A Secure Collaborative Machine Learning Framework Based on Data Locality

Kaihe Xu*, Haichuan Ding*, Linke Guo†, Yuguang Fang*
*Department of Electrical and Computer Engineering,
University of Florida, Gainesville, FL 32611, USA
{xukaihe, haichuanding}@ufl.edu, fang@ece.ufl.edu
‡Department of Electrical and Computer Engineering,
Binghamton University, Binghamton, NY 13902, USA
lguo@binghamton.edu

***Abstract*—Advancements in big data analysis offer cost-effective opportunities to improve decision-making in numerous areas such as health care, economic productivity, crime, and resource management. Nowadays, data holders are tending to sharing their data for better outcomes from their aggregated data. However, the current tools and technologies developed to manage big data are often not designed to incorporate adequate security or privacy measures during data sharing. In this paper, we consider a scenario where multiple data holders intend to find predictive models from their joint data without revealing their own data to each other. Data locality property is used as an alternative to multi-party computation (SMC) techniques. Specifically, we distribute the centralized learning task to each data holder as local learning tasks in a way that local learning is only related to local data. Along with that, we propose an efficient and secure protocol to reassemble local results to get the final result. Correctness of our scheme is proved theoretically and numerically. Security analysis is conducted from the aspect of information theory.**

I. Introduction

The amount of data in the world is exploding, but with advancements in big data technology, we know how to perform cost-effective analysis on these data to improve decision-making in numerous areas such as health care, economic productivity, crime, and resource management. However, existing tools and technologies developed to manage big data are often not designed to incorporate adequate security or privacy measures during data sharing. Currently, data holders are tending to sharing their data in order to get better outcomes from the aggregated data. It could be several banks wishing to conduct credit risk analysis based on their transaction records, or medical institutions trying to discover certain correlations between symptoms and diagnoses from patients' records, or E-commerce websites trying to build a recommendation engine based on their users' profiles. Usually, collaborative learning is conducted in one of the following fashions: horizontally, when each site holds some observations and they want to train a model with their joint observations; or vertically, when each site holds some feature attributes and they want to train a model spanning over their joint feature space. In either case, security is always an important concern because the shared data could be sensitive when referring to users' privacy and trade secrets.

This work was partially supported by the US National Science Foundation under grant CNS-1423165.

Over the last decades, numerous collaborative data mining methods have been proposed to deal with privacy issues. For example, in [14], Yuan and Yu use homomorphic encryption evaluate activation function during back-propagation training; in [15], Justin and Stan propose to use secure dot product protocol to achieve secure support vector machine learning; in [9], Lindell and Pinkas propose a secure protocol to compute the result of $(v_1 + v_2)\log(v_1 + v_2)$ without revealing any unnecessary information; in [8] Kantarcioglu and Clifton use commutative encryption and secure sum protocols to find the association rules over the horizontally partitioned data and in [11], Vaidya and Clifton apply secure dot protocols to find association rules over the vertically partitioned data.

However, the existing algorithms are based on a centralized model for small scale data. When facing with big data, existing schemes may not work because 1. they are not cost-effective; 2. they cannot fit to the existing big data processing framework such as hadoop. In this paper, we propose a scheme for data holders to do collaborative machine learning over their joint data with the guarantee that during this process, only the final results are revealed but nothing else. Our scheme is inspired by the data locality property: to move computation as close to the data source as possible in avoidance of data transmission. Specifically, we distribute the centralized learning task to each data holder as local learning tasks such that local learning is only related to local data. In other words, instead of sharing the raw data, we only allow data holders to share their local training results. Along with that, we propose an efficient and secure protocol to reassemble local results to get the final result. Similar idea has been applied to SVM learning in [13]. Compared with that paper, the proposed scheme is compatible with large numbers of machine learning algorithms and only a minor modification is needed to adapt to the existing implementations.

The rest of this paper is organized as follows: In Section II, we present our problem settings. In Section III we present

our system models for the horizontally partitioned case and the vertically partitioned case, respectively. In Section IV, security analysis is conducted. In Section V, the performance of our scheme is tested against two popular data sets and finally, in Section VI, we conclude this paper.

II. Problem Formulation

In this work, we assume that there are M learners trying to train a predictive model $h_\theta : \mathbb{R}^{p+1} \rightarrow \{0, 1\}$ with parameter θ over their joint training data by performing empirical risk minimization [12] as:

$$h_\theta = \arg\min_{h_\theta} \frac{1}{N} \sum_{i}^{N} \mathcal{I}\{h_\theta(x_i) \neq y_i\}. \tag{1}$$

In (1), N is the number of joint training records, y_i is the output of training record x_i. $\mathcal{I}\{h_\theta(x_i) \neq y_i\}$ is equals to 1 if $h_\theta(x_i) \neq y_i$ and 0 otherwise. The above problem is known as $\mathcal{NP}$ hard and heuristic solutions could be obtained by substituting the risk with a convex cost function J. For example, if linear regression is used, $J = \sum_m^M (h_\theta(x_m) - y_m)^2$ and $h_\theta(x_i) = \theta^T x_i$; if logistic regression is used as the model, $J = \sum_m^N -y_m \log(h_\theta(x_m)) - (1 - y_m) \log(1 - h_\theta(x_m))$ and $h_\theta(x_m) = \frac{1}{1+e^{-\theta^T x_m}}$ [4]; if SVM is used, $J = \frac{1}{2}\|\theta\|^2 + \sum_m^N \max(1 - y_m h_\theta(x_m), 0)^2$ and $h_\theta(x_m) = \theta^T x_m$ [16]; if multi-layer perceptions is used, $J = \sum_m^N (o_m - y_m)^2$, where o_i denotes neural network output of x_m and h_θ denotes the weight parameters for all the layers [3]. It could be shown that optimizing over the functional h_θ is equivalent to optimizing over its parameter θ. In the rest of this paper, we use θ and the predictive model interchangeably to denote the machine learning result. Hence the goal for M learners is to agree with θ, which is able to minimize the centralized cost function over their joint data.

However, the training data X of size $N \times P$ (N records, each records are with P features) and Y of size $N \times 1$ are distributed among the M learners as their private data. Generally speaking, data is usually distributed among multiple learners in two fashions. In a horizontally partitioned scenario, the joint data X and Y are partitioned by rows, each learner has N_m number of records, but their records are of the same length (same number of features). Namely, the private training data X_m is of size $N_m \times P$ denotes that learner m has N_m training records and the corresponding labels Y_m of size $N_m \times 1$. In a vertically partitioned scenario, X is partitioned by columns, each leaner has the same number of records, but the records are with different features. Specifically, X_m is of size $N \times P_m$ meaning that learner m is holding P_m features of the joint data. In this case, Y of size $N \times 1$ should agree with the M learners. In either case, our goal is to find a collaborative training scheme with the following two properties:

I Training result of the collaborative training scheme should be identical to the centralized training result obtained by (1).

II During the collaborative training process, only the final training result is revealed but nothing else.

Intuitively, learners need to exchange some information such that the joint learning result has considered all the N training records with P features. The first property assert that our scheme guaranteed that sufficient information is exchanged as if the learners are directly sharing their private data. While the second property asserts that the private training sets are secure during information exchange. In this paper, we assume that the M learners are semi-honest: they will follow the collaborative training scheme, but they are also curious about others' private training data. They will collect exchanged information and try to trace back others' private training data.

III. System Model

In this section, we introduce the proposed collaborative learning scheme under horizontally and vertically partitioned scenario, respectively. The idea is inspired by data locality property of Hadoop MapReduce framework. Firstly, each learner only trains the model with its own private data to get the local training result. These results are very likely to be local optimal and may different from each other. After local training results are obtained, the M learners will average their local results to find a global knowledge. In the next iteration, this global result is feeded back to each learner to guide its next local training. As iterations goes on, it could be shown that the local training results are converging to a same value, which is identical to the centralized learning result over the joint data. Notice that during this collaborative training process, only the global training results are revealed, the goal of our security analysis section is to show that an semi-honest adversary cannot trace back others' private data from these revealed information.

A. Horizontally Partitioned

For a horizontally partitioned case, each of the distributed learners will establish a full-featured predictive model, because each share of the training data is full-featured. Hence, there will be M copies of the model as θ_i, generated by M learners, respectively. $\{\theta_i\}$ are of the same length and should converge to the same value. Consider the following local training problems:

$$\begin{aligned} \min_{\theta_i} \quad & J_i(\theta_i) \\ \text{s.t.} \quad & \theta_i = z. \end{aligned} \tag{2}$$

If the constraint of $\theta_i = z$ is ignored, then θ_i is the optimal model to fit X_i. If the equivalent constraint is strictly enforced, all $\{\theta_i\}$ will be equals to z and hence equals to each other. In other words, while optimizing the local cost function J_i, $\{\theta_k | k \neq m\}$ are considered and thus $\{X_k | k \neq m\}$ are indirectly considered. θ_i is then the global optimal solution over the joint training data. In the rest of this subsection, we first show how to solve problem (2) and then prove that our solution is identical to the centralized optimal solution over the joint data.

Generally speaking, $\theta_i = z$ is hard to be enforced, because z itself is also a variable. Instead of trying to solve the local problems with $\theta_i = z$ enforced, we use a regularization term of

$\frac{\rho}{2}\|\theta_i - z\|^2$ to get rid of this constraint and form an augmented Lagrange function in the following form:

$$\mathcal{L} = \sum_i^M J_i(\theta_i) + \sum_i^M \frac{\rho}{2}\|\theta_i - z + r_i\|^2. \tag{3}$$

It is named "augmented Lagrange function" because (3) could be viewed as the Lagrange of (2) added by a ℓ^2 norm of $\theta_i - z$. An alternating method could be used to find θ_i and z iteratively [5]. The idea is to optimize θ_i while keeping z fixed and then use the latest θ_i to find z, and so forth. Notice that θ_i and $\{\theta_k | k \neq m\}$ are independent, hence we could optimize over them in parallel. r_i is the accumulated residue of $\theta_i - z$. The iterative update are listed as follow:

$$\theta_i^{t+1} = \arg\min_{\theta_i} \; J_i(\theta_i) + \frac{\rho}{2}\|\theta_i - z^t + r_i^t\|^2, \tag{4}$$

$$z^{t+1} = \frac{1}{M}\sum_i^M \theta_i^{t+1} + r_i^t, \tag{5}$$

$$r_i^{t+1} = r_i^t + \theta_i^{t+1} - z^{t+1}. \tag{6}$$

Notice that if we use gradient descent algorithm to calculate θ_i^{t+1} in (4), it could be written as:

$$\begin{aligned} \theta_i &\leftarrow \theta_i - \alpha(\frac{\partial L}{\partial \theta_i}) \\ &= \theta_i - \alpha[\frac{\partial J_i}{\partial \theta_i} + \rho(\theta_i - z^t + r_i^t)] \\ &= \theta_i - \alpha\nabla J_i - \alpha\nabla_2, \end{aligned} \tag{7}$$

where, $\nabla_2 = \rho(\theta_i - z^t + r_i^t)$ denotes the gradient for $\{\theta_i\}$ to achieve consensus. A nice property of our design is also illustrated in (7), with which we only need to do a minor modification to the existing implementation of various models. For example, in [7], a coordinate descent method is used to compute L1-SVM and L2-SVM and a trust region Newton method is used for large scale logistic regression. Our implementation could be easily fitted into the existing ones by plugging the gradient calculated in (7). Take neural network as another example, ∇J_i could be calculated by the classic back propagation algorithm and ∇_2 could be added to achieve consensus.

As a summary, the proposed scheme is able to achieve consensus by considering each learner's private training data since, during gradient descent update, ∇J_i makes θ_i to fit the local model better and ∇_2 makes θ_i converge to the global knowledge z. We use the following three theorems to prove the correctness.

Theorem 3.1: As t grows, $\{\theta_i\}$ and z are converging, i.e. $\theta_i^{t+1} = \theta_i^t$ and $z^{t+1} = z^t$ for a sufficient large t.

For a general machine learning problem, J is a closed and proper convex function, which is the sufficient condition to guarantee the convergence of the results [5].

Theorem 3.2: If θ_i^* is the final stable local result for learner i, then $\theta_i^* = \theta_k^*, \forall m, k$.

$\theta_i^{t+1} = \theta_i^t$ implies that problem (4) is giving the same solution for each learner. Knowing that $z^{t+1} = z^t$ and the fact that problem (4) is convex, then $r_i^{t+1} = r_i^t$. Hence, from (6), we know that $\theta_i^t = z^t$ and thus $\theta_i^* = \theta_k^*, \forall m, k$.

Theorem 3.3: Denote θ^* as the centralized training result over joint data, then $\theta_i^* = \theta^*$.

Let $J(\theta)$ denote the cost function over the joint training data, then $\theta^* = \arg\min_\theta J(\theta)$ and also $J(\theta) = \sum_i^M J_i(\theta)$ (cost over the joint data equals to summation of cost over each piece). The local optimum θ_i provides that

$$\begin{aligned} &\frac{\partial J_i(\theta_i)}{\partial \theta_i} + \rho(\theta_i - z^t + r_i^t)\Big|_{\theta_i^*} = 0 \\ \Rightarrow &\sum_i^M \left[\frac{\partial J_i(\theta_i)}{\partial \theta_i} + \rho(\theta_i - z^t + r_i^t)\right]\Big|_{\theta_i^*} = 0 \\ \Rightarrow &\sum_i^M \frac{\partial J_i(\theta_i)}{\partial \theta_i}\Big|_{\theta_i^*} + \rho\left[\sum_i^M(\theta_i^* + r_i^t) - Mz^t\right] = 0 \\ \Rightarrow &\frac{\partial J(\theta)}{\partial \theta}\Big|_{\theta_i^*} = 0. \end{aligned} \tag{8}$$

The above equality holds because $\frac{1}{M}\sum_i^M \theta_i^* = z^t$ and $\sum_i^M r_i^t = 0$.

B. Vertically Partitioned

In this subsection, we will talk about the case where the training data is vertically distributed among M learners. Notice that in this case, each learner has a matrix X_i of size $N \times P_i$ denotes that learner m has N records with P_i features. For this case, each learner only holds one part of the model θ as θ_i and they need to put their models together to find the final global model. Denote the centralized cost function J as a summation of two parts as $J = \sum_i^M f_i(\theta_i) + g(\sum_i^M X_i^T\theta_i)$, where $\sum_i^M f_i(\theta_i)$ could be the regularization terms of θ and $g(\sum_i^M X_i^T\theta_i)$ indicates that the distributed data $\{X_i\}$ should be put together to define the model θ. Then, we have the following optimization problem:

$$\begin{aligned} \min_{\{\theta_i\},\{z_i\}} \; &\sum_i^M f_i(\theta_i) + g(\sum_i^M z_i) \\ \text{s.t.} \;\; &z_i = X_i^T\theta_i, \quad \forall m = 1, 2, ...M \end{aligned} \tag{9}$$

The dual function of (9) $\Gamma(\lambda_1, \lambda_2, ...\lambda_i) : \mathbb{R}^N \times \mathbb{R}^N ... \times \mathbb{R}^N \rightarrow \mathbb{R}$ could be written as equation (10) [6].

$$\begin{aligned} \Gamma &= \inf_{\{\theta_i\},\{z_i\}} \left\{\sum_i^M f_i(\theta_i) + g(\sum_i^M z_i) - \sum_i^M \lambda_i(X_i^T\theta_i - z_i)\right\} \\ &= \sum_i^M \inf_{\theta_i}\{f_i(\theta_i) - \langle -\lambda_i X_i, \theta_i\rangle\} - \inf_z\{g(z) - \langle\lambda_i, z\rangle\} \\ &= \sum_i^M -f_i^\star(-\lambda_i X_i) - g^\star(\lambda) \end{aligned} \tag{10}$$

Notice that $f_i^\star$ and $g^\star$ denote the convex conjugate for f_i and g, respectively. For simplicity, z is used to substitute $\sum_i^M z_i$. The above derivation is true only when $\lambda_i = \lambda, \forall i = 1, 2, ...M$,

otherwise, $\Gamma = -\infty$. Denote $\psi_i(\lambda_i) = -f_i^\star(-\lambda_i X_i) - \frac{1}{M} g^\star(\lambda_i)$, problem (9) could be reformulated as

$$\begin{aligned} \min_{\lambda_i} \quad & \psi_i(\lambda_i) \\ \text{s.t.} \quad & \lambda_i = \lambda. \end{aligned} \tag{11}$$

Noticing that problem (11) is of the same form as problem (2), hence we could directly apply the derived solutions and proofs to problem (11). For this scenario, $\psi_i(\lambda_i)$ could be treated as the local machine learning problem and λ_i^t could be treated as the local learning result at iteration t. The final learning result is provided by λ^*, which could be used to find the model θ^*.

IV. Security Analysis

In this section, we present our design to guarantee that during this learning process, only the final learning result θ is revealed but nothing else. We assume the adversary is one of the learners, who is semi-honest and will collect data during collaborative training to trace back other's private training set X_i. The security analysis is conducted under the horizontally partitioned scenario and the adaption to vertically partitioned scenario is trivial because the later could be transformed into a dual problem with the same form as the horizontal case.

In the previous sections, we decompose the augmented Lagrange (3) in a way that local training of (4) is only related to the local cost function $J_i(\theta)$, r_i^t and the global knowledge z^t. Hence, we don't need to worry about privacy of X_i during this process because everything happens locally and learner i has never release X_i. However, to find the global knowledge z^t, we need to find the average of θ_i^t. Indeed, the only thing interesting for us is the average of local knowledge and there exist plenty of simple and efficient protocols to find the average value of a group of people without revealing individual values. For instance, assume there are M people who want to find the average of their private value v_i without revealing v_i to each other. This could be done by a simple secure sum protocol: pick randomly a person as initiator, who will pick a random value r and find $s := r + v_1$, and then, s is send to the next person who will update s as $s := s + v_2$. This process will continue until everybody has applied his/her private value to s, and s is sent back to the initiator, who will subtract s by the random number r to find the summation of their private value. Then, the difference is divided by M to find the average.

If z is calculated by the secure sum protocol, then the only thing available to an adversary is his own local training result for each round and the global knowledge of each iteration $\{z^t | t = 1, 2, ...T\}$. His local training result will not be useful because it is uncorrelated to other's private data. As iterations goes on, information about X revealed by the accumulated global knowledge equals to their mutual information, denoted as $I(X; z^T, z^{T-1}, ...z^1)$. Our goal is to show that as t goes on, there exists an upper bound of this mutual information, i.e. $I(X; z^T, z^{T-1}, ...z^1) \leq C$ for some constant C. Due to space limitation, we state a few theorems to help us develop our proof sketch, and these theorems will be proved in the possible journal version.

Theorem 4.1: When $t \geq t_\tau$, z^t is only related to the training data and the previous global knowledge z^{t-1} in a form of:

$$z^t = v^t X + z^{t-1} + n. \tag{12}$$

Theorem 4.2: When $t \geq t_\tau$, the representer coefficient v^t should satisfy

$$\lim_{t \to \infty} \frac{\|v^t\|}{\|v^{t-1}\|} < 1 \tag{13}$$

In (12), $v^t X$ denotes the knowledge generated from training data based on the representer theorem [10], n denotes the training error, which is assumed to be Gaussian. Basically, this theorem states that for iterations $t \geq t_\tau$, z^t is generated from a linear system with two sources. Theorem. 4.2 claims a property of v^t to ensure that z^t is converging that is $\|z^t - z^{t-1}\| \leq \|z^{t-1} - z^{t-2}\|$. Base on these two theorems, the revealed information should satisfy:

$$\begin{aligned} & \lim_{T \to \infty} I(X; z^T, z^{T-1}, ...z^1) \\ &= \sum_{t=t_\tau}^{\infty} I(X; z^t | z^{t-1}, ...z^1) + \sum_{t=1}^{t_\tau} I(X; z^t | z^{t-1}, ...z^1) \\ &= \sum_{t=t_\tau}^{\infty} h(z^t | z^{t-1}, ...z^1) - h(z^t | z^{t-1}, ...z^1, X) + C_1 \\ &\leq \sum_{t=t_\tau}^{\infty} h(v^t X + n) - h(n) + C_1 \\ &= \sum_{t=t_\tau}^{\infty} \log(1 + \frac{\|v^t\| \sigma_X^2}{\sigma_n^2}) + C_1 \\ &\leq \sum_{t=t_\tau}^{\infty} \frac{\|v^t\| \sigma_X^2}{\sigma_n^2} + C_1 = C. \end{aligned} \tag{14}$$

V. Simulation Results

In this section, we present the simulation results of our scheme against two popular data sets: a Higgs bosons presence data set [2] with 28 feature attributes and 11,000,000 data instances (which we only use 11,000 of them) and a OCR data set [1] with 64 feature attributes and 5620 data instances. For each data set, we assume that 50% of the data is distributed among 4 learners as the training data and we use the rest 50% for testing. Among these two training sets, OCR set is very easy to be classified, a centralized LR model is achieving 99.7% classification ratio. Higgs set is very noise and a centralized LR only achieves 64% classification ratio.

Among the 4 learners, learner 1's data is extremely unbalanced, which means it only has training samples of one class and absence of training samples of the other class. For each iteration, we test the local training result of learner 1 against the training set to study how knowledge is propagated to learner 1 as iterations goes on.

Two popular machine learning schemes are implemented under our framework, i.e. logistic regression (LR) and neural networks (NN). For these two schemes, LR is the one that fits our scheme better, because all our proofs in the previous section are based on an assumption that the cost function

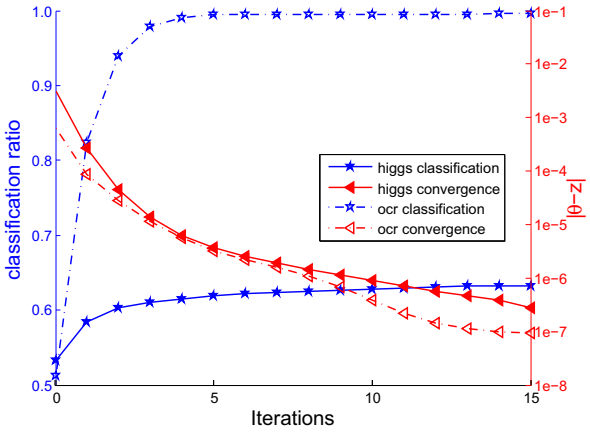

Fig. 1: Simulation results with logistic regression

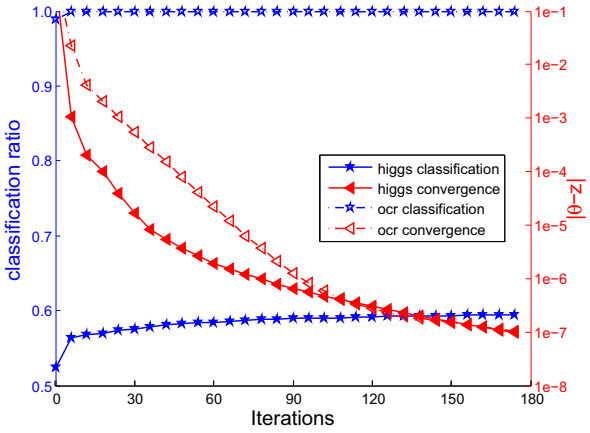

Fig. 2: Simulation results with neural networks

$J(\theta)$ is differentiable and convex. However, our simulation shows that even for a neural network with a non-convex cost function, our scheme could still generate a reasonably good model, even this model is different to the one provided by centralized training.

Fig. 1 plots the simulation results with logistic regression, in which curves marked with stars plot the classification ratios with the two data sets. We can see that for learner 1 with unbalanced training set, classification ratios are improving greatly as iterations goes on. Curves marked with triangles plot convergence of local training results. It shows that for LR, only tens of iterations are required to achieve consensus. Fig. 2 plots results with neural networks. It should be interesting to see that even the training set of learner 1 is unbalanced, classification ratio is as high as 99% at the very beginning. It may due to the very nature of neural networks: instead of learning difference between class 1 and class 2, it learns the structure of class 1 and hence know how to differentiate class 1 and class 2. knowledge propagation could be observed from curve of "higgs classification", which shows the slowly improvement of classification ratio. Compared with these two figures, we can see that knowledge is propagated much faster for LR (tens of iterations to converge) than NN (hundreds of iterations to converge). The reason is that cost function of LR is convex and the learners are working together to find this unique global optimum, however, the cost function of NN is non-convex such that the learners may not have a common target to converge.

VI. Conclusions

In this paper, we considered the scenario where multiple data holders were intended to find predictive models from their joint data without revealing their data to each other. We proposed a scheme in which data locality property was used as an alternative to multi-party computation (SMC) techniques. Specifically, the centralized learning task was distributed to each data holder as local learning tasks in a way that local learning was only related to local data. Along with that, an efficient and secure protocol was proposed to reassemble local results to get the final result. Correctness of our scheme has been proved theoretically and numerically. Security analysis has been conducted from the aspect of information theory to show that our scheme is secure.

References

[1] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[2] P. Baldi, P. Sadowski, and D. Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nat Commun*, 5, July 2014.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., New York, NY, USA, 1995.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, Jan. 2011.

[6] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June 2008.

[8] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Trans. Knowl. Data Eng.*, 16(9):1026–1037, 2004.

[9] Y. Lindell and B. Pinkas. Privacy preserving data mining. *J. Cryptology*, 15(3):177–206, 2002.

[10] B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*, COLT '01/EuroCOLT '01, pages 416–426, London, UK, UK, 2001. Springer-Verlag.

[11] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *KDD*, pages 639–644, 2002.

[12] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[13] K. Xu, H. Yue, L. Guo, Y. Guo, and Y. Fang. Privacy-preserving machine learning algorithms for big data systems. In *ICDCS*. IEEE, 2015.

[14] J. Yuan and S. Yu. Privacy preserving back-propagation neural network learning made practical with cloud computing. *Parallel and Distributed Systems, IEEE Transactions on*, 25(1):212–221, Jan 2014.

[15] J. Z. Zhan and S. Matwin. Privacy-preserving support vector machine classification. *IJIIDS*, 1(3/4):356–385, 2007.

[16] T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML 2004: PROCEEDINGS OF THE TWENTY-FIRST INTERNATIONAL CONFERENCE ON MACHINE LEARNING. OMNIPRESS*, pages 919–926, 2004.