s + \cdots + {}^{m}s)/({}^{1}n + {}^{2}n + \cdots + {}^{m}n)$, the overall average of the data in all parties. A straightforward method is securely computing the numerator $s = {}^{1}s + {}^{2}s + \cdots + {}^{m}s$ and the denominator $n = {}^{1}n + {}^{2}n + \cdots + {}^{m}n$ respectively, and then computing the average $r = \frac{s}{n}$. Unfortunately, since everyone knows the values of s and n, if party $2, 3, \cdots, m$ collude, they can infer ${}^{1}s$ and ${}^{1}n$ easily. That means the inputs of party 1 are revealed to others.

In order to prevent such a thing from happening, a protocol against collusions is necessary for each party. Here, the collusion problem is that a subset of the participants, a coalition, might get together after the execution of the protocol and attempt to deduce additional information from non-coalition parties.

The computation of the function over secure summations of data spreading around multiparty is a general problem in distributed computation and privacy-preserving data mining. We have proposed a method [7] enhancing the security against party collusion, i.e. satisfying full-privacy. It is also an efficient one with a running time of $O(m)$ in the case of full-privacy. However, since intermediate messages are exchanged between parties throughout this method, these messages may be used to deduce private information of other parties. In this paper, we will improve this method so that no private information is disclosed.

Contact: Bin Yang, Rakuten Institute of Technology, Rakuten Inc., yangbin@r.dl.itc.u-tokyo.ac.jp

2. Related Works

There have been many proposals in the area of privacy-preserving data mining recently. Following is an overview of the most pertinent proposals.

In 2005, Jha et al. [3] proposed an algorithm to securely compute the weighted average problem (WAP) for the 2-party privacy-preserving K-means algorithm. Mert Ozarar et al. [4] extended WAP to multi-party case. In Mert's method, if some parties collude, they can easily deduce the private information of other parties. Vaidya et al. [6] proposed another solution of computation of the ratio of (two) summations of data spreading around m parties to solve the problem of privacy-preserving naive Bayes. The final result can be securely computed without revealing any private information if there is no collusion. Unfortunately, if some of parties collude, privacy may be revealed.

3. Preliminaries

Homomorphic encryption is widely used in PPDM. Given a key pair (sk, pk) and a message m in Z_{N^*}, $c = E_{pk}(m)$ denotes an encryption of the plaintext m, and $d = D_{sk}(c)$ denotes the decryption of the cryptograph c. In particular, we call such an encryption system a homomorphic en-

Table 1: Secure Product of Summation Protocol (SPoS).

Input: Party i have private inputs ${}^{i}x, {}^{i}z$.
Output: Party i privately obtains number ${}^{i}p$.
begin
  for each $j \in 1..m$ $(j \neq i)$
    Running random share protocol with Party j,
    Party i generates random numbers ${}^{i}\delta_j$ and ${}^{i}\epsilon_j$,
    Party j generates random numbers ${}^{j}\delta_i$ and ${}^{j}\epsilon_i$,
    s.t. ${}^{i}\delta_j + {}^{j}\epsilon_i \equiv {}^{i}x^{j}z$ and ${}^{j}\delta_i + {}^{i}\epsilon_j \equiv {}^{j}x^{i}z$.
  end for
  $${}^{i}p = {}^{i}x^{i}z + {}^{i}\delta_1 + \cdots + {}^{i}\delta_{i-1} + {}^{i}\delta_{i+1} + \cdots + {}^{i}\delta_m + {}^{i}\epsilon_1 + \cdots + {}^{i}\epsilon_{i-1} + {}^{i}\epsilon_{i+1} + \cdots + {}^{i}\epsilon_m.$$
end

cryption, if there are two operations + and $\cdot$ satisfying the condition for any m_1 and m_2:

$$E_{pk}(m_1 + m_2) \equiv E_{pk}(m_1) \cdot E_{pk}(m_2). \quad (1)$$

The following equation is straightforward.

$$E_{pk}(mk) \equiv E_{pk}(m)^k. \quad (2)$$

There are several classical homomorphic cryptosystems. In this paper, we employ an additively homomorphic cryptosystem - the Paillier [5].

Goethals et al. [2] proposed a two-party random share protocol using homomorphic encryption. Two parties, Alice and Bob, each respectively having an n-dimensional vector, $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ and $\mathbf{y} = (y_1, y_2, \cdots, y_n)$, share the dot product $\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} x_i y_i$. More specifically, Alice and Bob securely generate random numbers s_a and s_b, respectively, such that $s_a + s_b = \mathbf{x} \cdot \mathbf{y}$. This protocol is secure because both parties only obtain a random number, s_a and s_b, respectively. Alice knows nothing about $\mathbf{y}$ if she does not know s_b, and so does Bob.

4. Collusion-Resistant Algorithms

We notice that all the problems in section 2. are the computation of secure ratio of summations [3, 4, 6], while those proposed methods do not deal with collusion problem. [7] proposed a collusion-resistant protocol for this problem. This protocol satisfied full-private ($(m-1)$-private), where m is the number of parties.

4.1 Secure Product of Summations Protocol

A Secure Product of Summation Protocol [7] (Table 1) is proposed to privately compute the product of secure summations. From Table 1, we have the following equation.

$$p = {}^{1}p + {}^{2}p + \cdots + {}^{m}p = \left(\sum_{i=1}^{m} {}^{i}x\right)\left(\sum_{i=1}^{m} {}^{i}z\right). \quad (3)$$

Therefore, if each party publishes its result ${}^{i}p$, every one can computes p in (3) by summing up all ${}^{i}p$s.

Table 2: Secure Ratio of Summation Protocol (SRoS).

Input: Party i have ${}^{i}x, {}^{i}y$.
Output: All parties obtain $r = \frac{\sum_{i=1}^{m} {}^{i}x}{\sum_{i=1}^{m} {}^{i}y}$.
begin
  Generate a real number randomly ${}^{i}z$.
  Running $SPoS$ protocol with other parties
  to generate and ${}^{i}p$ and ${}^{i}p'$, s.t.
  $$p = \sum_{i=1}^{m} {}^{i}p = \left(\sum_{i=1}^{m} {}^{i}x\right)\left(\sum_{i=1}^{m} {}^{i}z\right).$$
  $$p' = \sum_{i=1}^{m} {}^{i}p' = \left(\sum_{i=1}^{m} {}^{i}y\right)\left(\sum_{i=1}^{m} {}^{i}z\right).$$
  All parties compute $r = \frac{p}{p'} = \frac{\sum_{i=1}^{m} {}^{i}x}{\sum_{i=1}^{m} {}^{i}y}$.
end

4.2 Secure Ratios of Summations Protocol

Using SPoS protocol, [7] derived a Secure Ratios of Summation Protocol (SRoS), as Table 2, to compute the secure ratios of summations of data spreading around m parties.

4.3 Performance

The privacy of random share protocol has been guaranteed in [2]. A measure, t-private, is introduced to evaluate the level of security in term of collusion-resistance.

Definition 1 *(t-Private). Let $f : (\{0,1\}^*)^m \to \{0,1\}^*$ denote an m-input, single-output function, and let Π be an m-party protocol for computing f. We denote the party input sequence by $x = (x_1, x_2, \cdots, x_m)$. For the subset of $[m]$, $\mathbf{I} = \{i_1, i_2, \cdots, i_t\}$, we let $VIEW_{\mathbf{I}}^{\Pi}$ denote the joint protocol view of parties in $\mathbf{I}$, and OUT^{Π} denote the protocol output. For $0 < t < m$, we say that Π is a t-private protocol for computing f if there exists a probabilistic polynomial-time algorithm S, such that, for every subset of $[m]$, $\mathbf{I}$, with $|\mathbf{I}| \leq t$ and every x, it holds that*

$$< S(\mathbf{I}, x_{\mathbf{I}}, f(x)), f(x) > \stackrel{C}{=} < VIEW_{\mathbf{I}}^{\Pi}(x), OUT^{\Pi}(x) >$$

where $\stackrel{C}{=}$ denotes computational indistinguishability. A protocol is called private if it is 1-private. A protocol is called fully private if it is (m-1)-private [1].

In other words, a protocol is called t-private if no collusion containing at most t parties can get any additional information from its execution. Following theorem declared that SPoS protocol has the highest level of collusion-resistance, i.e., for any party, say Party i, even if all other parties collude, they can infer nothing about Party i.

Theorem 1 (Security of SPoS). *SPoS protocol is fully private ((m-1)-private) where m is the number of parties.*

The procedure in SPoS protocol includes many operations in which each party interacts with each other party. Since the number of the parties is m, the total cost is $O(m^2)$. The following theorem shows that the running time of these operations is just $O(m)$, not $O(m^2)$, because all parties perform these operations in parallel.

Theorem 2 *m parties collaborate to achieve a process in which each party achieves the same operation with each other party. Suppose that the running time of such an operation is T and no party can achieve that operation with more than one party simultaneously. The total cost is then $m(m-1)T/2$, and the running time is $(m-1)T$ when m is even, mT when m is odd.*

5. Security Enhancement

Although SPoS is a fully private protocol, we also need considering the privacy of SRoS protocol. More specifically, since publishing the secure product of summations is not necessary, we will now discuss the information leakage due to the publication of the secure product and then enhance the security of SRoS protocol.

5.1 Privacy Problem

Suppose there are m parties, and each party i ($\in [m]$) has a private input ${}^ix$ and a random number ${}^iz$, where ${}^ix, {}^iz \in (0,1)$. All of these m parties securely compute the value of $p = x \cdot z$ ($x = \sum_{i=1}^m {}^ix$ and $z = \sum_{i=1}^m {}^iz$) by using SPoS protocol. We assume that a coalition of $m-1$ parties, $\mathbf{I} = [m] - \{1\} = \{2, 3, \cdots, m\}$, is interested in obtaining the private value of party 1, ${}^1x$. Since $p = x \cdot z$, we have

$$x' + {}^1x = x = \frac{p}{z} = \frac{p}{z' + {}^1z}, \tag{4}$$

where $x' = \sum_{i=2}^m {}^ix$ and $z' = \sum_{i=2}^m {}^iz$. Therefore,

$${}^1x = \frac{p}{z' + {}^1z} - x'. \tag{5}$$

Coalition $\mathbf{I}$ has the values of ${}^2x, {}^3x, \cdots, {}^mx$, ${}^2z, {}^3z, \cdots, {}^mz$, p, but does not know the value of ${}^1z$. Nevertheless, it knows that $0 < {}^1z < 1$, which implies that

$${}^1x \in \left(\frac{p}{z'+1} - x', \frac{p}{z'} - x' \right). \tag{6}$$

Now, let us discuss the length of this interval. We consider a specific case in which ${}^ix \approx 0$ and ${}^iz \approx 1$ ($\forall i \in \mathbf{I}$). Hence, $x' = {}^2x + {}^3x + \cdots + {}^mx \approx 0$ and $z' = {}^2z + {}^3z + \cdots + {}^mz \approx m-1$. Without loss of generality, we assume that $m - 1 - \epsilon < c' < m - 1$ and x' is efficiently small so that $x' < \frac{m}{z'+1} - 1$ (Notice that $z' + 1 < m$). Therefore, $p = (z' + {}^1z)(x' + {}^1x) < (z'+1)(x'+1) < m$. Hence, the interval in (6) is

$$\begin{aligned} &\left(\frac{p}{c'} - x'\right) - \left(\frac{p}{c'+1} - x'\right) = \frac{p}{c'(c'+1)} < \frac{m}{c'(c'+1)} \\ &< \frac{m}{(m-1-\epsilon)(m-\epsilon)} < \frac{1}{m-(1+2\epsilon)} \approx \frac{1}{m-1}. \end{aligned} \tag{7}$$

That is to say, the length of this interval is near $\frac{1}{m-1}$ if ϵ is efficiently small. However, m is generally quite large, so $\frac{1}{m-1}$ is quite small. As a result, coalition $\mathbf{I}$ can locate ${}^1x$ in a small interval (6). I.e., the privacy of ${}^1x$ is revealed.

5.2 Security Improvement

This privacy problem is due to the publication of the secure product of summations. To avoid this problem, we will improve our algorithm so that no intermediate information is published. Our basic idea is converting the objective function on secure summations into a polynomial of these summations so that the objective function can be directly computed by using the SPoS protocol. Now, let us review the SRoS problem. In this problem, each party i($\in [m]$) has two inputs, ${}^ix$ and ${}^iy$. These m parties attempt to securely compute the value of $r = \frac{\sum_{i=1}^m {}^ix}{\sum_{i=1}^m {}^iy}$ without revealing any secure input to other parties. It can be rewritten as follows.

$$r = ({}^1x + {}^2x + \cdots + {}^mx) \cdot \frac{1}{{}^1y + {}^2y + \cdots + {}^my}. \tag{8}$$

Let $y = {}^1y + {}^2y + \cdots + {}^my \in (0, m)$. Suppose $\tilde{y} := \frac{y}{m} \in (0,1)$. Then, the Taylor series of the second item is

$$\begin{aligned} \frac{1}{y} &= \frac{1}{m} \cdot \frac{1}{\tilde{y}} = \frac{1}{m} \cdot \sum_{t=0}^{\infty} (1-\tilde{y})^t = \frac{1}{m} \cdot \sum_{t=0}^{\infty} \left(\frac{m-y}{m}\right)^t \\ &= \frac{1}{m} \cdot \sum_{t=0}^{\infty} \left(\sum_{i=1}^m \frac{1 - {}^iy}{m} \right)^t = \frac{1}{m} \cdot \sum_{t=0}^{\infty} \left(\sum_{i=1}^m {}^i\bar{y} \right)^t, \end{aligned} \tag{9}$$

where ${}^i\bar{y} = \frac{1 - {}^iy}{m} \in (0, \frac{1}{m})$ is a private input of party i. Therefore,

$$\begin{aligned} r &= \frac{{}^1x + {}^2x + \cdots + {}^mx}{m} \cdot \sum_{t=0}^{\infty} \left({}^1\bar{y} + {}^2\bar{y} + \cdots + {}^m\bar{y}\right)^t \\ &\approx \frac{{}^1x + {}^2x + \cdots + {}^mx}{m} \cdot \sum_{t=0}^{T} \left({}^1\bar{y} + {}^2\bar{y} + \cdots + {}^m\bar{y}\right)^t. \end{aligned} \tag{10}$$

In the following, we will randomly share r (10) directly by performing the SPoS protocol repeatedly. Throughout our approach, no intermediate information is revealed other than random numbers.

Suppose ${}^ix$ in (10) is a real number smaller than 1 and is accurate to several decimal places. It can then be represented as a ratio of two integers, i.e., ${}^ix = \frac{{}^ia}{A}$, where A is a constant. For example, if ${}^ix$ is accurate to 6 decimal places, then ${}^ix = \frac{{}^ia}{A}$, where $A = 10^6$, and ${}^ia < A$. Similarly, we also suppose ${}^i\bar{y} = \frac{{}^ib}{B}$, where ${}^ib < \frac{B}{m}$ (since ${}^i\bar{y} \in (0, \frac{1}{m})$). Therefore, (10) becomes the following formulation.

$$\begin{aligned} r &\approx \frac{1}{m} \cdot \frac{{}^1a + \cdots + {}^ma}{A} \cdot \sum_{t=0}^{T} \left(\frac{{}^1b + \cdots + {}^mb}{B} \right)^t \\ &= \frac{{}^1a + \cdots + {}^ma}{mAB^T} \cdot \sum_{t=0}^{T} B^{T-t} \left({}^1b + \cdots + {}^mb\right)^t. \end{aligned} \tag{11}$$

Now, we randomly share (11). We first share the item $({}^1b + {}^2b + \cdots + {}^mb)^t$ ($t = 2, 3, \cdots, T$) by using the SPoS protocol (Table 1) iteratively. Suppose the random shares of $({}^1b + {}^2b + \cdots + {}^mb)^{t-1}$ are ${}^1r_{t-1}, {}^2r_{t-1}, \cdots, {}^mr_{t-1}$, i.e.,

$${}^1r_{t-1} + {}^2r_{t-1} + \cdots + {}^mr_{t-1} = ({}^1b + \cdots + {}^mb)^{t-1}. \tag{12}$$

Then, the random shares of $({}^1b + {}^2b + \cdots + {}^mb)^t$ can be generated by using the SPoS protocol by regarding ${}^1r_{t-1}, {}^2r_{t-1}, \cdots, {}^mr_{t-1}, {}^1b, {}^2b, \cdots, {}^mb$ as its inputs, i.e.,

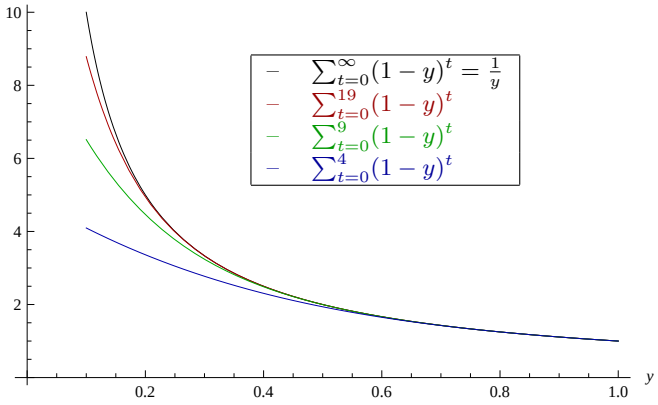

Figure 1: Taylor series of $\frac{1}{y}$.

$$\begin{aligned} &{}^1r_t + {}^2r_t + \cdots + {}^mr_t \\ &= ({}^1b + {}^2b + \cdots + {}^mb)({}^1r_{t-1} + \cdots + {}^mr_{t-1}) \\ &= ({}^1b + {}^2b + \cdots + {}^mb)^t. \end{aligned} \tag{13}$$

Using the shares of $({}^1b + {}^2b + \cdots + {}^mb)^t$ for $t = 2, 3, \cdots, T$, r can be written as

$$r \approx \frac{{}^1a + \cdots + {}^ma}{mAB^T} \cdot \sum_{t=0}^{T} B^{T-t} \left({}^1r_t + \cdots + {}^mr_t\right). \tag{14}$$

Each party i privately computes ${}^ir = \sum_{t=0}^{T} B^{T-t} \cdot {}^ir_t$. (14) then becomes

$$r \approx \frac{{}^1a + \cdots + {}^ma}{mAB^T} \cdot \left({}^1r + \cdots + {}^mr\right). \tag{15}$$

Last, they perform the SPoS protocol for (15), so that

$${}^1s + \cdots + {}^ms = ({}^1a + \cdots + {}^ma) \cdot \left({}^1r + \cdots + {}^mr\right). \tag{16}$$

Here, each ${}^is$ is privately held by party i. If every party publishes its ${}^is$, every one can compute r as follows.

$$r \approx \frac{1}{mAB^T} \cdot ({}^1s + {}^2s + \cdots + {}^ms). \tag{17}$$

In this method, we only computed the T items in the Taylor series. The Taylor series converges to the original function when $T \to \infty$. Therefore, the larger the T, the more accurate the result of the Taylor series. Figure 1 shows the precisions of Taylor series with respect to different values of T. Therefore, it is necessary to perform the SPoS protocol for T times. Accordingly, the running time is also T times longer than the original SRoS protocol.

Moreover, since ${}^ia < A$, and ${}^ib < \frac{B}{m}$, we have

$$\begin{aligned} &{}^1a + {}^2a + \cdots + {}^ma < mA, \\ &B^{T-t}\left({}^1b + {}^2b + \cdots + {}^mb\right)^t < B^T. \end{aligned} \tag{18}$$

Therefore,

$$({}^1a + \cdots + {}^ma) \cdot \textstyle\sum_{t=0}^{T} B^{T-t}\left({}^1b + \cdots + {}^mb\right)^t < mTAB^T. \tag{19}$$

However, ${}^1s, {}^2s, \cdots, {}^ms$ are the shares of (19), so we should choose a modulus, N, for the encryption, such that $N > mTAB^T$.

5.3 Generalization

We showed that the ratio of secure summations can be securely computed by using the above approach. In fact, the secure computation for any infinitely differentiable function on secure summations can be performed by using this method since an arbitrary infinitely differentiable function can be converted to a Taylor series.

6. Conclusion

In this paper, we summed up several problems in privacy-preserving data mining into a formal problem of secure computation for the product of secure summation. We discussed the privacy of the proposed protocol, Secure Product of Summations. This protocol satisfies the highest level of collusion resistance – full-privacy, and therefore is more secure than other related methods. In addition, we also analyzed the privacy leakage due to the publication of the intermediate massages and improved the privacy of this protocol. Since the secure computation of functions on secure summations is a general issue in privacy-preserving data mining, our proposed protocol can be applied to solve many of the problems in this field.

References

[1] Yvo Desmedt, Josef Pieprzyk, Ron Steinfeld, and Huaxiong Wang, *On secure multi-party computation in black-box groups*, CRYPTO, 2007.

[2] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikinen, *On private scalar product computation for privacy-preserving data mining*, ICISC, 2004.

[3] Somesh Jha, Luis Kruger, and Patrick McDaniel, *Privacy preserving clustering*, ESORICS, 2005.

[4] Mert Ozarar and Attila Ozgit, *Secure multiparty overall mean computation via oblivious polynomial evaluation*, International Conference on Security of Information and Networks, 2007.

[5] Pascal Paillier, *Public-key cryptosystems based on composite degree residuosity classes*, EUROCRYPT, 1999.

[6] Jaideep Vaidya, Murat Kantarcioglu, and Chris Clifton, *Privacy-preserving naive bayes classification*, VLDB, 2008.

[7] Bin Yang, Hiroshi Nakagawa, Issei Sato, and Jun Sakuma, *Collusion-resistant privacy-preserving data mining*, KDD, 2010.