Soft Computing
https://doi.org/10.1007/s00500-018-3233-7

FOCUS

Verifiable privacy-preserving single-layer perceptron training scheme in cloud computing

Xiaoyu Zhang[1] · Xiaofeng Chen[1] · Jianfeng Wang[1] · Zhihui Zhan[2] · Jin Li[3]

Abstract
With the advent of artificial intelligence, machine learning has been well explored and extensively applied into numerous fields, such as pattern recognition, image processing and cloud computing. Very recently, machine learning hosted in a cloud service has gained more attentions due to the benefits from the outsourcing paradigm. Based on cloud-aided computation techniques, the heavy computation tasks involved in machine learning process can be off-loaded into the cloud server in a pay-per-use manner, whereas outsourcing large-scale collection of sensitive data risks privacy leakage since the cloud server is semi-honest. Therefore, privacy preservation for the client and verification for the returned results become two challenges to be dealt with. In this paper, we focus on designing a novel privacy-preserving single-layer perceptron training scheme which supports batch patterns training and verification for the training results on the client side. In addition, adopting classical secure two-party computation method, we design a novel lightweight privacy-preserving predictive algorithm. Both two participants learns nothing about other's inputs, and the calculation result is only known by one party. Detailed security analysis shows that the proposed scheme can achieve the desired security properties. We also demonstrate the efficiency of our scheme by providing the experimental evaluation on two different real datasets.

Keywords Single-layer perceptron · Privacy preservation · Batch training · Verifiability · Cloud computing

1 Introduction

According to the report that the quantity of available data generated will be exceed 15 zettabytes by 2020 compared with 0.9 zettabytes in 2013 Adshead (2014). With the increasing amount of data generated by various equipments, machine learning techniques have been drawing more attentions. As we all known, machine learning is used to process abundant data and produce predictive models. Very recently, machine learning has been extensively applied in plenty of research fields (Chang et al. 2017a, b; Chang and Yang 2017; Chang et al. 2017), such as spam classification Yu and Xu (2008), disease diagnosis Fakoor et al. (2013), credit-risk assessment Yu et al. (2008). Generally speaking, machine learning techniques consist of two stages, i.e., training and prediction. Given a set of training data records and desired outputs, a predictive model can be finally derived after a series of iteration. In prediction paradigm, taking some new data as inputs the trained model can predict the classification or a certain continuous value. Especially, among numerous machine learning frameworks, neural network has gained much popularity due to its nice performance

Communicated by B. B. Gupta.

✉ Jin Li
jinli71@gmail.com

Xiaoyu Zhang
moliyanyan@163.com

Xiaofeng Chen
xfchen@xidian.edu.cn

Jianfeng Wang
jfwang@xidian.edu.cn

Zhihui Zhan
cszhanzhh@scut.edu.cn

[1] State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi'an 710071, People's Republic of China

[2] Guangdong Provincial Key Lab of Computational Intelligence and Cyberspace Information, School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, People's Republic of China

[3] School of Computational Science and Education Software, Guangzhou University, Guangzhou 510006, People's Republic of China

Published online: 16 May 2018

in many research goals. As one of the most simplest neural network tools, single-layer perceptron (SLP) Shynk (1990) has been successfully used to predict classification.

Due to the limited local storage and computing resources, cloud-based machine learning paradigm is becoming a newly developing research area. Cloud computing makes it possible to view computing as a kind of resource (Chen and Zhong 2009; Chen et al. 2016, 2015a, b, 2014a, b). In addition, the client can off-load their heavy computational tasks to the cloud server in a pay-per-use manner (Gao et al. 2018; Jiang et al. 2017, 2016, 2018; Li et al. 2015a, b, 2017a, b, 2016; Wang et al. 2015; Wen et al. 2014). Although there exist many benefits in cloud computing, this outsourcing paradigm may result in privacy leakage issue (Zhang et al. 2017a, b). In most cases, the inputs of the clients may contain some sensitive information and the cloud server is honest but curious. Therefore, considering the privacy protection into SLP training process in cloud computing is a significant challenge to deal with. Moreover, for some reasons such as hardware failures, software bugs or even malicious attacks, the cloud server may return a computationally indistinguishable result. In this case, the client should have the ability to check the validity of the returned result, which is a necessity in cloud-based SLP training process. Otherwise, outsourcing the complexity training task will become an impossible and meaningless issue.

Considering privacy protection in SLP training, traditional cryptographic primitives such as fully homomorphic encryption (FHE) can make it possible. However, the existing FHE schemes are not practical and efficient Wang et al. (2015). Recently, Zhang et al. (2018) proposed an efficient and privacy-preserving disease prediction scheme using SLP learning algorithm, named PPDP. In training stage, each medical sample is encrypted before uploading to the cloud server, which costs $O(n^3)$ on the hospital (client) side. It implies that if the number of iterative round is exactly equals to the number of training samples, it will make no sense to resort to the cloud server. The reason is that the most complicated calculation involved in SLP training stage costs $O(n^3)$ in Zhang et al. (2018). Besides, the verification mechanism is not considered in Zhang et al. (2018), and then the cloud server can deceive the hospital (client) by sending back an invalid result. Apart from that, the predictive model trained by the client, to some extent, should be regarded as the client's own asset and well protected during predictive stage. Moreover, since a new record is submitted by the requester, the predictive result should be protected and only be known by itself. Therefore, it is urgent and necessary to design an efficient and secure SLP training scheme which satisfies the aforementioned requirements.

1.1 Contributions

In order to address the issues mentioned above, we propose a verifiable privacy-preserving SLP training scheme (VPSPT) with the aid of the cloud server. Our contributions are summarized as follows.

- We propose a novel SLP training scheme, which can derive s predictive models for s different patterns simultaneously. Furthermore, based on the technique of mini-batch Mohassel and Zhang (2017), the trained model $\mathbf{w}$ can smoothly and rapidly converge to the optimum value. Compared with the scheme in Zhang et al. (2018), the computational complexity can be dramatically reduced from $O(n^3)$ to $O(n^2)$.
- We first introduce the verification mechanism into SLP training scheme. If the cloud server cheats the client by returning an incorrect value, the dishonest behavior will be detected by the client definitely.
- We design a lightweight privacy-preserving predictive algorithm based on secure two-party computation Malek and Miri (2006). With this method, both the predictive model $\mathbf{w}$ and the new data record can be well protected. Moreover, the final classification result is only privately hold by the requester.

The rest of this paper is organized as follows: Sect. 2 presents some basic notations and concepts involved in our scheme. The system and security models are given in Sect. 3. In the following, we propose a concrete scheme containing training and predictive stages in Sect. 4, followed by security and efficiency analysis in Sect. 5. Experimental evaluation is given in Sect. 6. Finally, conclusions will be made in Sect. 7.

1.2 Related work

Differing from traditional machine learning methods, cloud-aided privacy-preserving machine learning has been well explored and drawn more attentions. Clifton et al. (2002) presented a survey on some basic tools for privacy-preserving distributed data mining. As a toolkit, these techniques can be used to solve some privacy-preserving machine learning problems. Graepel et al. (2012) proposed a new class of machine learning algorithms where the predictions can be expressed as polynomials of bounded degree. Nikolaenko et al. (2013) designed a privacy-preserving ridge regression on hundreds of data records, which can be regard as a building block for many machine learning operations. Raymond et al. Tai et al. (2017) studied privacy-preserving decision trees evaluation via linear functions, and more. Generally speaking, privacy-preserving machine learning can be roughly divided into two research goals, data perturbation and cryptographic tools. The first method can be represented by differential privacy, which has been successfully applied into protecting the privacy of statistical database (Li et al. 2016; Zhang and Zhu 2017; Abadi et al. 2016). What we have to emphasize is that the first method is orthogonal to our work, and the readers can refer to related papers for further study.

The second research area is supported by cryptographic methods. Gupta et al. (2016) identified emergent research and techniques being utilized in the field of cryptology and cyber threat prevention. Zheng et al. (2017) proposed a lightweight authenticated encryption scheme based on chaotic SCML for railway cloud service. Ibtihal and Naanani (2017) focused on secure outsourcing of images by using homomorphic encryption in mobile cloud computing environment. Bhushan and Gupta (2018) proposed a flow confidence-based discrimination algorithm to distinguish between flash crowd event and DDoS attack. By incorporating Shamir's secret sharing and quantum byzantine agreement, AlZain et al. (2015) presented a practical data management model in a public and private multi-cloud environment. Lin et al. (2018) constructed a new ID-based linear homomorphic signature scheme, which avoided the shortcomings of the use of public-key certificates. Gao et al. (2018) proposed a privacy-preserving Naive Bayes classifier that is resistant to an easy-to-perform, but difficult-to-detect attack. Li et al. (2018) proposed a novel privacy-preserving Naive Bayes learning scheme with multiple data sources.

Based on the computational ability of participants, the second research field, cryptographic methods, can be split into two categories. The first scenario is training without the cloud server. That implies that all the participants are equal to each other in computational ability aspect. So far, plenty of works focus on this setting. Chen and Zhong (2009) proposed privacy-preserving backpropagation neural network learning scheme which allowed two parties jointly to train model over vertically partitioned data. In the following, based on their aforementioned work, Bansal et al. (2001) proposed a training scheme over arbitrarily partitioned data, which can be applied into more common scenes. In both two schemes (Bansal et al. 2001; Chen and Zhong 2009), the privacy of client can be guaranteed by using ElGamal scheme. Combined with several data-oblivious algorithms, Ohrimenko et al. (2016) designed a method to enable multiple parities to cooperatively conduct training program while each parties' privacy can be well protected. Very recently, based on the two servers model, Mohassel and Zhang (2017) proposed a system for scalable privacy-preserving machine learning. In this model, the data owners randomly distribute data to two non-conclude servers to train several models. Among them, they focused on training neural networks by using secure two-party or multiparty computation theory.

Obviously, the second scenario is where the training process involves in the cloud server. This implies that the resource-constrained client relies on the powerful cloud server to train models. Li et al. (2017) proposed multi-key privacy-preserving deep learning schemes. Liu et al. (2017) only explored the privacy-preserving predictive process, which requires no change to how models are trained. After prediction, the server learns nothing about client's inputs while the client learns nothing about the model. Very recently, considering the privacy protection of training stage, Wang et al. (2015) proposed a SLP learning scheme for e-healthcare. However, this scheme adopted Paillier homomorphic cryptosystem, which is time-consuming. In Zhang et al. (2018), a privacy-preserving disease prediction scheme in cloud-based e-healthcare system are proposed. Although this scheme provided the hospital (client) privacy protection in predictive stage, the privacy of patients (potential patients) was not considered, whereas in some specific scenarios it is necessary to design privacy-preserving algorithm in predictive stage. Besides, random matrices in Zhang et al. (2018) are utilized to encrypt training samples as we mentioned before, it is not efficient and practical.

2 Preliminaries

In this section, we will present some basic notations, machine learning and mathematical tools. Firstly, we briefly revisit the classical SLP learning algorithm which can be referred to in many standard machine learning textbooks Michalski et al. (2013). Furthermore, based on the basic idea in Mohassel and Zhang (2017), we propose a mini-batch SLP training scheme. In the following, we will introduce the privacy-preserving method for large-scale matrix multiplication in cloud computing Lei et al. (2014). Finally, we will give a secure dot-secure technique by using trace functions Malek and Miri (2006).

2.1 Mini-batch SLP training algorithm

In this section, we will recall the principles and some main properties of SLP. As one of the most important and simplest neural network architectures, SLP can be used as a powerful classifier whose output belongs to one class or another. Given a set of training data samples $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ with associated desired output $\{o_1, o_2, \ldots, o_n\}$ $(o_i \in \{1, -1\})$, the goal of the SLP training algorithm is to obtain a prediction model. In most cases, when a new data record is submitted to this prediction model, it can give an accurate classification result t. The basic framework of SLP is depicted in Fig. 1. And in the following, let's look the SLP training algorithm in detail.

As we can see from Fig. 1, the SLP consists of two layer neuron cells, input nodes and output node. Among these nodes, the input nodes are denoted as $\left\{x_{i,1}, x_{i,2}, \ldots, x_{i,n}\right\}$ which means each piece of sample $\mathbf{x}_i$ has n features. The out-

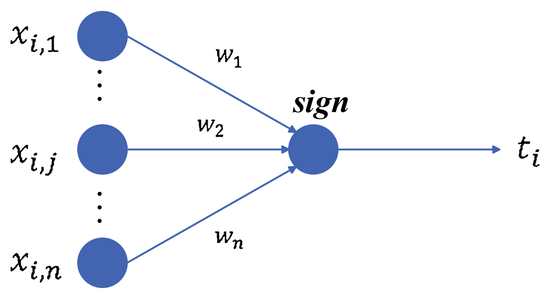

Fig. 1 Model of single-layer perceptron

put layer is a linear combination of input nodes and weight vector $\mathbf{w} = \{w_1, w_2, \ldots, w_n\}$. After that, a specific activation function is acted on the output node, and then it outputs the classification result $t_i \in \{1, -1\}$. In this paper, we select the sign function as the activation function since its simplicity and popularity.

$$t_i = sign(\mathbf{w}^T \mathbf{x}_i)$$

If $o_i \neq t_i$, the weight parameter w will be updated according to the following equation:

$$\mathbf{w} = \mathbf{w} + \eta \mathbf{x}_i o_i$$

Different from traditional SLP training algorithm, based on the intuition presented in Mohassel and Zhang (2017) we randomly select a small batch m of samples instead of a piece of sample per iteration, because the overwhelming advantage of mini-batch method is that it can be used to speed up the computation. Besides, with this method, the weight vector $\mathbf{w}$ can converge faster to the minimum value. Therefore, the weight vector $\mathbf{w}$ can be updated by averaging the partial derivatives of m samples on the current $\mathbf{w}$. For some $t_i \neq o_i$, the updating formula for weight $\mathbf{w}$ can be adapted as:

$$\mathbf{w} = \mathbf{w} + \frac{\eta}{m} \sum_{t_i \neq o_i} \mathbf{x}_i o_i$$

If it meets one of the two requirements, the number of iterations more than the preset threshold or the prediction model converges to a constant, the SLP training algorithm will be terminated. Furthermore, the elaborate mini-batch SLP training algorithm is described in Algorithm 1.

Algorithm 1 Mini-batch SLP training algorithm

Input:

- A set of training samples $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ with associated desired output $\{o_1, o_2, \ldots, o_n\}$ $(o_i \in \{1, -1\})$.
- Learning rate η.
- The size of mini-batch m.
- Preset iteration threshold p.

Output:

- Prediction model: $\mathbf{w}$.

```
1: Randomly select a weight vector $\mathbf{w} = \{w_1, w_2, \ldots, w_n\}$.
2: for $iteration = 1, 2, \ldots, p$ do
3:     for $1 \leq i \leq m$ do
4:         Compute the sign function:
               $t_i = sign(\mathbf{w}^T \mathbf{x}_i)$
5:     end for
6:     For some $o_i \neq t_i$, the client will update:
               $\mathbf{w} = \mathbf{w} + \frac{\eta}{m} \sum_{t_i \neq o_i} \mathbf{x}_i o_i$
7: end for
8: Return: Prediction model $\mathbf{w}$.
```

2.2 Privacy-preserving method for outsourcing matrix multiplication

Securely outsourcing large-scale matrix multiplication has been researched for many years (Atallah et al. 2002; Lei et al. 2014), which is commonly used as the building block in scientific and engineering fields. Next, we will introduce a completed protocol for securely outsourcing matrix multiplication which consists of five algorithms (**KeyGen**, **MMEnc**, **Compute**, **MMDec**, **Verify**) as follows.

- ***KeyGen***: On input the security parameter λ, the client randomly chooses three sets $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, $\{\beta_1, \beta_2, \ldots, \beta_n\}$ and $\{\gamma_1, \gamma_2, \ldots, \gamma_n\}$ from specific key space. By using the same method in Lei et al. (2014), the client generates three random permutations, π_1, π_2, π_3. Similarly, the client generates three sparse matrices, $\mathbf{F}_1(i, j) = \alpha_i \delta_{\pi_1(i),j}, \mathbf{F}_2(i, j) = \beta_i \delta_{\pi_2(i),j}, \mathbf{F}_3(i, j) = \gamma_i \delta_{\pi_3(i),j}$, where the formula of the Kronecker delta function $\delta_{x,y}$ is as follows.

$$\delta_{x,y} = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases}$$

- ***MMEnc***: Given two large-scale matrices $\mathbf{X}, \mathbf{Y}$, the resource-constrained client needs to calculate matrix multiplication. In order to protect his own private information, the client will encrypt his inputs before uploading them to the cloud server to compute with. Therefore, by using the matrix blinding technique the client computes $\hat{\mathbf{X}} = \mathbf{F}_1 \mathbf{X} \mathbf{F}_2^{-1}$ and $\hat{\mathbf{Y}} = \mathbf{F}_2 \mathbf{X} \mathbf{F}_3$ locally and sends the blinding inputs $\hat{\mathbf{X}}$, $\hat{\mathbf{Y}}$ to the cloud server.
- ***Compute***: After receiving two matrices $\hat{\mathbf{X}}, \hat{\mathbf{Y}}$ from the client, the cloud server conducts this algorithm to compute $\hat{\mathbf{T}} = \hat{\mathbf{X}}\hat{\mathbf{Y}}$. Subsequently, the cloud server sends the blinding result $\hat{\mathbf{T}}$ to the client.
- ***MMDec***: On input the returned result $\hat{\mathbf{T}}$, the client will decrypt it, $\mathbf{T} = \mathbf{F}_1^{-1} \hat{\mathbf{T}} \mathbf{F}_3^{-1} = \mathbf{XY}$. Therefore, the client will obtain the final result.

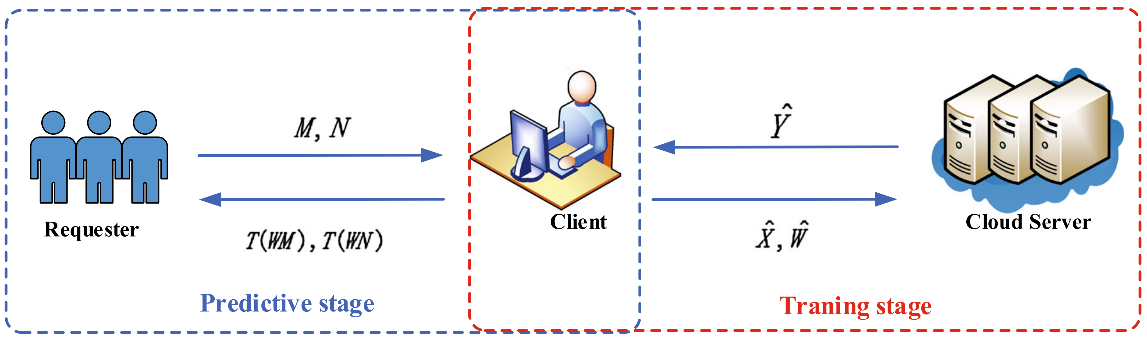

Fig. 2 Model of our VPSPT scheme

- ***Verify***: Considering the cloud server is honest but curious, after decrypting the result the client should check the correctness of the calculation result $\mathbf{T}$. The client firstly selects a vector $\mathbf{r} = \{r_1, r_2, \ldots, r_n\}$ and checks the equation $\mathbf{Tr} \overset{?}{=} \mathbf{XYr}$. If yes, the result $\mathbf{T}$ will pass verification; otherwise, this algorithm will output $\perp$.

2.3 Secure dot-product protocol

Definition 1 A trace function is a linear mapping from $\mathbb{F}_{p^n}$ over $\mathbb{F}_{p^q}$, where q can divide n. Let's denote that $\alpha \in \mathbb{F}_{p^n} = F$ and $\alpha \in \mathbb{F}_{p^q} = K$, then the trace of element α over K is:

$$T_{F/K}(\alpha) = \alpha + \alpha^p + \alpha^{p^2} + \cdots + \alpha^{p^{q-1}}$$

To simplify the representation, we denote the trace function by T. Furthermore, the above trace function has the four following properties:

- For $\alpha, \beta \in \mathbb{F}_{p^n}$, $T(\alpha + \beta) = T(\alpha) + T(\beta)$;
- For $\alpha \in \mathbb{F}_{p^n}$, $c \in \mathbb{F}_p$, $T(c\alpha) = cT(\alpha)$;
- For $a \in \mathbb{F}_p$, $T(a) = na$;
- For $\alpha, \in \mathbb{F}_{p^n}$, $T(\alpha^p) = T(\alpha)$.

Suppose that $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ is a basis of $\mathbb{F}_{p^n}$ over $\mathbb{F}_p$, and $\{\beta_1, \beta_2, \ldots, \beta_n\}$ is its dual basis that satisfies the following equation.

$$T(\alpha_i \beta_j) = \begin{cases} 1, \ for\ i \neq j; \\ 0, \ for\ i = j. \end{cases}$$

Then, for some $x_i, y_i \in \mathbb{F}_p$, $\mathbf{X}, \mathbf{Y} \in \mathbb{F}_{p^n}$ can be represented as

$$\mathbf{X} = x_1\alpha_1 + x_2\alpha_2 + \cdots + x_n\alpha_n$$
$$\mathbf{Y} = y_1\beta_1 + y_2\beta_2 + \cdots + y_n\beta_n$$

Most important of all, we have the following equation holds.

$$T(\mathbf{XY}) = \mathbf{x} \cdot \mathbf{y}$$

3 System and security models

In this section, we focus on formalizing the system model and security model.

3.1 System model

In this paper, we propose an efficient and secure VPSPT scheme, which consists of three entities, the client, the cloud server and the requester. Furthermore, the system model of VPSPT scheme is described as Fig. 2.

- *The client* The main task of the client is to train s prediction models for s different patterns. The client takes the training cases $\{x_{i,j}\}$ $(1 \leq i \leq n, 1 \leq j \leq m)$, random weight $\{w_{j,k}\}$ $(1 \leq j \leq m, 1 \leq k \leq s)$, learning rate η, the size of mini-batch n and the predetermined iteration round p as inputs. And the client takes a final weight matrix $\mathbf{W}$ for s different patterns as its output.
- *The cloud server* A cloud server possesses substantial computation and storage resources. With the help of the cloud server, the client can outsource the heavy computational operations in order to save the local resources by pay-per-use manner. Generally speaking, the cloud server is curious but honest. That is, the cloud server can follow the protocol honestly, but he will try his best to dig up some sensitive information beyond what he has known.
- *The requester* A requester who owns a new data record wants to know the classification result under a specific prediction model. On the one hand, the new data record is privately held by the requester. On the other hand, the specific prediction model belongs to the client's own asset

which costs the client substantial resources to obtain. Therefore, the requester should learn nothing about the prediction model other than the final result.

3.2 Security model

In training stage, we consider that the adversary is an untrusted server in honest but curious model Goldreich et al. (1987) (also called “semi-honest"). That is, the cloud server will faithfully follow the protocol, but he may try to learn additional information by analyzing the messages that he receives during the execution. In predictive process, we assume that both the client and the requester are honest but curious. On the one hand, the query record submitted by the requester may contain some private information and should not be leaked to others. On the other hand, the malicious requester may want to know the training model which is the client’s own asset. Therefore, in our threat model, it must be ensured that each party learns nothing beyond what they should know.

In this paper, our aim is to propose an efficient and verifiable SLP training scheme which can produce s prediction models for different patterns. In the meanwhile, we aim at guaranteeing the privacy protection both in training and predictive stages. Besides, the following properties should be satisfied.

– *Privacy* In training stage, we require that the client’s data are secure against the cloud server. Given the encrypted sample cases, the cloud server cannot get the client’s original data. Furthermore, the result is also hidden from the server. In predictive stage, both the new query record and prediction model should be well protected. That is, the two participants cannot learn nothing beyond what they have known.
– *Verifiability* Since the cloud server is semi-honest, the client should have the ability to detect errors. That is to say, any error result from a cheating cloud server cannot pass the verification.
– *Efficiency* In training process, for the client, the computation cost for preparing outsourcing calculation task to the cloud server and extracting the results from the returned values should be less than that of computing the computation task by its own.

4 The proposed VPSPT scheme

4.1 High description

In this section, we will outline the training process for s different models from a set of training sample cases. On the one hand, we adopt the main idea in Mohassel and Zhang (2017) to choose mini-batch cases instead of a piece of case per iteration. That is to say, using the stochastic gradient descent method we expand the sample vector $\mathbf{x} = \{x_1, x_2, \ldots x_n\}$ into the matrix $\mathbf{X} = \left\{x_{i,j}\right\}$ $(1 \leq i \leq n, 1 \leq j \leq m)$ to improve the iteration speed. On the other hand, since the same batch of cases can be used to train for different models, then we can train s different models $\mathbf{W} = \left\{w_{j,k}\right\}$ $(1 \leq j \leq m, 1 \leq k \leq s)$ simultaneously. In the training process, the client off-loads the heavy computation task to the cloud server with the help of the cloud computing architectures. Since the cloud servers are semi-honest, the client should conduct some blinding operations to encrypt input matrices $\mathbf{X}$ and $\mathbf{W}$ before uploading them. By using the random permutations and sparse matrix techniques which were firstly proposed by Atallah et al. (2002), we can achieve the aim of protecting privacy of the client.

Due to some financial reasons or hardware failures in cloud computing, the client must have the ability to detect errors in each iteration process. Compared to the existing works, we propose an efficient and verifiable SLP training algorithm. After decrypting the results returned by the cloud server $\mathbf{Y}$, the client randomly selects a vector $\mathbf{r}$ and checks whether the equation $\mathbf{XWr} = \mathbf{Yr}$ holds. If yes, the calculation result $\mathbf{Y}$ will pass the verification.

Next, for some misclassification cases, the client will update the weight parameter $\mathbf{w}_k$ following to the updating formula we mentioned before. If the training algorithm achieves the iteration termination condition, this algorithm will output s different models for s different patterns. Otherwise, the client will continue to conduct the training scheme for next round.

For a new coming case, based on trace function Malek and Miri (2006), we propose a lightweight privacy-preserving predictive algorithm. At the end of this algorithm, only will the requester know the prediction result. Besides, considering that the inputs of the requester contain some sensitive personal information, and the trained model $\mathbf{w}_k$ is owned by the client, it is essential to design a privacy-preserving predictive algorithm. By this way, the client learns nothing about the inputs of the requester, and vice versa.

4.2 Verifiable privacy-preserving SLP training scheme

In this section, we propose the VPSPT scheme, which is composed of three stages, initialization, training stage and predictive stage. Moreover, the elaborate training process and predictive stage are described in Algorithm 2 and Algorithm 3, respectively.

– *Initialization* Firstly, in order to protect the client’s sensitive information, it is necessary to encrypt the input information before uploading them to the cloud server. Therefore, the client conducts ***KeyGen*** algorithm to generate three secret sparse matrices $\mathbf{F}_1 \in \mathbb{R}^{n\times n}, \mathbf{F}_2 \in \mathbb{R}^{m\times m}$ and $\mathbf{F}_3 \in \mathbb{R}^{s\times s}$, which are used to blind input matrices. Secondly, the client randomly selects a weight matrix $\mathbf{W} \in \mathbb{R}^{m\times s}$ where all elements are equal to small random numbers.

– *Training stage* In the following, the completed protocol will be depicted. The privacy-preserving and verifiable SLP training scheme is described by Algorithm 2.

  – *Step 1* Based on the mini-batch idea in Mohassel and Zhang (2017), the client randomly selects a small batch of samples instead of a piece of data per iteration. The client chooses n pieces of training data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n\}$ with associated desired outputs $\{\mathbf{o}_1, \mathbf{o}_2, \ldots \mathbf{o}_n\}$, and each piece of training data has m feature values. Hence, we denote these training data as matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$. In order to protect the sensitive information of the client $\mathbf{X}$ and the training models $\mathbf{W}$, the client needs to conduct the ***MMEnc*** algorithm to obtain $\hat{\mathbf{X}} = \mathbf{F}_1\mathbf{X}\mathbf{F}_2$ and $\hat{\mathbf{W}} = \mathbf{F}_2^{-1}\mathbf{W}\mathbf{F}_3$, and then uploads the ciphertext tuple $\left\langle \hat{\mathbf{X}}, \hat{\mathbf{W}} \right\rangle$ to the cloud server.
  – *Step 2* Upon receiving the ciphertext tuple $\left\langle \hat{\mathbf{X}}, \hat{\mathbf{W}} \right\rangle$ from the client, the cloud server executes the matrix multiplication function, i.e., $\hat{\mathbf{Y}} = \hat{\mathbf{X}} \times \hat{\mathbf{W}}$. In the following, the cloud server sends the blinding training result $\hat{\mathbf{Y}}$ to the client.
  – *Step 3* In this step, the client conducts the decryption operation as $\mathbf{Y} = \mathbf{F}_1^{-1}\hat{\mathbf{Y}}\mathbf{F}_3^{-1}$ and derives the final result matrix $\mathbf{Y}$. Furthermore, in order to build confidence of the outsourcer, the client will check the correctness of the computation result $\mathbf{Y}$ which is returned by the cloud server. Firstly, the client randomly selects a vector $\mathbf{r} = \{r_1, r_2, \ldots r_s\}$ where not all elements are equal to zero. Secondly, the client locally calculates $\mathbf{Yr}$ and $\mathbf{XWr}$, respectively, and checks whether they are equal to each other. If yes, the computation result $\mathbf{Y}$ will pass the verification. Otherwise, the client will terminate this algorithm.
  – *Step 4* To simplify the representation, we select one column of matrix $\mathbf{Y}$ and denoted by $\mathbf{y}_k$. It implies that we just elaborately discuss the training process of a specific model and other models are trained by the same method. Later, for each element of the vector $\mathbf{y}_k$, the client executes the sign function as

$$t_{i,k} = sign(y_{i,k})(for 1 \le i \le n)$$

and then the client compares each element of the $\left\{t_{i,k}\right\}$ with the desired classification results $\left\{o_{i,k}\right\}$ detailedly. For some $t_{i,k} \neq o_{i,k}$ (for $1 \le i \le n$), the client updates the weight vector $\mathbf{w}_k$ as

$$\mathbf{w}_k = \mathbf{w}_k + \frac{\eta}{n}\sum_{t_{i,k}\neq o_{i,k}} \mathbf{x}_i o_{i,k}$$

If the weight vector $\mathbf{w}_k$ satisfies one of the two terminating conditions, i.e., the number of iteration round exceeds the preset value or all classification results for this model are correct, the SLP training algorithm will be terminated and go to step 5. Otherwise, the client will repeat the steps from step 1 with the help of the cloud server.
  – *Step 5* In this paper, we assume that these s models synchronously achieve the convergence condition or they have the same preset threshold. After several iterations by conducting above training process, finally, the client obtains s prediction models from a set of samples for s different patterns.

– *Predictive stage* To predict a new data record for the requester, based on the main idea in Malek and Miri (2006) we design a lightweight privacy-preserving predictive algorithm to obtain the classification result. The requester who owns the new data tuple $\mathbf{x} = \{x_1, x_2, \ldots x_n\}$ will cooperate with the client owned the predictive model $\mathbf{w} = \{w_1, w_2, \ldots w_n\}$ to conduct the predictive algorithm.
Finally, only will the requester know the final classification result. What's more, the client learns nothing about the requester's input and the requester learns nothing about the model within the whole process. Especially, the predictive algorithm consists of the following three steps.

  – *Step 1* Assume that $\{\alpha_1, \alpha_2, \ldots \alpha_n\}$ is a basis of $\mathbb{F}_{p^n}$ over $\mathbb{F}_{\mathrm{I}}$, and $\{\beta_1, \beta_2, \ldots \beta_n\}$ is its dual basis. Therefore, the two vectors $\mathbf{X}$ and $\mathbf{W}$ can be presented over $\mathbb{F}_{p^n}$ as

$$\mathbf{X} = x_1\alpha_1 + x_2\alpha_2 + \cdots + x_n\alpha_n$$
$$\mathbf{W} = w_1\beta_1 + w_2\beta_2 + \cdots + w_n\beta_n$$

The requester randomly chooses $\mathbf{Z} \in \mathbb{F}_{p^n}$, and $a, b, c, d \in \mathbb{F}_p$, s.t. $(ad-bc) \neq 0$. Next, the requester locally computes the two following messages

$$\mathbf{M} = a\mathbf{X} + b\mathbf{Z}$$
$$\mathbf{N} = c\mathbf{X} + d\mathbf{Z}.$$

Then, the requester sends the ciphertext tuple $\langle \mathbf{M}, \mathbf{N}\rangle$ to the client for prediction.
  – *Step 2* On receiving the ciphertext tuple $\langle \mathbf{M}, \mathbf{N}\rangle$ from the requester, the client who owns the model $\mathbf{w}$ will compute

$$\mathbf{WM} = \mathbf{W}(a\mathbf{X} + b\mathbf{Z})$$
$$\mathbf{WN} = \mathbf{W}(c\mathbf{X} + d\mathbf{Z}).$$

In the meanwhile, the client computes the trace function $T(\mathbf{WM})$, $T(\mathbf{WN})$ and sends them to the requester.

- *Step 3* After receiving the trace functions from the client, the requester who owns the random numbers a, b, c, d will compute the message

$$o = (ad - bc)^{-1}(d\, T(\mathbf{WM}) - b\, T(\mathbf{WN})).$$

Subsequently, the requester conducts the activation function, i.e., $t = sign(o)$. Therefore, the requester can obtain the final classification result t in secure manner without privacy information leakage. The detailed predictive algorithm is depicted in Algorithm 3.

4.3 Correctness

In this section, we will present the correctness of VPSPT scheme both in training stage and predictive stage, respectively.

- *Training stage* In step 2 and step 3, on receiving the blinding result $\hat{\mathbf{Y}}$, the client who possesses the secret keys $\mathbf{F}_1^{-1}$ and $\mathbf{F}_3^{-1}$ will conduct the following decryption operations.

$$\begin{aligned}\mathbf{F}_1^{-1}\hat{\mathbf{Y}}\mathbf{F}_3^{-1} &= \mathbf{F}_1^{-1}\hat{\mathbf{X}}\hat{\mathbf{W}}\mathbf{F}_3^{-1}\\ &= \mathbf{F}_1^{-1}\mathbf{F}_1\mathbf{X}\mathbf{F}_2\mathbf{F}_2^{-1}\mathbf{W}\mathbf{F}_3\mathbf{F}_3^{-1}\\ &= \mathbf{XW} = \mathbf{Y}\end{aligned}$$

By selecting a random vector $\mathbf{r}$, the client checks $\mathbf{Yr} \overset{?}{=} \mathbf{XWr}$. If the result passes verification, that means the client can derive a series of correct computational results. In addition, the rest of training tasks per round are conducted by the client locally.

- *Predictive stage* Next, we will illustrate the correctness of the predictive algorithm. After receiving two trace functions $T(\mathbf{WM})$ and $T(\mathbf{WN})$ from the client, the requester privately computes

$$\begin{aligned}o &= (ad - bc)^{-1}(d\, T(\mathbf{WM}) - b\, T(\mathbf{WN}))\\ &= (ad - bc)^{-1}(d\, T(\mathbf{W}(a\mathbf{X} + b\mathbf{Z}))\\ &\quad - b\, T(\mathbf{W}(c\mathbf{X} + d\mathbf{Z})))\\ &= (ad - bc)^{-1}(ad - bc)T(\mathbf{XW})\\ &= T(\mathbf{XW})\ mod\ p\\ &= \mathbf{x} \cdot \mathbf{w}\end{aligned}$$

Later, the requester carries out the sign function to achieve the final classification result $t = sign(o)$.

Algorithm 2 Verifiable privacy-preserving SLP training algorithm

Input:
- Blinding matrices $\mathbf{F}_1, \mathbf{F}_2, \mathbf{F}_3$.
- Training data $\{x_{i,j}\}$ ($1 \le i \le n, 1 \le j \le m$) with desired outputs $\{o_{i,k}\}$ ($1 \le i \le n, 1 \le k \le s$).
- Learning rate η.
- The size of mini-batch n.
- Preset iteration threshold p.

Output:
- Weight matrix $\mathbf{W}$

```
1: Randomly select the initial values of the weight matrix $\{w_{j,k}\}$ ($1 \le j \le m, 1 \le k \le s$).
2: for $iteration = 1, 2, \ldots, p$ do
3:     The client randomly selects a mini-batch of training data $\mathbf{X} \in \mathbb{R}^{n\times m}$ and computes $\hat{\mathbf{X}} = \mathbf{F}_1\mathbf{X}\mathbf{F}_2$, $\hat{\mathbf{W}} = \mathbf{F}_2^{-1}\mathbf{W}\mathbf{F}_3$
4:     The cloud server calculates $\hat{\mathbf{Y}} = \hat{\mathbf{X}}\hat{\mathbf{W}}$ and returns $\mathbf{Y}$ to the client.
5:     The client decrypts the final result $\mathbf{Y} = \mathbf{F}_1^{-1}\hat{\mathbf{Y}\mathbf{F}_3^{-1}}$. To verify the correctness of the result, the client generates a random vector $\mathbf{r} = \{r_1, r_2, \ldots, r_s\}$ and checks whether the equation $\mathbf{Yr} = \mathbf{XWr}$ holds.
6:     if Yes, then
7:         The result will pass the verification and the algorithm will continue;
8:     else
9:         Break;
10:    end if
11:    for $1 \le k \le s$ do
12:        for $1 \le i \le n$ do
13:            The client computes
                   $t_{i,k} = sign(y_{i,k})$
14:        end for
15:        For some $o_{i,k} \ne t_{i,k}$, the client will update:
                   $\mathbf{w}_k = \mathbf{w}_k + \frac{\eta}{n} \sum_{t_{i,k}\ne o_{i,k}} \mathbf{x}_i o_{i,k}$
16:    end for
17:    if the predictive model $\mathbf{w}_k$ satisfies the iterative terminating condition then
18:        this algorithm will be terminated.
19:    else
20:        The algorithm will jump to line 3.
21:    end if
22: end for
23: Return: $s$ prediction models $\mathbf{W}$.
```

5 Security and efficiency analysis

In this section, we will present the security and efficiency analysis for our proposed VPSPT scheme.

Algorithm 3 Lightweight privacy-preserving predictive algorithm

```
Input:
    – A new data tuple $\mathbf{x} = \{x_1, x_2, \ldots x_n\}$.
    – Predictive model $\mathbf{w} = \{w_1, w_2, \ldots w_n\}$.
Output:
    – Classification result $t$.
1: The requester randomly chooses $\mathbf{Z} \in \mathbb{F}_{p^n}$, and $a, b, c, d \in \mathbb{F}_p$, s.t. $(ad - bc) \neq 0$. And the requester locally computes:
       $\mathbf{M} = a\mathbf{X} + b\mathbf{Z}$
       $\mathbf{N} = c\mathbf{x} + d\mathbf{Z}$
2: The requester sends the ciphertext tuple $\langle \mathbf{M}, \mathbf{N} \rangle$ to the client for prediction.
3: The client computes:
       $\mathbf{WM} = \mathbf{W}(a\mathbf{X} + b\mathbf{Z})$
       $\mathbf{WN} = \mathbf{W}(c\mathbf{X} + d\mathbf{Z})$.
4: In the meanwhile, the client computes the trace functions $T(\mathbf{WM})$, $T(\mathbf{WN})$ and sends them to the requester.
5: Then, the requester computes
       $o = (ad - bc)^{-1}(d\,T(\mathbf{WM}) - b\,T(\mathbf{WN}))$
6: Finally, the requester conducts the sign function
       $t = sign(o)$
7: return $t$;
```

5.1 Security analysis

In training and predictive stages, the training sample cases contain some private information. And the training process of the prediction models also requires substantial resources. In other words, these trained models are valuable assets owned by the client. In addition, the query record submitted by the requester contains some personal private information. Therefore, the training sample cases, prediction models and the query record should be well protected. That is, the cloud server, the client and the requester cannot learn anything other than what they have known.

In fact, in VPSPT scheme, most of the training process are carried out on the client side only apart from the process of outsourcing complicated computation. Therefore, we will elaborately present the security analysis for the whole process of outsourcing computation.

Theorem 1 *The proposed training algorithm can ensure the input and output privacy of the client.*

Proof Considering that the client encrypts two input matrices $\mathbf{X}$ and $\mathbf{W}$, the semi-honest cloud server cannot recover the original matrices. Concretely, the client's samples matrix $\mathbf{X}$ is multiplied by two sparse matrices $\mathbf{F}_1$ and $\mathbf{F}_2$. In other words, each element in matrix $\mathbf{X}$ is rearranged under both the row and column permutations. In addition, each element is further blinded by multiplying a factor, *i.e.*, (α_i/β_k). These entries both in matrices $\mathbf{F}_1$ and $\mathbf{F}_2$ are all not zero, and there exists exactly one nonzero value in each row or column. That implies that if an attacker launches a brute-force attack to obtain the two secret key sets $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ and $\{\beta_1, \beta_2, \ldots, \beta_n\}$, the success probability is $\frac{1}{|K_\alpha|^n |K_\beta|^n}$. And furthermore, the success probability of recovering the original position in matrix $\mathbf{X}$ is $((n!)^2)$. Obviously, the security level for input privacy depends on the size of key space. A choice of enough large key spaces $|K_\alpha|$ and $|K_\beta|$ can threat this attack effectively. Likewise, the input privacy of weight matrix $\mathbf{W}$ can be guaranteed in the same way. Similarly, the semi-honest cloud server cannot recover the final result $\mathbf{Y}$ from the blinded result $\hat{\mathbf{Y}}$. In this paper, we will omit the security analysis for output privacy since it can be analyzed in the same way with that of input privacy. Supposed that even the cloud server know the final result $\mathbf{Y}$ for some other reasons, it does not make any sense for the cloud server. Because the desired output for training sets is privately held by the client, and the iterative process of weight matrix $\mathbf{W}$ is also conducted on the client side. □

Theorem 2 *The privacy of requester can be guaranteed in the proposed predictive algorithm.*

Proof Given the ciphertext tuple $\langle \mathbf{M}, \mathbf{N} \rangle$, for some random number a, b, c, d and suitable choice of $\mathbf{Z}$, there exist many $\mathbf{X} \in \mathbb{F}_{p^n}$ that can ensure the following matrix equation holds.

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \mathbf{X} \\ \mathbf{Z} \end{pmatrix} = \begin{pmatrix} \mathbf{M} \\ \mathbf{N} \end{pmatrix}$$

The process of guessing the true value for $\mathbf{X}$ can be considered as solving the above equations. We use the symbol S for marking the set of all $\mathbf{X}$'s when given the same $\langle \mathbf{M}, \mathbf{N} \rangle$. Therefore, the success probability of the client (adversary) is $\frac{1}{|S|}$. In order to make the right guess for $\mathbf{X}$, the client has to have the ability to find all invertible matrices of two multiplied by two over $\mathbb{F}_p$. The number of invertible 2-by-2 matrices over $\mathbb{F}_p$ is $(p^2 - 1)(p^2 - 1) = \mathbf{O}(p^4)$. Without any doubt, the number of invertible 2-by-2 matrices represents the size of S. This implies that we can reduce the success probability of the adversary by increasing the size of S. For the convenience of understanding, we assume that the requester's inputs are selected from $\mathbb{F}_{p^m}$ instead of $\mathbb{F}_p$. Similar to above analysis, the size of S is changed into $(p^{2m} - 1)(p^{2m} - p^m) = \mathbf{O}(p^{4m})$. Therefore, we have

$$\mathrm{Adv}_{\mathrm{client}} = \frac{1}{p^{4m}} < \frac{1}{\mathrm{poly(m)}} \text{ (for } m > m_0).$$

In practice, 2^{80} can be considered to resist brute-force attack. At least, we choose $p = 2$ and $m_0 = 20$ in this paper. The requester's choice is evenly distributed over $S \subset \mathbb{F}_{p^n}$. So far, we have presented the security analysis for the requester. □

In the following, we will present the security analysis for the client. The reason is that the predictive model is the client's own asset. The predictive model $\mathbf{w}$ should be privately held by the client, and the requester learns nothing about $\mathbf{w}$ in the execution of the predictive algorithm.

Theorem 3 *The privacy of client can be guaranteed in the proposed predictive algorithm.*

Proof On receiving the trace functions $T(\mathbf{WM})$ and $T(\mathbf{WN})$, the aim of malicious requester is to find the predictive model $\mathbf{w}$. As mentioned earlier, the trace function is a linear mapping from $\mathbb{F}_{p^n}$ onto $\mathbb{F}_p$. Then there are $\frac{|\mathbb{F}_{p^n}|}{p} = p^{n-1}$ elements in $\mathbb{F}_{p^n}$ which have the same trace function values as $T(\mathbf{WM})$, where we denote the size of field $\mathbb{F}_{p^n}$ by $|\mathbb{F}_{p^n}|$. And then the malicious requester can further increase the probability of guessing the correct $\mathbf{w}$ by using the second message $T(\mathbf{WN})$. More specifically, there are p^{n-2} elements in $\mathbb{F}_{p^n}$ that meet the both requirements.

$$\begin{cases} T(\mathbf{W}^*\mathbf{M}) = T(\mathbf{WM}) \\ T(\mathbf{W}^*\mathbf{N}) = T(\mathbf{WN}) \end{cases}$$

It's obvious that the advantage of a malicious requester successfully guessing the predictive model can be made arbitrarily small by increasing n. Furthermore, we let $n = km$, we can achieve the desired security by increasing m. The security parameter m satisfies the following:

$$\mathrm{Adv}_{\mathrm{requester}} = \frac{1}{p^{\mathrm{km}-2}} < \frac{1}{\mathrm{poly(m)}} \; (for\ m > m_0).$$

□

5.2 Efficiency analysis

In this section, we will present the computation overhead and communication costs of the VPSPT scheme in one round as well as the process of the predictive stage. Before terminating, the VPSPT scheme will be conducted one round by one round. In this paper, we only consider the scenario where the amount of predetermined iterative rounds for s different models are identical to each other. In other words, we preset a constant threshold for s different models before conducting the VPSPT scheme.

- *Computation overhead* We will illustrate the computation cost containing three phases, initialization, training and prediction in Table 1. In the following, we will present the detailed efficiency analysis. In addition, we denoted by G an operation of generating a random number, M an operation of multiplication, E an operation of exponentiation, I an operation of inversion over finite field. In initialization phase, we call the ***KeyGen*** algorithm to generate three secret sparse matrices $\mathbf{F}_1 \in \mathbb{R}^{n\times n}$, $\mathbf{F}_2 \in \mathbb{R}^{m\times m}$ and $\mathbf{F}_3 \in \mathbb{R}^{s\times s}$, which cost $(n+m+s)$G in total.
  In Step 1, in order to protect the sensitive information in training samples $\mathbf{X}$ and s training models $\mathbf{W}$, the client conducts encryption operations, which costs $(nm+ms)$M. In step 2, after receiving the blinding inputs, the cloud server conducts the computation task according to the protocol. In fact, the cloud server multiplies $\hat{\mathbf{X}}$ with $\hat{\mathbf{W}}$ and achieves the blinding result $\hat{\mathbf{Y}}$, which costs (nms)M. In Step 3, the client extracts the final result $\mathbf{Y}$ from the blinding result $\hat{\mathbf{Y}}$ by computing $\mathbf{F}_1^{-1}\hat{\mathbf{Y}}\mathbf{F}_3^{-1}$, which costs $(2ns)$M. Since the cloud server is always semi-honest, it is necessary for the client to build the verification mechanism and check whether the returned result is correct or not. Therefore, the client conducts verification algorithm which costs $(3ns)$M to verify the result of decryption $\mathbf{Y}$. In Step 4, the client conducts sign function to achieve the classification result $t_{i,k}$ for training model k. For some incorrect classification results, the client updates the values of current model k. Especially, the computational overhead in this step is ranging from 0 to nM corresponding to the number of incorrect classification result ranging from 0 to n. So far, we have presented the detailed efficiency analysis for training process in each round. Besides, the rest of training process before arriving at the terminating condition is identical to the mentioned process.
  Next, we will introduce the efficiency analysis for our privacy-preserving predictive algorithm. Before submitting the data record to the client, the requester need to deal with some encryption operations, which costs $(4n)$M computational complexity. In the following, the client multiplies his own predictive model $\mathbf{W}$ with the coming data record $\langle\mathbf{M},\mathbf{N}\rangle$, which costs $(2n)$M. In addition, in order to assist the requester with computing the classification result, the client needs to spend $(2n)$E to compute two trace functions $T(\mathbf{WM})$ and $T(\mathbf{WN})$. Finally, the requester computes the final result locally, which costs (5M+1I).
- *Communication overhead* The communication overhead in three stages is also described in Table 1. As we can see from this table, both in training stage and predictive stage are only involved in one interaction by come-and-go manner. In training process, the client off-loads the complicated computation task to the cloud server by uploading the blinding inputs matrices $\hat{\mathbf{X}}$ and $\hat{\mathbf{W}}$, which costs $(nm+ms)$. The cloud server calculates the matrix multiplication and sends back the blinding result $\hat{\mathbf{Y}}$ to the client, costing (ns). In predictive stage, the requester submits his data to predict the result with the cost of $(2n)$. Subsequently, the client returns the two messages $T(\mathbf{WM})$ and $T(\mathbf{WN})$ to the requester with the cost of 2.

Table 1 Efficiency analysis for VPSPT scheme per round

Phase	Step	Entity	Computation cost	Communication cost
Initialization	–	Client	$n+m+s$G	–
Training	Step 1	Client	$nm+ms$M	$nm+ms$
	Step 2	Server	nmsM	ns
	Step 3	Client	$5ns$M	–
	Step 4	Client	$\leq n$M	–
Prediction	Step 1	Requester	$4n$M	2n
	Step 2	Client	$2n$M+$2n$E	2
	Step 3	Requester	5M+1I	–

Table 2 Efficiency comparison for two schemes per round

	Party	Scheme in Zhang et al. (2018)	Our scheme
Computation (initialization)	Hospital (client)	n^3M	$(n+m+s)$G
Computation(training)	Hospital (client)	n^3M	$(nm+ms+5ns)$M
	Cloud server	n^4M	nmsM
Verification	–	No	Yes
Privacy in prediction	–	No	Yes

Firstly, compared to the scheme in Zhang et al. (2018), our VPSPT scheme has plenty of advantages in computational cost among three phases. In Table 2, we will present the computation comparison between two schemes. We have analyzed the entire computational overhead in our scheme above, and the readers can refer to Zhang et al. (2018) for more details. We remark that the dimension of the training sample vector in Zhang et al. (2018) is also denoted by n. Secondly, in VPSPT scheme, the result returned by the cloud server can be checked by the client in order to avoid cheating by the malicious attacker. Finally, in predictive stage, both two participants' privacy protection are considered in our scheme.

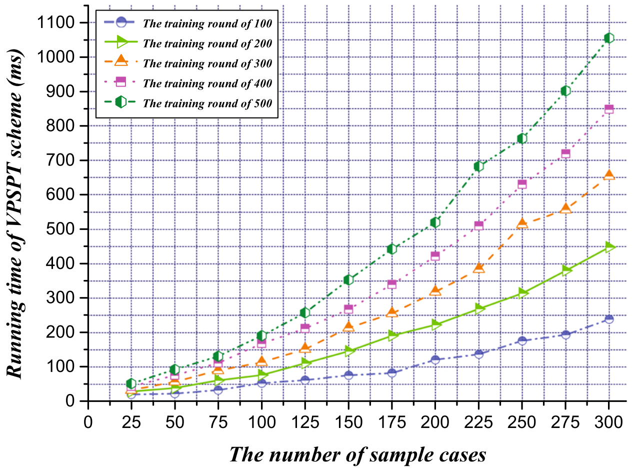

Fig. 3 Time cost for dataset A varying with the number of sample cases

6 Performance evaluation

In this section, we provide an experimental evaluation of the proposed VPSPT scheme by using Java language. We run the cloud server side on a computer with Intel Xeon(R) E5-1620 CPU processor running at 3.50GHZ, 16GB RAM memory. And the client side is operated on a laptop with Intel(R) Core(TM) i7-4770 CPU processor running at 3.40GHz, 16GB RAM memory. In our experiment, two real datasets are conducted to training several models simultaneously. We set the iteration threshold as 100, 200, 300, 400 and 500, respectively. These two real datasets are utilized from the hospital repository.

The first real dataset A includes 300 instances with each instance containing 13 attributes: AST, ALT, $\nu-$GT TG, TC, HDL, LDL, VLDL, FFA, FBG, BUN, UA, IL-6. In this experiment, 7 disease prediction models can be trained, i.e., we let $n = 300, m = 13, s = 7$. The running time of our VPSPT training scheme with different numbers of sample cases is depicted in Fig. 3. As we can seen from this, the running time of VPSPT time of 100 rounds is ranging from 20 to 239 ms with the amount of sample cases varying from 25 to 300. The running time of VPSPT time of 500 rounds is ranging from 52 to 1055 ms with the amount of sample cases varying from 25 to 300. Moreover, we also give the running time in training stage varies with the number of symptom attributes, where $3 \leq N_a \leq 13$. As we can see from Table 3, for 300 sample cases in our VPSPT scheme, the time cost of 200 training rounds ranges from 180 to 435 ms and the time cost of 500 training rounds ranges from 454 to 1060 ms. The experiment results are elaborately sketched in Fig. 4.

Table 3 Running time of training algorithm for two datasets

Time cost for training stage (ms)												
Number of attributes		3	4	5	6	7	8	9	10	11	12	13
Dataset A	200 rounds	180	214	238	265	302	307	336	352	379	407	435
	500 rounds	454	535	594	732	709	803	826	892	977	1026	1060
Number of attributes		4	10	16	22	28	34	40	46	52	58	64
Dataset B	200 rounds	314	476	643	800	974	1166	1317	1495	1617	1860	2046
	500 rounds	804	1210	1593	1996	2419	2939	3327	3716	4187	4697	5190

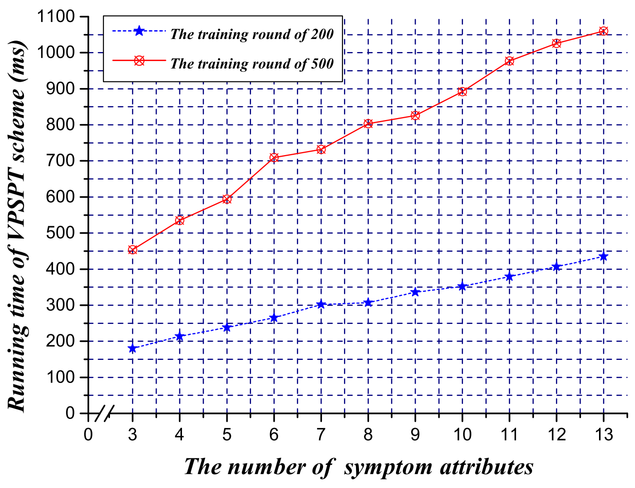

Fig. 4 Time cost for dataset A varying with the number of symptom attributes

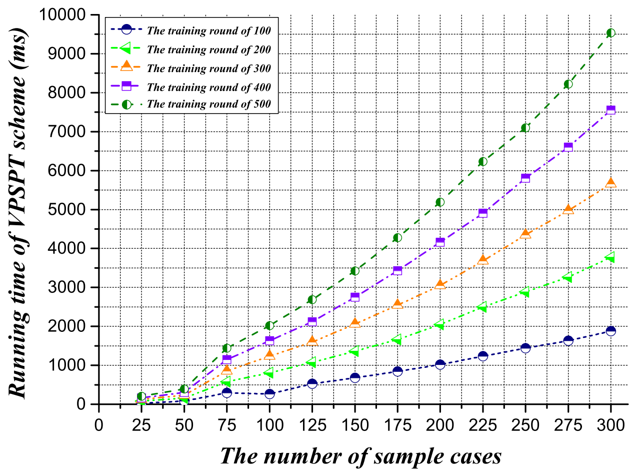

Fig. 5 Time cost for dataset B varying with the number of sample cases

The second real dataset B includes 300 instances with each instance containing 64 attributes. In this experiment, 26 disease prediction models can be trained synchronously, i.e., we let $n = 300, m = 64, s = 26$. The running time of our VPSPT training scheme is also depicted in Fig. 5. The running time of VPSPT time of 100 rounds is ranging from 81 to 1880 ms and the running time of VPSPT time of 500 rounds is ranging from 205 to 9537 ms with the amount of sample cases varying from 25 to 300. Moreover, we also give the running time in training stage varies with the number of symptom attributes, where $4 \leq N_b \leq 64$. As we can see from Table 3, for 300 sample cases in our VPSPT scheme, the time cost of 200 training rounds ranges from 314ms to 2046ms and the time cost of 500 training rounds ranges from 804ms to 5190ms with the amount of symptom attributes varying from 4 to 64. The experiment results are elaborately sketched in Fig. 6.

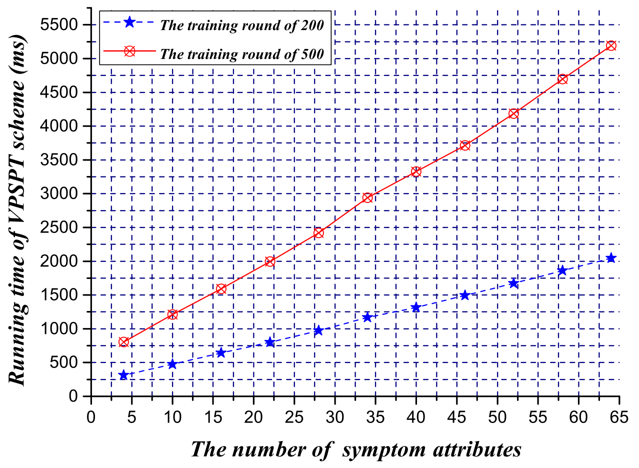

Fig. 6 Time cost for dataset B varying with the number of symptom attributes

7 Conclusion

In this paper, we propose a verifiable privacy-preserving single-layer perceptron training scheme. For a set of training samples, we can obtain s different models. Meanwhile, with the help of the cloud-aided computing technique, most of heavy computation tasks are transferred from the client to the cloud server. Therefore, the overhead of computation has been dramatically reduced. In predictive stage, both two participants can protect their inputs without privacy leakage. Moreover, only will the requester obtain the predictive result which is also private. In the future, we will focus on devoting ourselves to design more efficient and flexible machine learning schemes.

Acknowledgements This work is supported by National Natural Science Foundation of China (Grant Nos. 61572382, 61702401), Key Project of Natural Science Basic Research Plan in Shannxi Province of China (No. 2016JZ021) and China 111 Project (Grant No. B16037).

Compliance with ethical standards

Conflict of interest All authors declare that there is no conflict of interest.

Ethical standard This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abadi M, Chu A, Goodfellow I, McMahan H, Mironov I, Talwar K, Zhang L (2016) Deep learning with differential privacy. In: Proceedings of 2016 ACM SIGSAC Conference on Computer and Communications Security, pp 308–318

Adshead A (2014) Data set to grow 10-fold by 2020 as internet of things takes off. ComputerWeekly.com, 9

AlZain M, Li A, Soh B, Pardede E (2015) Multi-cloud data management using Shamir's secret sharing and quantum byzantine agreement schemes. Int J Cloud Appl Comput 5(3):35–52

Atallah M, Pantazopoulos K, Rice J, Spafford E (2002) Secure outsourcing of scientific computations. Adv Comput 54:215–272

Bansal A, Chen T, Zhong S (2001) Privacy preserving back-propagation neural network learning over arbitrarily partitioned data. Neural Comput Appl 20(1):143–150

Bhushan K, Gupta B (2018) A novel approach to defend multimedia flash crowd in cloud environment. Multimed Tools Appl 77(4):4609–4639

Chang X, Ma Z, Lin M, Yang Y, Hauptmann A (2017) Feature interaction augmented sparse learning for fast kinect motion detection. IEEE Trans Image Process 26(8):3911–3920

Chang X, Ma Z, Yang Y, Zeng Z, Hauptmann A (2017) Bi-level semantic representation analysis for multimedia event detection. IEEE Trans Cybern 47(5):1180–1197

Chang X, Yang Y (2017) Semisupervised feature analysis by mining correlations among multiple tasks. IEEE Trans Neural Netw Learn Syst 28(10):2294–2305

Chang X, Yu Y, Yang Y, Xing E (2017) Semantic pooling for complex event analysis in untrimmed videos. IEEE Trans Pattern Anal Mach Intell 39(8):1617–1632

Chen T, Zhong S (2009) Privacy-preserving backpropagation neural network learning. IEEE Trans Neural Netw 20(10):1554–1564

Chen X, Li J, Weng J, Ma J, Lou W (2016) Verifiable computation over large database with incremental updates. IEEE Trans Comput 65(10):3184–3195

Chen X, Li J, Huang X, Ma J, Lou W (2015) New publicly verifiable databases with efficient updates. IEEE Trans Dependable Sec Comput 12(5):546–556

Chen X, Huang X, Li J, Ma J, Lou W, Wong D (2015) New algorithms for secure outsourcing of large-scale systems of linear equations. IEEE Trans Inf Forensics Secur 10(1):69–78

Chen X, Li J, Ma J, Tang Q, Lou W (2014) New algorithms for secure outsourcing of modular exponentiations. IEEE Trans Paral Distr Syst 25(9):2386–2396

Chen X, Zhang F, Susilo W, Tian H, Li J, Kim K (2014) Identity-based chameleon hashing and signatures without key exposure. Inf Sci 265:198–210

Clifton C, Kantarcioglu M, Vaidya J, Lin X, Zhu M (2002) Tools for privacy preserving distributed data mining. Sigkdd Explor Newsl 4(2):28–34

Fakoor R, Ladhak F, Nazi A, Huber M (2013) Using deep learning to enhance cancer diagnosis and classification. InProceedings of the 30th International Conference on Machine Learning, 28

Gao CZ, Cheng Q, Li X, Xia SB (2018) Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network. Clust Comput. https://doi.org/10.1007/s10586-017-1649-y

Gao CZ, Cheng Q, He P, Susilo W, Li J (2018) Privacy-preserving naive bayes classifiers secure against the substitution-then-comparison attack. Inf Sci. https://doi.org/10.1016/j.ins.2018.02.058

Goldreich O, Micali S, Wigderson A (1987) How to play any mental game. In: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp 218–229

Graepel T, Lauter K, Naehrig M (2012) ML zon encrypted data. In: Proceedings of the International Conference on Information Security Cryptol, pp 1–21

Gupta B, Agrawal DP, Yamaguchi S (2016) Handbook of research on modern cryptographic solutions for computer and cyber security. IGI Global, Hershey

Ibtihal M, Naanani H (2017) Homomorphic encryption as a service for outsourced images in mobile cloud computing environment. Int J Cloud Appl Comput 7(2):27–40

Jiang T, Chen X, Wu Q, Ma J, Susilo W, Lou W (2017) Secure and efficient cloud data deduplication with randomized tag. IEEE Trans Inf Foren Secur 12(3):532–543

Jiang T, Chen X, Ma J (2016) Public integrity auditing for shared dynamic cloud data with group user revocation. IEEE Trans Comput 65(8):2363–2373

Jiang J, Wen S, Yu S, Xiang Y, Zhou W (2018) Rumor source identification in social networks with time-varying topology. IEEE Trans Dependable Sec Comput 15(1):166–179

Lei X, Liao X, Huang T, Heriniaina F (2014) Achieving security, robust cheating resistance, and high-efficiency for outsourcing large matrix multiplication computation to a malicious cloud. Inf Sci 280:205–217

Li N, Lyu M, Su D, Yang W (2016) Differential privacy: from theory to practice. Synth Lect Inf Secur Priv Trust 8(4):1–138

Li J, Li Y, Chen X, Lee P, Lou W (2015) A Hybrid cloud approach for secure authorized deduplication. IEEE Trans Paral Distr Syst 26(5):1206–1216

Li J, Chen X, Huang X, Tang S, Xiang Y, Hassan M, Alelaiwi A (2015) Secure distributed deduplication systems with improved reliability. IEEE Trans Comput 64(12):3569–3579

Li Z, Nie F, Chang X, Yang Y (2017) Beyond trace ratio: weighted harmonic mean of trace ratios for multiclass discriminant analysis. IEEE Trans Knowl Data Eng 29(10):2100–2110

Li H, Lu R, Misic JV (2017) Guest editorial big security challenges in big data era. IEEE Internet Things J 4(2):521–523

Li H, Yang Y, Luan TH, Liang X, Zhou L, Shen X (2016) Enabling fine-grained multi-keyword search supporting classified subdictionaries over encrypted cloud data. IEEE Trans Dependable Sec Comput 13(3):312–325

Li T, Li J, Liu Z, Li P, Jia C (2018) Differentially private naive bayes learning over multiple data sources. Inf Sci 444:89–104

Li P, Li J, Huang Z, Li T, Gao CZ, Yiu SM, Chen K (2017) Multi-key privacy-preserving deep learning in cloud computing. Future Gener Comput Syst 74:76–85

Lin Q, Yan H, Huang Z, Chen W, Shen J, Tang Y (2018) An ID-based linearly homomorphic signature scheme and its application in blockchain. IEEE Access. https://doi.org/10.1109/ACCESS.2018.2809426

Liu J, Juuti M, Lu Y, Asokan N (2017) Oblivious neural network predictions via minionn transformations. In: Proceedings of the 2017 ACM SIGSAC Conference on Conference Computer Communications Security, pp 619–631

Malek B, Miri A (2006) Secure dot-product protocol using trace functionsm. In: Information Theory, 2006 IEEE International Symposium, pp 927–931

Michalski R, Carbonell J, Mitchell T (2013) Machine learning: an artificial intelligence approach. Springer, Berlin

Mohassel P, Zhang Y (2017) SecureML: A System for scalable privacy-preserving machine learning. In: 2017 IEEE Symposium on Security and Privacy, pp 19–38

Nikolaenko V, Weinsberg U, Ioannidis S, Joye M, Boneh D, Taft N (2013) Privacy-preserving ridge regression on hundreds of millions of records. In: Proceedings of the Security and Privacy, pp 334–348

Ohrimenko O, Schuster F, Fournet C, Mehta A, Nowozin S, Vaswani K, Costa M (2016) Oblivious multi-party machine learning on trusted processors. In: USENIX Security Symposium, pp 619–636

Shynk J (1990) Performance surfaces of a single-layer perceptron. IEEE Trans Neural Netw 1:268–274

Tai R, Ma J, Zhao Y, Chow S (2017) Privacy-preserving decision trees evaluation via linear functions. In: Proceedings of European Symposium on Research in Computer Security, pp 494–512

Wang G, Lu R, Huang C (2015) PSLP: Privacy-preserving single-layer perceptron learning for e-Healthcare. In: Proceedings of information, communications and signal processing (ICICS), pp 1–5

Wang J, Chen X, Huang X, You I, Xiang Y (2015) Verifiable auditing for outsourced database in cloud computing. IEEE Trans Comput 64(11):3293–3303

Wen S, Jiang J, Xiang Y, Yu S, Zhou W, Jia W (2014) To shut them up or to clarify: restraining the spread of rumors in online social networks. IEEE Trans Parallel Distrib Syst 25(12):3306–3316

Yu B, Xu Z (2008) A comparative study for content-based dynamic spam classification using four machine learning algorithms. Knowl Based Syst 21(4):355–362

Yu L, Wang S, Lai K (2008) Credit risk assessment with a multistage neural network ensemble learning approach. Expert Syst Appl 34(2):1434–1444

Zhang C, Zhu L, Xu C, Lu R (2018) PPDP: an efficient and privacy-preserving disease prediction scheme in cloud-based e-Healthcare system. Future Gener Comput Syst 79:16–25

Zhang T, Zhu Q (2017) Dynamic differential privacy for ADMM-based distributed classification learning. IEEE Trans Inf Foren Secur 12(1):172–187

Zhang X, Jiang T, Li KC, Chen X (2017) New publicly verifiable computation for batch matrix multiplication. In: Proceedings of the GPC'17, pp 53–65

Zhang X, Jiang T, Li KC, Castiglione A, Chen X (2017) New publicly verifiable computation for batch matrix multiplication. Inf Sci. https://doi.org/10.1016/j.ins.2017.11.063

Zheng Q, Wang X, Khan M, Zhang W, Gupta B, Guo W (2017) A lightweight authentication encryption based on chaotic SCML for railway cloud service. IEEE Access 6:711–722