Mining Distributed Private Databases Using Random Response Protocols

Li Xiong and Pawel Jurczyk
Emory University
{lxiong, pjurczy}@mathcs.emory.edu

Ling Liu
Georgia Institute of Technology
lingliu@cc.gatech.edu

Abstract

There is a growing demand for sharing data repositories that often contain personal information across multiple autonomous, possibly untrusted, and private databases. This paper discusses constraints imposed by individual privacy as well as institutional data confidentiality on data mining across multiple databases and presents our initial solutions. We develop a suite of decentralized protocols that aim to effectively anonymize the data for each individual database and compute the query results across databases in a probabilistically secure manner. By relaxing the privacy constraints and accuracy requirement, the protocols achieve efficiency and scalability not offered by traditional multi-party secure computation approaches. Our primary viewpoint is that some approximation is tolerable and even desirable for scalable and robust mining across large, multi-party distributed environment.

1 Introduction

The information age has enabled many organizations to collect large amounts of data that often contains personal information. There is a growing demand for sharing such data repositories across multiple autonomous, possibly untrusted, and private databases. An example scenario is the Shared Pathology Informatics Network (SPIN))[1] initiative by National Cancer Institute for researchers throughout the country to share pathology-based data sets. However, personal health information is protected under the Health Insurance Portability and Accountability Act (HIPAA)[2][3]. and cannot be revealed without de-identification or anonymization. In addition, institutions may not want to reveal their databases even after de-identification for various legal or commercial reasons.

[1] Shared Pathology Informatics Network. http://www.cancerdiagnosis.nci.nih.gov/spin/

[2] Health Insurance Portability and Accountability Act (HIPAA). http://www.hhs.gov/ocr/hipaa/.

[3] State law or institutional policy may differ from the HIPAA standard and should be considered as well.

We consider these constraints imposed by individual privacy as well as institutional data confidentiality on data mining across multiple distributed databases. The first constraint can be generalized into the problem of privacy preserving data publishing where a data custodian needs to distribute an anonymized view of the data that does not contain individually identifiable information to a data recipient (either a shared network or an individual researcher or institution). The second constraint can be generalized into the problem of multi-party secure computation where we wish to compute an answer given a query or data analysis task spanning multiple databases without revealing any information of each individual database apart from the result. In a distributed environment, if we can guarantee the data confidentiality imposed by the second constraint, the individual privacy imposed by the first constraint is also guaranteed as long as the mining result alone does not reveal any personal information.

We identify three important dimensions that we should consider when designing a privacy preserving distributed mining algorithm, namely, *accuracy, efficiency, and privacy*. Thinking of the design space in terms of these three dimensions presents many advantages. Ideally, we would like the algorithm to have a comparable accuracy to its non-privacy preserving counterpart, and an absolute privacy wherein no information other than the trained model or mined results should be revealed to any node. At one end of the spectrum, we have the non-privacy preserving classifier algorithms, which are highly efficient but are not secure. At the other end, we have the secure multi-party computation protocols [7, 6], using which we can construct classifiers which are provably secure in the sense that they reveal the least amount of information and have the highest accuracy; but are very inefficient. Our design goal is to look for algorithms that can provide a desired level of tradeoff between the accuracy of the classifier constructed and the stringency of the privacy requirements while maintaining efficiency.

With these design objectives in mind, we present a set of decentralized protocols that effectively anonymize the data for each individual database and compute the mining results across databases in a probabilistically secure man-

ner. Rather than relying on cryptographic techniques, it is built on top of the random response idea and utilizes a set of probabilistic multi-round protocols. By relaxing the privacy and confidentiality constraints and accuracy requirement, the protocols achieve efficiency and scalability not offered by traditional multi-party secure computation approaches. The primary contribution of the paper does not lie in each of the protocols themselves, but rather in illustrating that we can build primitive as well as complex protocols from multi-round random response protocols without relying on encryption techniques and some approximation in the protocols is tolerable and even desirable for scalable and robust mining across large, multi-party distributed environment.

2 Related Work

The approach of protecting privacy of distributed sources was first addressed by the construction of decision trees [13]. This work closely followed the traditional secure multiparty computation approach and achieved perfect privacy. There has since been work to address association rules [18, 8], naive Bayes classification [10, 20, 27], and k-means clustering [19] as well as general tools for privacy preserving data mining [5]. As a recent effort, there is also research on privacy preserving top$k$ queries [21] and privacy preserving distributed k-NN classifier [9], both across vertically partitioned data using k-anonymity privacy model. A few specialized protocols have been proposed, typically in a two party setting, e.g., for finding intersections [2], and kth ranked element [1]. [23] studied the problem of integrating private data sources with vertically partitioned data while satisfying k-anonymity of the data. Though still based on cryptographic primitives, they achieve better efficiency than traditional multi-party secure computation methods by allowing minimal information disclosure. Another main approach to achieve privacy preserving data mining is to use data perturbation techniques, either additive or multiplicative randomization [11, 22, 14]. There are also work [12] focused on coping with potential malicious behaviors of participating parties, instead of the traditionally assumed semi-honest behavior.

In contrast, our protocol does not rely on cryptographic primitives or data perturbation. It leverages the large multi-party network ($n > 3$) and utilizes probabilistic multi-round distributed protocols to achieve minimal information disclosure and minimal overhead. We illustrate the idea by building a kNN classifier across horizontally partitioned data.

Another related area is the anonymous network where the requirement is that the identity of a user be masked from an adversary. There have been a number of application specific protocols proposed for anonymous communication, including anonymous messaging (Onion Routing [17]), anonymous web transactions (Crowds [16]), anonymous indexing (Privacy Preserving Indexes [3]) and anonymous peer-to-peer systems (Mutual anonymity protocol [24]). Some of these techniques may be applicable for data integration tasks where parties opt to share their information anonymously. However, anonymity is a less strong requirement than data privacy.

3 Primitive Protocols

Consider a large multi-party network ($n > 3$) where data are *horizontally partitioned* across the private databases, we first present a set of protocols that provide primitive operations common to and required by many data mining applications. The key idea of the protocols is to leverage the inherent anonymity in the large network and utilize probabilistic multi-round distributed protocols to achieve minimal information disclosure and minimal overhead rather than relying on computation-heavy encryption techniques. The primitive protocols are not exhaustive but sufficient for us to illustrate the mining examples in a later section.

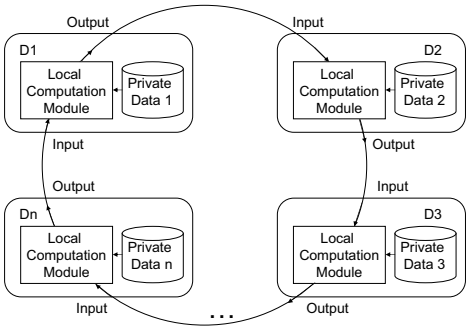

Figure 1. Protocol Overview

Protocol Structure. Figure 1 presents a system overview. Nodes are mapped into a *ring topology* randomly. Each node has a predecessor and successor. It is important to have the random mapping to reduce the cases where two colluding adversaries are the predecessor and successor of an innocent node. The ring setting is commonly used by distributed consensus protocols such as leader election algorithm [15]. We also plan to explore other topologies such as hierarchy for designing potentially more efficient protocols. The *initialization module* is designed to select the starting node among the n participating nodes and then initialize a set of parameters used in the local computation algorithms. The *local computation module* is a standalone component that each node executes independently illustrated in Figure

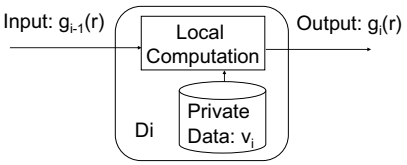

Figure 2. Protocol Local Algorithm

2. The essential idea of the protocol is to perform one or multiple rounds in which a global value is passed from node to node along the ring. Each node can inject some randomization into the local computation, such that the chance of data value disclosure at each node is minimized and at the same time the eventual result of the protocol is guaranteed to be correct.

We assume a semi-honest model for the participating nodes in the sense that they correctly follow the protocol specification, yet attempt to learn additional information about other nodes by analyzing the transcript of messages received during the execution of the protocol. One of our ongoing efforts is to develop a decentralized kNN classification protocol that is resilient against malicious nodes.

Sum. Distributed data mining algorithms frequently calculate the sum of values from individual sites. We first present a simple PrivateSum protocol for multiple nodes ($n >= 3$) similar to [5]. Note that it is a deterministic protocol where only the first node uses a random value. Assume that the domain for the values lies in the range $[0, n]$ and each node i holds a private value v_i. Node 1 generates a random value g_1 between $[0, n]$ and passes it to its successor. Since the value is chosen uniformly from the range, node 2 learns nothing about the actual value of v_1. Node i, upon receiving the global value g_{i-1} from its predecessor, compute the sum $g_{i-1} + v_i$ and sends the value to its successor. Node $i + 1$ does not learn anything about the values held by previous nodes. At the end of the round, node 1 computes the global sum by $g_n - g_1 + v_1$.

Union. Set union is another common operation in data mining (such as finding frequent itemsets over the union of databases). A commutative encryption based approach is suggested in [5]. We present a protocol similar to PrivateSum protocol that does not rely on encryption. Assume each node i hold a private set S_i. Node 1 generates a random set G_1 and passes it to its successor. Node 2 learns nothing about the actual set of node 1. Node i, upon receiving the global set G_{i-1} from its predecessor, compute the union $G_{i-1} \cup S_i$ and sends the value to its successor. With a large random set to start with, node $i+1$ learns little about the set held by previous nodes. At the end of the round, node 1 computes the global union by $(G_n - G_1) \cup S_1$. Note that this protocol does not remove duplicates. Alternatively, we can have each node create a binary vector where 1 in the ith entry represents that the node has the ith item. We can use a probabilistic OR protocol [4] to compute the OR of the bit vectors and derive the set union.

Max. The PrivateMax protocol is proposed in [25] for max(min) selection for multiple nodes ($n >= 3$). It is a probabilistic protocol where each node injects certain randomization in their local computation with a given randomization probability associated with each round. The randomization probability decreases in each round to ensure that the final result will be produced in a bounded number of rounds. Given an initial an initial probability, p_0, and a dampening factor, d, the randomization probability for round r, $P_r(r)$, can be defined as $P_r(r) = p_0 * d^{r-1}$. At round r, node i performs a local randomized algorithm described in Algorithm 1. For detailed illustration and analysis of the algorithm, please refer to [25].

Algorithm 1 Local Algorithm for PrivateMax Protocol (executed by node i at round r)

INPUT: $g_{i-1}(r)$, v_i, OUTPUT: $g_i(r)$
$P_r(r) \leftarrow p_0 * d^{r-1}$
if $g_{i-1}(r) \geq v_i$ **then**
  $g_i(r) \leftarrow g_{i-1}(r)$
else
  with probability P_r: $g_i(r) \leftarrow$ a random value between $[g_{i-1}(r), v_i)$
  with probability $1 - P_r$: $g_i(r) \leftarrow v_i$
end if

Top$k$. The PrivateTop$k$ protocol finds the top$k$ values and works similarly as PrivateMax ($k = 1$) protocol. Each node uses a local vector to participate in the protocol. The protocol performs multiple rounds in which a current global top$k$ vector is passed from node to node along the ring. Each node i, upon receiving the global vector from its predecessor at round r, performs a randomized algorithm and passes its output to its successor node. The complexity of extending the protocol from max to top$k$ lies in the design of the randomized algorithm. For detailed description and analysis of the protocol, please refer to [25].

4 Data Mining Protocols

In this section, we illustrate how we can build aggregate protocols based on previous protocols for mining in a distributed and privacy preserving manner.

kNN Classification. We first consider the problem where the nodes want to train a kNN classifier on the union of their databases while revealing as little information as possible to the other nodes during the construction of the classifier (*training* phase) and the classification of a new query (*test* phase) and present a distributed kNN protocol [26].

To solve the kNN classification problem, we need to adapt the basic distance weighted kNN classification algorithm to work in a distributed setting in a privacy preserving manner. We divide the kNN classification problem into the following two sub-problems.

1. **Nearest neighbor selection:** Given a query instance x to be classified, the databases need to identify all points that are among the k nearest neighbors of x in a privacy preserving manner.
2. **Classification:** Each node calculates its local classification of x and then cooperate to determine the global classification of x in a privacy preserving manner.

Algorithm 2 kNN Classification

Input: x, an instance to be classified
Output: $classification(x)$, classification of x

- Each node computes the distance between x and each point y in its database, $d(x, y)$, selects k smallest distances (locally), and stores them in a local distance vector ldv.
- Using ldv as input, the nodes use the PrivateTop$k$ protocol to select k nearest distances (globally), and stores them in gdv.
- Each node selects the kth nearest distance Δ: $\Delta = gdv(k)$.
- Assuming there are v classes, each node calculates a local classification vector lcv for all points y in its database: $\forall 1 \leq i \leq v, lcv(i) = \sum_y w(d(x,y)) * [f(y) == i] * [d(x,y) \leq \Delta]$, where $d(x, y)$ is the distance between x and y, $f(y)$ is the classification of point y, and $[p]$ is a function that evaluates to 1 if the predicate p is true, and 0 otherwise.
- Using lcv as input, the nodes use the PrivateSum protocol to calculate the global classification vector gcv.
- Each node assigns the classification of x as $classification(x) \leftarrow \arg\max_{i \in V} gcv(i)$.

In order to determine the points in their database that are among the k nearest neighbors of x, each node calculates k smallest distances between x and the points in their database (locally) and then we can use the PrivateTop$k$ protocol to determine k smallest distances between x and the points in the union of the databases. We can assume that the distance is a one-way function so that nodes do not know the exact position of each other node by distance. There has been privacy preserving algorithms recently proposed [1] for finding kth element that we can use for implementing this step. Although information-theoretically secure, it is still computationally expensive.

After each node determines the points in its database which are within the kth nearest distance from x, each node computes a local classification vector of the query instance where the ith element is the amount of vote the ith class received from the points in this node's database which are among the k nearest neighbors of x. The nodes then participate in a privacy preserving term-wise addition of these local classification vectors using the PrivateSum protocol to determine the global classification vector. Once each node knows the global classification vector, it can find the class with the global majority of the vote by determining the index of the maximum value in the global classification vector.

Putting things together, Algorithm 2 shows a sketch of the complete protocol, Private$k$NN, that builds a kNN classifier across multiple private databases. We have conducted an initial set of experimental evaluation of this protocol in terms of its correctness, efficiency, and privacy characteristics and interested readers can refer to [26] for the detailed results.

Algorithm 3 k-Means Clustering

Input: k, the number of clusters
Output: the cluster centers

- Nodes agree on an initial random set of cluster centers $c_i, i = 1..k$
- Each node computes the distance between each point x in its database and each cluster center c_i, and assigns each point to its closest cluster center.
- For each cluster center, the nodes use the PrivateSum protocol to compute the new cluster center.
- The nodes again use the PrivateSum protocol to compute the total distance between each point and its cluster center. The algorithm repeats until the distance is within a specified value.

k-Means Clustering. k-means clustering is a simple technique to group data points into k clusters. Each data point is placed in its closest cluster given an initial set of cluster centers, and the cluster centers are then adjusted based on

the data placement. This repeats until the positions stabilize. Algorithm 3 shows a sketch of the distributed protocol that performs k-means clustering across multiple private databases. At the end of the protocol, each node knows the cluster center but nothing else and they can assign each of their local points to the appropriate cluster.

5 Discussion

In this paper, we presented a set of distributed protocols for primitive operations and using those protocols to construct a k-Nearest Neighbor classifier and a k-means clustering algorithm across horizontally partitioned private databases. It is a proof-of-concept for building simple and complex mining protocols utilizing probabilistic multi-round protocols by leveraging the large distributed network that provides inherent anonymity.

Our work continues on several directions. First, we are thoroughly analyzing the efficiency and privacy properties of the algorithms under various circumstances such as repeated classifications and presence of malicious behaviors. Second, we are exploring different topologies and other performance optimization techniques for achieving further scalability in large distributed environment. Finally, we are also interested in investigating the possibility of building adaptive protocols based on different privacy requirements and natures of the mining tasks.

Acknowledgements

This research is partially supported by Emory URC grant and ITSC grant from the first author, and grants from NSF CSR, NSF CyberTrust, NSF ITR, AFOSR, IBM Faculty Award, and an IBM SUR grant from the last author.

References

[1] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the kth ranked element. In *IACR Conference on Eurocrypt*, 2004.

[2] R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *ACM SIGMOD Conference*, 2003.

[3] M. Bawa, R. J. Bayardo, and R. Agrawal. Privacy-preserving indexing of documents on the network. In *29th International Conference on very large databases (VLDB)*, 2003.

[4] M. Bawa, R. Jr, and R. Agrawal. Privacy-preserving indexing of documents on the network. In *VLDB*, 2003.

[5] C. Clifton, M. Kantarcioglu, X. Lin, J. Vaidya, and M. Zhu. Tools for privacy preserving distributed data mining. In *SIGKDD Explorations*, 2003.

[6] O. Goldreich. Secure multi-party computation, 2001. Working Draft, Version 1.3.

[7] S. Goldwasser. Multi-party computations: past and present. In *ACM Symposium on Principles of Distributed Computing (PODC)*, 1997.

[8] M. Kantarcioglu and C. Clifton. Privacy preserving data mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(9), 2004.

[9] M. Kantarcioglu and C. Clifton. Privacy preserving k-nn classifier. In *ICDE*, 2005.

[10] Murat Kantarcoglu and J. Vaidya. Privacy preserving naive bayes classifier for horizontally partitioned data. In *IEEE ICDM Workshop on Privacy Preserving Data Mining*, 2003.

[11] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the IEEE International Conference on Data Mining*, page 99, Melbourne, FL, November 2003.

[12] Hillol Kargupta, Kamalika Das, and Kun Liu. Multi-party, privacy-preserving distributed data mining using a game theoretic framework. In *PKDD*, pages 523–531, 2007.

[13] Y. Lindell and B. Pinkas. Privacy preserving data mining. *Journal of Cryptology*, 15(3), 2002.

[14] Kun Liu, Hillol Kargupta, and Jessica Ryan. Random Projection-Based Multiplicative Data Perturbation for Privacy Preserving Distributed Data Mining. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 18(1):92–106, January 2006.

[15] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.

[16] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1), 1998.

[17] S. Syverson, D. M. Coldsehlag, and M. C. Reed. Anonymous connections and onion routing. In *IEEE Symposium on Security and Privacy*, 1997.

[18] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *SIGKDD*, 2002.

[19] J. vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *SIGKDD*, 2003.

[20] Jaideep Vaidya and Chris Clifton. Privacy preserving naive bayes classifier for vertically partitioned data. In *SIGKDD*, 2003.

[21] Jaideep Vaidya and Chris Clifton. Privacy-preserving top-k queries. In *ICDE*, 2005.

[22] 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*, 33(1), 2004.

[23] K. Wang, B. C. M. Fung, and G. Dong. Integrating private databases for data analysis. In *IEEE ISI*, 2005.

[24] L. Xiao, Z. Xu, and X. Zhang. Mutual anonymity protocols for hybrid peer-to-peer systems. In *IEEE International Conference on Distributed Systems (ICDCS)*, 2003.

[25] L. Xiong, S. Chitti, and L. Liu. Topk queries across multiple private databases. In *25th International Conference on Distributed Computing Systems (ICDCS)*, 2005.

[26] Li Xiong, Subramanyam Chitti, and Ling Liu. Mining multiple private databases using a knn classifier. In *SAC*, pages 435–440, 2007.

[27] Z. Yang, S. Zhong, and R. N. Wright. Privacy-preserving classification of customer data without loss of accuracy. In *SIAM SDM*, 2005.