International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue-01
January 2018

A Survey: Reliable Mining of Association Rules in Distributed Databases

Md Ahmed Aziz & Dr Asma Parveen
[1]M. Tech Student KBNCE, Kalaburagi
[2] Head of CSE Dept, KBNCE, Kalaburagi

Abstract:

We propose a protocol for reliable mining of association rules in horizontally distributed databases. The current leading protocol is that of Kantarcioglu and Clifton. Our protocol, like theirs, is based on the Fast Distributed Mining (FDM) algorithm of Cheung et al. which is an unsecured distributed version of the Apriori algorithm. The main ingredients in our protocol are two novel secure multi-party algorithms — one that computes the union of private subsets that each of the interacting players hold, and another that tests the inclusion of an element held by one player in a subset held by another. Our protocol offers enhanced privacy with respect to the protocol. In addition, it is simpler and is significantly more efficient in terms of communication rounds, communication cost and computational cost.

Keywords

Association rules, Privacy-preserving Data Mining, Database management, Differential Privacy

1. INTRODUCTION

Data mining can extract important knowledge from large data collections but sometimes these collections are split among various parties. Privacy liability may pre-vent the parties from directly sharing the data, and some types of information about the data. Data mining technology has become prominent as a means of identifying patterns and trends from large quantities of data. Data mining and data warehousing co-jointly: most popular tools operate by gathering all data into a central site then running an algorithm against that data. However, privacy liability can prevent building a centralized warehouse data may be distributed among several custodians none of which are allowed to transfer their data to another site In Horizontally partitioned database there are several players that hold homogeneous database. The goal is to find all association rules with support at least s and confidence at least c, for some given minimal support size s and confidence level c, that hold in the unified database, while minimizing the information disclosed about the private databases held by those players. That goal defines a problem of secure multi-party computation.

If there existed a trusted third party, the players could surrender to him their inputs and he would perform the function evaluation and send to them the resulting output. In the absence of such a trusted third party, it is needed to devise a protocol that the players can run on their own in order to arrive at the required output y. Such a protocol is considered perfectly secure if no player can learn from his view of the protocol more than what he would have learnt in the idealized setting where the computation is carried out by a trusted third party.

In previous year various techniques are applied for secure mining of association rules in horizontally partitioned database. These approaches use various techniques such as data perturbation, homo-morphic encryption, keyword search and oblivious pseudorandom functions etc. These privacy preserving approaches are inefficient due to

- Homo-morphic encryption
- Higher computational cost
- In some of the techniques data owner tries to hide data from data miner.

Our proposed protocol based on two novel secure multiparty algorithms using these algorithms the protocol provides enhanced privacy, security and efficiency as it uses commutative encryption.

In this project we propose a protocol for secure mining of association rules in horizontally distributed database. This protocol is based on: FDM Algorithm which is an unsecured distributed version of the Apriori algorithm. In our protocol two secure multiparty algorithms are involved:

1. Computes the union of private subsets that each interacting players hold.
2. Tests the inclusion of an element held by one player in subset held by another.

In Horizontally partitioned database there are several players that hold homogeneous database. Our protocol offers enhanced privacy with respect to the current leading K and C protocol simplicity, more efficient in terms of communication rounds, communication cost and computational cost. In our problem, the inputs are the partial databases and the required output is the list of association rules that

hold in the unified database with support and confidence no smaller than the given thresholds s and c, respectively. The paper is organized as follows. Section 2 overviews the details about your proposed work. This section includes the details about Algorithms, flowchart etc. Sections 3 presents explain all the result of work in the form of graph, figure, chart etc. Section 4 explains analysis part of the application and feature scope of current work. Section 5 states the possible follow-ups of this work and draws the conclusions.

2. RELATED WORK

Previous work in privacy preserving data mining has considered two related settings. One, in which the data owner and the data miner are two different entities, and another, in which the data is distributed among several parties who aim to jointly perform data mining on the unified corpus of data that they hold. In the first setting, the goal is to protect the data records from the data miner. Hence, the data owner aims at anonymizing the data prior to its release. The main approach in this context is to apply data perturbation [2], [11].

In the second setting, the goal is to perform data mining while protecting the data records of each of the data owners from the other data owners. This is a problem of secure multiparty computation. The usual approach here is cryptographic rather than probabilistic. Lindell and Pinkas showed how to securely build an ID3 decision tree when the training set is distributed horizontally. Lin et al. discussed secure clustering using the EM algorithm over horizontally distributed data. The problem of distributed association rule mining was studied in the vertical setting, where each party holds a different set of attributes, and in the horizontal setting. Also the work of considered this problem in the horizontal setting, but they considered large-scale systems in which, on top of the parties that hold the data records (resources) there are also managers which are computers that assist the resources to decrypt messages; another assumption made in that distinguishes it from and the present study is that no collusions occur between the different network nodes — resources or managers.

The problem of secure multiparty computation of the union of private sets was studied in [7] as well as in. Freedman et al. present a privacy-preserving protocol for set intersections. It may be used to compute also set unions through set complements, since A ∪ B = A ∩ B. Kissner and Song present a method for representing sets as polynomials, and give several privacy-preserving protocols for set operations using these representations. They consider the threshold set union problem, which is closely related to the threshold function (Definition 2.1). The communication overhead of the solutions in those two works, as well as and in our solutions, depends linearly on the size of Fig. 3. Computation and communication costs versus the support threshold s the ground set. However, as the protocols in use homomorphic encryption, while that of uses commutative encryption, their computational costs are significantly higher than ours. The work of Brickell and Shmatikov [7] is an exception, as their solution entails a communication overhead that is logarithmic in the size of the ground set. However, they considered only the case of two players, and the logarithmic communication overhead occurs only when the size of the intersection of the two sets is bounded by a constant. The problem of set inclusion can be seen as a simplified version of the privacy-preserving keyword search. In that problem, the server holds a set of pairs {(xi, pi)}ni =1, where xi are distinct ―keywords‖, and the client holds a single value w. If w is one of the server's keywords, i.e., w = xi for some 1 ≤ i ≤ n, the client should get the corresponding pi.

In case w differs from all xi, the client should get notified of that. The privacy requirements are that the server gets no information about w and that the client gets no information about other pairs in the server's database. This problem was solved by Freedman et. al. If we take all pi to be the empty string, then the only information the client gets is whether or not w is in the set {x1, . . . , xn}. Hence, in that case the privacy-preserving keyword search problem reduces to the set inclusion problem. Another solution for the set inclusion problem was recently proposed.

3. IMPLEMENTATION MODULES:

1. Privacy Preserving Data Mining
2. Distributed Computation
3. Frequent Itemsets
4. Association Rules

MODULES DESCRIPTION:

A. Privacy Preserving Data Mining:

One, in which the data owner and the data miner are two different entities, and another, in which the data is distributed among several parties who aim to jointly perform data mining on the unified corpus of data that they hold. In the first setting, the goal is to protect the data records from the data miner. Hence, the data owner aims at anonym zing the data prior to its release. The main approach in this context is to apply data perturbation. The idea is that. Computation and communication costs versus the number of transactions *N* the perturbed data can be used to infer general trends in the data, without

revealing original record information. In the second setting, the goal is to perform data mining while protecting the data records of each of the data owners from the other data owners. This is a problem of secure multiparty computation. The usual approach here is cryptographic rather than probabilistic.

B. Distributed Computation:

We compared the performance of two secure implementations of the FDM algorithm Section In the first implementation (denoted FDM-KC), we executed the unification step using Protocol UNIFI-KC, where the commutative cipher was 1024-bit RSA in the second implementation (denoted FDM) we used our Protocol UNIFI, where the keyed-hash function was HMAC. In both implementations, we implemented Step 5 of the FDM algorithm in the secure manner that was described in later. We tested the two implementations with respect to three measures:
1) Total computation time of the complete protocols (FDMKC and FDM) over all players. That measure includes the Apriori computation time, and the time to identify the globally *s*-frequent item sets, as described in later.
2) Total computation time of the unification protocols only (UNIFI-KC and UNIFI) over all players. 3) Total message size. We ran three experiment sets, where each set tested the dependence of the above measures on a different parameter: • N — the number of transactions in the unified database,

C. Frequent Itemsets:

We describe here the solution that was proposed by Kantarcioglu and Clifton. They onsidered two possible settings. If the required output includes all globally *s*-frequent item sets, as well as the sizes of their supports, then the values of $\Delta(x)$ can be revealed for all. In such a case, those values may be computed using a secure summation protocol, where the private addend of Pm is $suppm(x) - sNm$. The more interesting setting, however, is the one where the support sizes are not part of the required output. We proceed to discuss it.

D. Association Rules:

Once the set Fs of all *s*-frequent itemsets is found, we may proceed to look for all (s, c)-association rules (rules with support at least sN and confidence at least c). In order to derive from Fs all (s, c)-association rules in an efficient manner we rely upon the straightforward lemma.

4. PERFORMANCE EVALUATIONS

We describe the synthetic database that we used for our experimentation. In Section VI we explain how the database was split horizontally into partial databases. The results are given in Section VI

A. Synthetic database generation:

The databases that we used in our experimental evaluation are synthetic databases that were generated using the same techniques that were introduced in [1] and then used also in subsequent studies such as [8]. Table 1 gives the parameter values that were used in generating the synthetic database. The reader is referred to [8] for a description of the synthetic generation method and the meaning of each of those parameters. The parameter values that we used here are similar to those used in [8]. Parameter Interpretation Value N Number of transactions in the whole database 500,000 L Number of items 1000 At Transaction average size 10 Af Average size of maximal potentially large item sets 4 Nf Number of maximal potentially large itemsets 2000 CS Clustering size 5 PS Pool size 6 Correlation level 0.5 MF Multiplying factor 1800. Parameters for generating the synthetic database.

B. Distributing the database

Given a generated synthetic database D of N transactions and a number of players M, we create an artificial split of D into M partial databases, Dm, $1 \leq m \leq M$, in the following manner: For each $1 \leq m \leq M$ we draw a random number wm from a normal distribution with mean 1 and variance 0.1, where numbers outside the interval [0.1, 1.9] are ignored. Then, we normalize those numbers so that _M m=1 wm = 1. Finally, we randomly split D into m partial databases of expected sizes of wmN, $1 \leq m \leq M$, as follows: Each transaction $t \in D$ is assigned at random to one of the partial databases, so that $Pr(t \in Dm) = wm$, $1 \leq m \leq M$.

C. Experimental setup

We compared the performance of two secure implementations of the FDM algorithm (Section 1.1.2). In the first implementation (denoted FDM-KC), we executed the unification step (Step 4 in FDM) using Protocol UNIFI-KC, where the commutative cipher was 1024-bit RSA [25]; in the second implementation (denoted FDM) we used our Protocol UNIFI, where the keyed-hash function was HMAC [4]. In both implementations, we implemented Step 5 of the FDM algorithm in the secure manner that was described in Section 3. We

tested the two implementations with respect to three measures:
1) Total computation time of the complete protocols (FDMKC and FDM) over all players. That measure includes the Apriori computation time, and the time to identify the globally s-frequent itemsets, as described in Section 3. (The latter two procedures are implemented in the same way in both Protocols FDM-KC and FDM.)
2) Total computation time of the unification protocols only (UNIFI-KC and UNIFI) over all players.
3) Total message size. We ran three experiment sets, where each set tested the dependence of the above measures on a different parameter: N — the number of transactions in the unified database.

5. CONCLUSION

We proposed a protocol for secure mining of association rules in horizontally distributed databases that improves significantly upon the current leading protocol in terms of privacy and efficiency. One of the main ingredients in our proposed protocol is a novel secure multi-party protocol for computing the union (or intersection) of private subsets that each of the interacting players hold. Another ingredient is a protocol that tests the inclusion of an element held by one player in a subset held by another. Those protocols exploit the fact that the underlying problem is of interest only when the number of players is greater than two.

6. REFERENCES

s1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *SIGMOD Conference*, pages 439–450, 2000.

[3] D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *STOC*, pages 503–513, 1990.

[4] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In *Crypto*, pages 1–15, 1996.

[5] A. Ben-David, N. Nisan, and B. Pinkas. FairplayMP - A system for secure multi-party computation. In *CCS*, pages 257–266, 2008.

[6] J.C. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret secret. In *Crypto*, pages 251–260, 1986.

[7] J. Brickell and V. Shmatikov. Privacy-preserving graph algorithms in the semi-honest model. In *ASIACRYPT*, pages 236–252, 2005.

[8] D.W.L. Cheung, J. Han, V.T.Y. Ng, A.W.C. Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In *PDIS*, pages 31– 42, 1996.

[9] D.W.L Cheung, V.T.Y. Ng, A.W.C. Fu, and Y. Fu. Efficient mining of association rules in distributed databases. *IEEE Trans. Knowl. Data Eng.*, 8(6):911–922, 1996.

[10] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[11] A.V. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *KDD*, pages 217–228, 2002.

[12] R. Fagin, M. Naor, and P. Winkler. Comparing Information without Leaking It. *Communications of the ACM*, 39:77–85, 1996.