Peer-to-Peer Networking and Applications (2020) 13:2356–2366
https://doi.org/10.1007/s12083-019-00869-2

Adaptive privacy-preserving federated learning

Xiaoyuan Liu[1,2] · Hongwei Li[1,2] · Guowen Xu[1,3] · Rongxing Lu[4] · Miao He[5]

Received: 28 August 2019 / Accepted: 27 December 2019 / Published online: 8 May 2020

Abstract
As an emerging training model, federated deep learning has been widely applied in many fields such as speech recognition, image classification and classification of peer-to-peer (P2P) Internet traffics. However, it also entails various security and privacy concerns. In the past years, many researchers have been carried out toward elaborating solutions to alleviate the above challenges via three underlying technologies, i.e., Secure Multi-Party Computation (SMC), Homomorphic Encryption (HE) and Differential Privacy (DP). Compared with SMC and HE, differential privacy is outstanding in terms of efficiency. However, due to the involvement of noise, DP always needs to make a trade-off between security and accuracy. i.e., achieving a strong security requirement has to sacrifice certain accuracy. To seek the optimal balance, we propose APFL, an Adaptive Privacy-preserving Federated Learning framework in this paper. Specifically, in the APFL, we calculate the contribution of each attribute class to the outputs with a layer-wise relevance propagation algorithm. By injecting adaptive noise to data attributes, our APFL significantly reduces the impact of noise on the final results. Moreover, we introduce the Randomized Privacy-preserving Adjustment Technology to further improve the prediction accuracy of the model. We present a formal security analysis to demonstrate the high privacy level of APFL. Besides, extensive experiments show the superior performance of APFL in terms of accuracy, computation and communication overhead.

Keywords Privacy protection · Differential privacy · Federated learning · Distributed system

1 Introduction

Deep learning has demonstrated superior performance in many fields, such as autonomous driving [12], medical diagnosis [7, 10, 11], and image recognition [21]. However, traditionally centralized deep learning usually trains a network with large amounts of data collected from users, which potentially leads to privacy leakages for users. Recently, federated learning proposed by Google has attracted much attention, as it only requires users to upload the gradients of the local model to the cloud server, instead of users' original data. Federated learning has been used in many scenarios, such as natural language processing [6], classification of peer-to-peer (P2P) Internet traffics [19] and ransomware classification [29].

✉ Hongwei Li
hongweili@uestc.edu.cn

Extended author information available on the last page of the article.

Compared with traditionally centralized deep learning, federated learning mitigates privacy leaks to some extent [16, 25, 26]. However, many studies show that attackers can still compromise users' privacy through gradients [13]. Particularly, Song et al. [20] shown that deep learning technology can "memorize" information about the training data in the model. Under this situation, once an adversary obtains white- or black-box access to the resulting model, training data could be exposed [13, 27, 32, 33].

In order to alleviate the above privacy problem, many researchers have put their efforts to come up with solutions via the following three types of technologies [5, 8, 9, 22–24, 31, 34] : Secure Multi-party Computation (SMC), Homomorphic Encryption (HE), and Differential Privacy (DP). SMC focuses on how to safely calculate an appointed function without a trusted third party. Meanwhile, each participant is required to obtain nothing about other entities except for the sum of gradients. For example, by exploiting SMC, Bonawitz et al. [28] designed a privacy-preserving federated learning framework, which can securely aggregate the users' gradients, and be robust to users' dropping out. However, their model leads to huge communication overhead due to multiple interactions involving in the learning process. HE allows the third party to perform algebraic operations over the encrypted domain without decryption. For example, Phong et al. [2] proposed a privacy-preserving scheme based on HE for federated learning, which utilizes additively homomorphic encryption to protect the gradients against the curious server. However, once users who hold the same secret key collude with each other, their scheme will fail to protect the users' privacy. DP is a strong defense tool against inference attacks, which, compared with SMC and HE, is outstanding in efficiency [14, 17]. Unfortunately, DP always needs to make a trade-off between security and accuracy. For example, Shokri et al. [18] proposed a method by injecting noise into gradients of model at every training step for protecting privacy. However, injecting noise with a constant privacy budget will dramatically degrade the accuracy of prediction.

Aiming at the above challenges, we propose APFL, an Adaptive Privacy-Preserving Federated Learning framework with differential privacy. To find the optimal balance between security and accuracy, we improve the layer-wise relevance propagation algorithm and design a Randomized Privacy-preserving Adjustment Technology. Specifically, our contributions summarize into the following three aspects:

- We first improve the layer-wise relevance propagation algorithm to calculate the contribution of each data attribute to the model outputs. Then, we develop an adaptive scheme of injecting noise with different privacy budget according to the contribution. Compared with the traditional methods of injecting noise, we maximize the accuracy of the model under the same degree of privacy protection.
- To further improve the accuracy of APFL, we design a Randomized Privacy-preserving Adjustment Technology (RPAT), by which users can personalize parameters to filter superfluous noise. Furthermore, we claim that RPAT satisfies differential privacy.
- We theoretically prove the privacy of APFL. Besides, our experiments demonstrate the high accuracy and high efficiency simultaneously in our model, compared with existing frameworks. Especially, our work still maintains a high accuracy of 88.46% even under a strong privacy guarantee ($\epsilon = 0.1$).

The remainder of this paper is organized as follows. In Section 2, we outline the system model, threat model and design goal. In Section 3, we describe the pre-requisites of the proposed schemes. Then, we present Adaptive Privacy-preserving Federated Learning (APFL) in detail in Section 4 and carry out the security analysis in Section 5. Finally, Section 7 comes to a conclusion about this paper.

2 System model, threat model and design goal

2.1 System model

As Fig. 1 shows, there are two parties, namely cloud server and users in our system model.

- *Cloud server*: The cloud server negotiates a network framework with users in advance. Then, the server trains an initial model over public data, then broadcasts the parameters of the initial model to the users. After the users train respective models locally, the cloud server collects the model gradients sent by users, and updates the global model.
- *Users*: Users download the model parameters initialized by the cloud server. Then, each user trains the private model over the local data set. Finally, users send the perturbed gradients of the local model to the cloud server.

2.2 Threat model and design goal

We consider the cloud server to be an "honest-but-curious" entity. i.e., the server will follow the agreement with all users. However, by exploiting the convenience of full access to users' gradients, it also attempts to obtain additional information in the training process. For this reason, the goal of our APFL is to protect the local gradients sent to the server from being inferred any extra information about users.

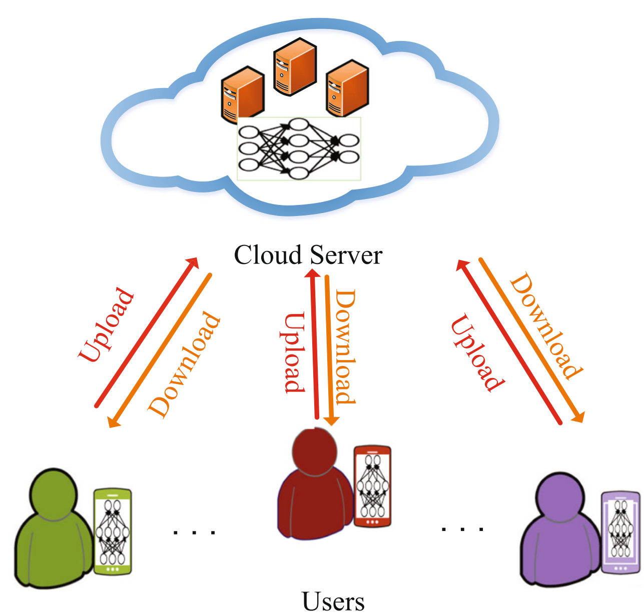


Fig. 1 System model

3 Preliminaries

In this section, we review the concept of federated learning, differential privacy and layer-wise relevance propagation algorithm, which serve as the underlying structure of our APFL.

3.1 Federated learning

Traditionally centralized deep learning requires training data to be put together to a data center. The model is trained in a centralized manner. While federated learning allows data owners to hold a private learning network, which trains with local data set. After that, each participant uploads the gradients of the local model to the cloud server. By updating with the global gradients gathered at the cloud server, the local model can be avoided being over-fitting. Besides, it also protects local data from being directly known to other participants or the cloud server.

Each user U_i owns a database D_i which contains n data items (X_i, Y_i), where $i \in [1, n]$. Each data item includes u attributes and v labels. i.e., $x_{i,1}, x_{i,2}, ..., x_{i,u}, y_{i,1}, y_{i,2}, ..., y_{i,v}$. We run the mini-batch gradient descent algorithm to optimize the learning model. User trains local model in each iteration with a random subset of data $D_i^t \in D_i$ ($|D_i^t| = t$), where t means the value of batch. The loss function $L(Y_i, f(X_i, \omega_i^r))$ is used to estimate the degree of inconsistency between the predicted value $f(X_i, \omega_i^r)$ of model and the true label Y_i after iteration r, where ω_i^r represent weight matrix or system parameters of user U_i after iteration r. The gradients by deriving the loss function can be rewritten as follows:

$$\nabla g(D_i^t, \omega_i^r) = \frac{\partial L(Y_i, f(X_i, \omega_i^r))}{\partial \omega_i^r} \tag{1}$$

The updating step for local model can be rewritten as:

$$\omega_i^{r+1} \leftarrow \omega_i^r - \eta_i \nabla g(D_i^t, \omega_i^r) \tag{2}$$

where η_i is the learning rate of user U_i.

For updating the model of the cloud server, the server collects the gradients of a random subset of users $U^s \in U$ ($|U^s| = s$). After each user shares vector $\{| D_i^t |\} \| \varsigma_{U_i}$ with the cloud server, where $\varsigma_{U_i} = | D_i^t | \nabla g(D_i^t, \omega_i)$. The cloud server calculates the weighted average and performs a gradient descent step:

$$\omega^{r+1} \leftarrow \omega^r - \eta \frac{\sum_{U_i \in U^t} \varsigma_{U_i}}{\sum_{U_i \in U^t} | D_i^t |} \tag{3}$$

where η is the learning rate of the cloud server, and ω^r are the system parameters of the cloud server after iteration r.

3.2 Differential privacy

The definition of differential privacy is as follow:

Definition 3.1 ϵ-Differential Privacy: An algorithm A satisfies ϵ-differential privacy, where $\epsilon \geq 0$. If databases D and D' that differ in only one tuple, we have:

$\forall T \subseteq \mathbf{R}(A) : \mathbf{P}[A(D) \in T] \leq e^{\epsilon}\mathbf{P}[A(D') \in T]$

where $\mathbf{R}(A)$ represent all possible outputs of the algorithm A. ϵ is privacy budget, which decides the privacy level. i.e., the smaller ϵ, the stronger privacy guarantee.

Theorem 3.1 ***Sequential Composition:*** *Given A_1, A_2 satisfying ϵ_1-differential privacy, ϵ_2-differential privacy respectively, we have: $A_2(A_1(D), D)$ satisfies $(\epsilon_1 + \epsilon_2)$-differential privacy.*

There are many ways [3] *to implement differential privacy, one of the most popular methods is the Laplace mechanism.*

Definition 3.2 Laplace Mechanism: Given a function $f(D)$ over database D, $\ddot{f}(D) = f(D) + Lap(\frac{GS}{\epsilon})$ satisfies ϵ-differential privacy. Where $Lap(\frac{GS}{\epsilon})$ is sampled from Laplace distribution, sensitivity GS reflects the maximum range that varies over two neighboring databases: $GS = \max \|f(D) - f(D')\|_1$, where $\|f(D) - f(D')\|_1$ means the manhattan distance between $f(D)$ and $f(D')$.

4 Our proposed scheme

We present the details of our APFL in the following phases. Before starting training, users download the system parameters of the initial model from the cloud server. Each user normalizes each data attribute value $\sqrt{\sum_{j=1}^{v} x_{i,j}^2} \leq 1$ in the local database D_i, which makes the training faster convergence. Each user trains local model, the steps are as follows:

4.1 Layer-wise relevance propagation algorithm

In this paper, we decompose the outputs into every layer with the Layer-wise Relevance Propagation (LRP) algorithm. We place more details about the LRP algorithm in the following part.

Each user locally performs the training feed-forward operation with original data, which can obtain the output of local model. According to the linear relationship between adjacent layers, the contribution $C_{a_i}^{l_k}(x_i)$ of the neuron a_i in the k-th layer equals to the sum of the contributions of the adjacent layers connected to neuron a_i:

$$C_{a_i}^{l_k}(x_i) = \sum_{a_j \in l_{k+1}} C_{a_i \leftarrow a_j}^{l_k \leftarrow l_{k+1}}(x_i) \tag{4}$$

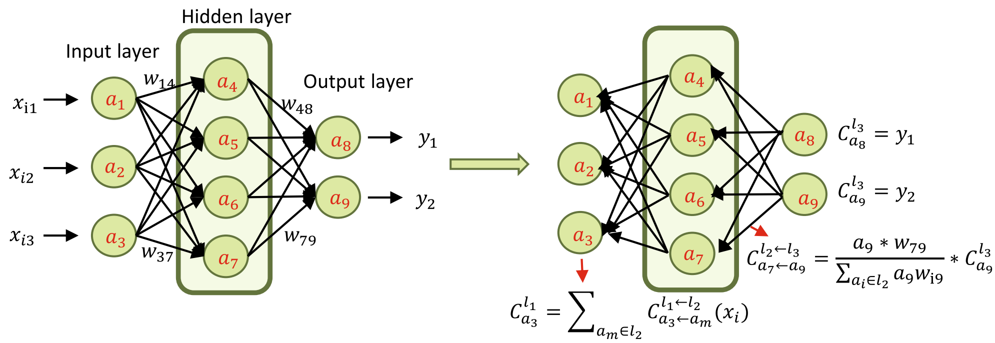


Fig. 2 Layer-wise relevance propagation

For example, as Fig. 2 shows, we have:

$$C_{a_7}^{l_2}(x_i) = \sum_{a_j \in l_3} C_{a_7 \leftarrow a_j}^{l_2 \leftarrow l_3}(x_i) = C_{a_7 \leftarrow a_8}^{l_2 \leftarrow l_3}(x_i) + C_{a_7 \leftarrow a_9}^{l_2 \leftarrow l_3}(x_i) \quad (5)$$

where "$\leftarrow$" means the connection relations between two parts. Specifically, "$l_2 \leftarrow l_3$" is the connection relations of the adjacent layers between the $2\text{-}th$ layer and the $3\text{-}th$ layer in Deep Neural Networks (DNNs).

When the $k\text{-}th$ layer is output layer, we have:

$$C_{a_i}^{l_k}(x_i) = f(x_i, \omega_i^r) \quad (6)$$

That is, the contribution $C_{a_i}^{l_o}(x_i)$ of the neuron a_i in output layer is equal to the output of model.

The contribution $C_{a_i \leftarrow a_j}^{l_{k-1} \leftarrow l_k}(x_i)$ from the neurons a_j in the $k\text{-}th$ layer to the neurons a_i in the $k-1\text{-}th$ layer is as follows:

$$C_{a_i \leftarrow a_j}^{l_{k-1} \leftarrow l_k}(x_i) = \begin{cases} \frac{a_i w_{i,j}}{\sum_{a_i \in l_{k-1}} a_i w_{i,j}} C_{a_j}^{l_k}(x_i) & \sum_{a_i \in l_{k-1}} a_i w_{i,j} \neq 0 \\ \mu & \sum_{a_i \in l_{k-1}} a_i w_{i,j} = 0 \end{cases} \quad (7)$$

where μ is a number that is infinitely close to zero, but greater than zero. From the above formulas, we can hold that the contribution of each layer is equal, and the contributions are transmitted layer by layer:

$$\begin{aligned} \sum f(x_i, \omega_i^r) &= C_{a_8}^{l_3}(x_i) + C_{a_9}^{l_3}(x_i) \\ &= C_{a_4}^{l_2}(x_i) + C_{a_5}^{l_2}(x_i) + C_{a_6}^{l_2}(x_i) + C_{a_7}^{l_2}(x_i) \\ &= C_{a_1}^{l_1}(x_i) + C_{a_2}^{l_1}(x_i) + C_{a_3}^{l_1}(x_i) \end{aligned} \quad (8)$$

where $\sum f(x_i, \omega_i^r)$ represents the sum of model outputs.

So far, we can get all the contributions of neurons and the contributions of layers based on formulas above.

4.2 Perturb contribution

By extracting the contribution of the same attribute from data tuple, we can calculate the average contribution of every attribute class to the output:

$$C_j(x_i) = \frac{1}{n} \sum_{i=1}^{n} C_{x_{i,j}}(x_i), \; j \in [1, u] \quad (9)$$

Due to users calculate the contribution with original data, we inject noise into the contribution of the attribute class to protect original data:

$$\ddot{C}_j(x_i) = C_j(x_i) + Lap(\frac{GS_c}{\epsilon_c}), \; j \in [1, u] \quad (10)$$

where sensitivity $GS_c = \frac{2u}{|D|}$, and u, $|D|$ represent the max number of attributes and tuples, respectively.

4.3 Randomized privacy-preserving adjustment technology

The following shows the transformation process of each hidden neuron in the learning model:

$$y = a(\mathbf{x} * \omega + b)$$

where $\mathbf{x}$ represents input vector, y is output, b and ω represent bias term and weight matrix, respectively. $a()$ is an activation function used to combine linear transformation with nonlinear transformation. $z(\omega) = \mathbf{x} * \omega + b$ is the linear transformation part.

Due to the structure of the neural network, the output of the upper layer is the input of the next layer, from which we can obtain that the original data is only used by the linear transformation in the first hidden layer. Intuitively, to obtain a learning model with privacy protection, we can inject noise into the data in the first layer of the hidden layer. As Phan et al. [15] mentioned, there is a conventional approach for linear transformation to inject noise with the same privacy budget into the original data, and the enhanced version is to inject noise with different privacy budget. But our work is more competitive.

We creatively propose a Randomized Privacy-preserving Adjustment Technology (RPAT), which can improve the accuracy and availability of the system. In particular, we introduce two adjustment factors: f and p ($f \in [0, 1]$, $p \in [0, 1]$) [4], where f represents a threshold to decide whether the contribution of the attribute to the output is high or low, whose value is defined by users. i.e., the attribute classes, whose contributions which

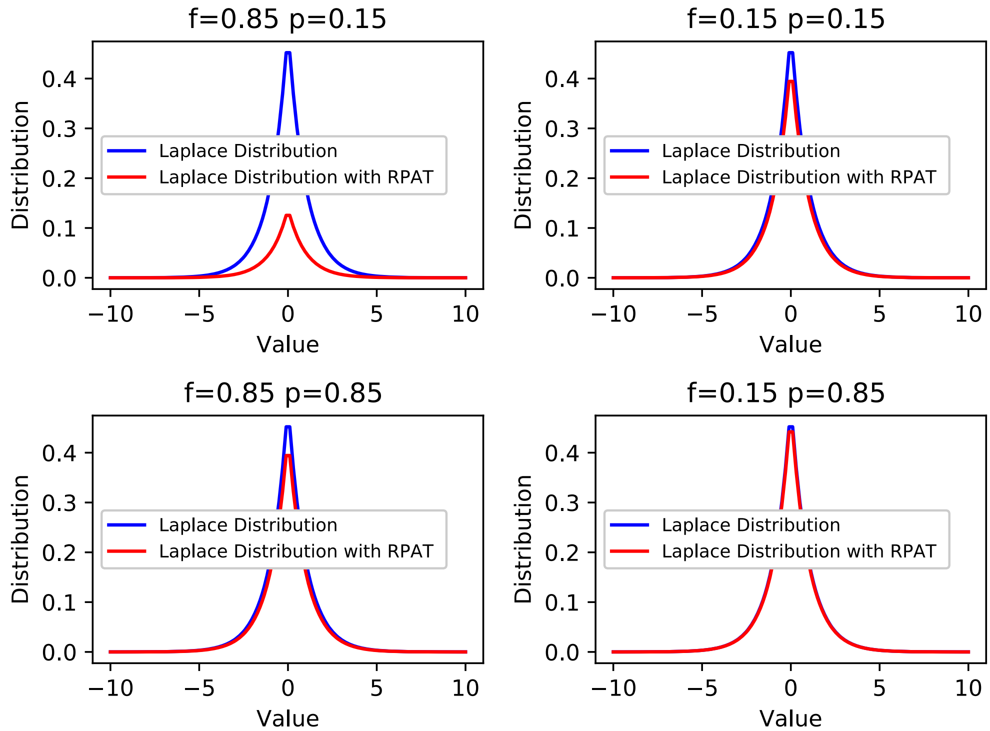


Fig. 3 Laplace distribution

exceeds threshold f, have a greater contribution to output. Then, we inject adaptive Laplace noise to all these attributes. While the contribution is lower than threshold f, a probability selection is made for such attributes. i.e., we choose the original data with probability $1-p$, and to inject adaptive Laplace noise to some attributes with probability p. The formula is as follows:

$$\tilde{x}_{i,j} = \begin{cases} \ddot{x}_{i,j} & \beta \geq f \\ \bar{x}_{i,j} & \beta < f \end{cases} \tag{11}$$

where β represents the ratio of contribution: $\beta = \frac{|\ddot{C}_j|}{\sum_{j=1}^{u}|\ddot{C}_j|}$. When $\beta < f$, we have:

$$\bar{x}_{i,j} = \begin{cases} \ddot{x}_{i,j} & with\ probability\ p \\ x_{i,j} & with\ probability\ 1-p \end{cases} \tag{12}$$

f and p are hyper-parameters, which can be adjusted by users according to their own situation.

The privacy budget ratio ϵ_j for each attribute class by: $\epsilon_j = \frac{u*|\ddot{C}_j|}{\sum_{j=1}^{u}|\ddot{C}_j|} * \epsilon_l$. That is, the privacy budget ϵ_l is proportionally distributed to each attribute class based on the contribution. The adaptive noise is injected into the attributes as follow:

$$x'_{i,j} = x_{i,j} + \frac{1}{|D_i^t|} Lap(\frac{GS_l}{\epsilon_j}) \tag{13}$$

Without loss of generality, the value of adjustment factors f and p are related to the accuracy and privacy level of the system. i.e., the smaller f and the greater p, the higher privacy level but lower accuracy, and vice versa. The impact of RPAT on the Laplace distribution is shown in Fig. 3. When the value of f is 0.15 and p is setting to 0.85, the noise distribution using the RPAT substantially coincides with the Laplace mechanism. We can draw a convincing conclusion that the privacy budget of RPAT is close to the original Laplace mechanism at the same noise level. Our technology is more privacy-protected with shrinking the range of adjustment factors.

4.4 Perturb objection function

To protect the labels in the original data tuple, we expand the loss function into a polynomial by Taylor Expansion, and inject Laplace noise into the coefficients of the polynomial. For more details, please refer to [30].

5 Security analysis

As discussed in Section 4, the primary privacy issues in the APFL are the confidentiality of original data and system parameters. In this section, we focus on analyzing that every operation satisfies differential privacy.

5.1 Perturbing the contribution

Lemma 5.1 *Assuming that there are two neighboring databases D and D', which only differ in last tuple x_n and*

x'_n. $C(D)$ and $C(D')$ are the contribution of all attributes to output, respectively.

$$C(D) = \{C_j(x_i)\},\ j \in [1,u],\ where\ C_j(x_i) = \frac{1}{n}\sum_{i=1}^{n} C_{x_{i,j}}(x_i),\ j \in [1,u], x_i \in D$$
$$C(D') = \{C_j(x'_i)\},\ j \in [1,u],\ where\ C_j(x'_i) = \frac{1}{n}\sum_{i=1}^{n} C_{x'_{i,j}}(x'_i),\ j \in [1,u], x'_i \in D' \tag{14}$$

The perturbation for the contribution can be written as:

$$\ddot{C}_j(x_i) = C_j(x_i) + Lap(\frac{GS_c}{\epsilon_c}),\ j \in [1,u] \tag{15}$$

which satisfies ϵ_c-differential privacy.

Proof The sensitivity GS_c of the contribution is as follows:

$$\begin{aligned} GS_c &= \frac{1}{|D|}\sum_{j=1}^{u} \| \sum_{x_i \in D} C_{x_{i,j}}(x_i) - \sum_{x'_i \in D'} C_{x'_{i,j}}(x'_i)\|_1 \\ &= \frac{1}{|D|}\sum_{j=1}^{u} \|C_{x_{n,j}}(x_n) - C_{x'_{n,j}}(x'_n)\|_1 \\ &\le \frac{2}{|D|}\max\sum_{j=1}^{u}\|C_{x_{i,j}}(x_i)\|_1 \\ &\le \frac{2u}{|D|} \end{aligned}$$

where u, $|D|$ represent the max number of attributes and tuples, respectively. Then, we have:

$$\begin{aligned} \frac{\Pr(\ddot{C}(D))}{\Pr(\ddot{C}(D'))} &= \frac{\prod_{j=1}^{u}\exp(\frac{\epsilon_c\|\frac{1}{|D|}\sum_{x_i\in D} C_j(x_i) - \ddot{C}_j(x_i)\|_1}{GS_c})}{\prod_{j=1}^{u}\exp(\frac{\epsilon_c\|\frac{1}{|D'|}\sum_{x'_i\in D'} C_j(x'_i) - \ddot{C}_j(x'_i)\|_1}{GS_c})} \\ &= \prod_{j=1}^{u}\exp(\frac{\epsilon_c}{|D|GS_c}\|C_j(x_n) - C_j(x'_n)\|_1) \\ &\le \prod_{j=1}^{u}\exp(\frac{\epsilon_c}{|D|GS_c}\max\|C_j(x_n)\|_1) \\ &= \exp(\epsilon_c\frac{\max_{x_i\in D}\sum_{j=1}^{u}\|C_j(x_n)\|_1}{|D|GS_c}) \\ &\le \exp(\epsilon_c) \end{aligned}$$

Consequently, the operation satisfies ϵ_c-differential privacy. □

5.2 Randomized privacy-preserving adjustment technology

The Randomized Privacy-preserving Adjustment Technology (RPAT) perturb the linear transformation function discussed in Section 4.3, which satisfies $(\epsilon_c + \epsilon_l)$-differential privacy. The proof is as follows.

Lemma 5.2 *Assuming that two neighboring batches D_i^t and $D_i^{t'}$, which differ in last tuple x_n and x'_n. $z(D_i^t)$ and $z(D_i^{t'})$ are the linear transformation functions, respectively. The RPAT satisfies $(\epsilon_c + \epsilon_l)$-differential privacy.*

Proof In general, we consider the bias term as the first type of data attribute. i.e., $x_{i,0} = b_i$. The linear transformation can be rewritten as: $\ddot{\mathbf{z}}_{x\in D_i^t}(\omega) = \ddot{\mathbf{x}} * \omega$. The sensitivity GS_l of the linear transformation is as follows:

$$\begin{aligned} GS_l &= \sum_{a_i\in l_1}\sum_{j=1}^{u}\|\sum_{x_i\in D_i^t} x_{i,j} - \sum_{x'_i\in D_i^{t'}} x'_{i,j}\|_1 \\ &= \sum_{a_i\in l_1}\sum_{j=1}^{u}\|x_{n,j} - x'_{n,j}\|_1 \\ &\le \sum_{a_i\in l_1}\sum_{j=1}^{u}\max_{x_i\in D_i^t}\|x_{n,j}\|_1 \\ &\le \sum_{a_i\in l_1} u \end{aligned}$$

where $a_i \in l_1$ means the neuron a_i in the first hidden layer l_1, u is the number of attributes in data tuple $x_i \in D_i^t$.

We design the RPAT, which includes two adjustment factors: f and p, which can filter superfluous noise. The general expression of the attribute after the RPAT is as follows:

$$\begin{aligned} \tilde{x}_{i,j} &= [(1-f) + f*p]*\ddot{x}_{i,j} + f*(1-p)*x_{i,j} \\ &= [(1-f) + f*p][x_{i,j} + Lap(\frac{GS_l}{\epsilon_j})] + [f*(1-p)]x_{i,j} \\ &= x_{i,j} + [(1-f) + f*p][Lap(\frac{GS_l}{\epsilon_j})] \end{aligned} \tag{16}$$

Then, we can obtain:

$$\begin{aligned} \frac{\Pr(\ddot{\mathbf{z}}_{D_i^t}(\omega))}{\Pr(\ddot{\mathbf{z}}_{D_i^{t'}}(\omega))} &= \frac{\prod_{a_i\in l_1}\prod_{j=1}^{u}\exp(\frac{\epsilon_j\|\sum_{x_i\in D_i^t} x_{i,j} - \sum_{x_i\in D_i^t}\tilde{x}_{i,j}\|_1}{GS_l})}{\prod_{a_i\in l_1}\prod_{j=1}^{u}\exp(\frac{\epsilon_j\|\sum_{x'_i\in D_i^{t'}} x'_{i,j} - \sum_{x'_i\in D_i^{t'}}\tilde{x}'_{i,j}\|_1}{GS_l})} \\ &\le \prod_{a_i\in l_1}\prod_{j=0}^{u}\exp(\frac{\epsilon_j}{GS_l}\|\sum_{x_i\in D_i^t} x_{i,j} - \sum_{x'_i\in D_i^{t'}} x'_{i,j}\|_1) \\ &\le \prod_{a_i\in l_1}\prod_{j=0}^{u}\exp(\frac{\epsilon_j}{GS_l}\max_{x_i\in D_i^t}\|x_{n,j}\|_1) \\ &\le \exp(\epsilon_l\frac{\sum_{a_i\in l_1} u[\sum_{j=1}^{u}\frac{|\ddot{C}_j|}{\sum_{j=1}^{u}|\ddot{C}_j|}]}{GS_l}) \\ &= \exp(\epsilon_l) \end{aligned}$$

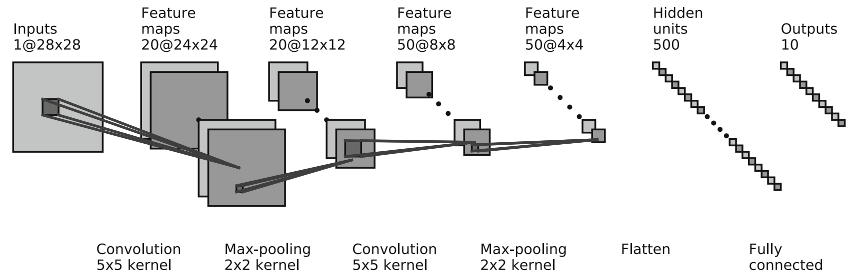


Fig. 4 Neural network architecture

According to the sequential composition of differential privacy, the linear transformation with RPAT satisfies $(\epsilon_c + \epsilon_l)$-differential privacy. □

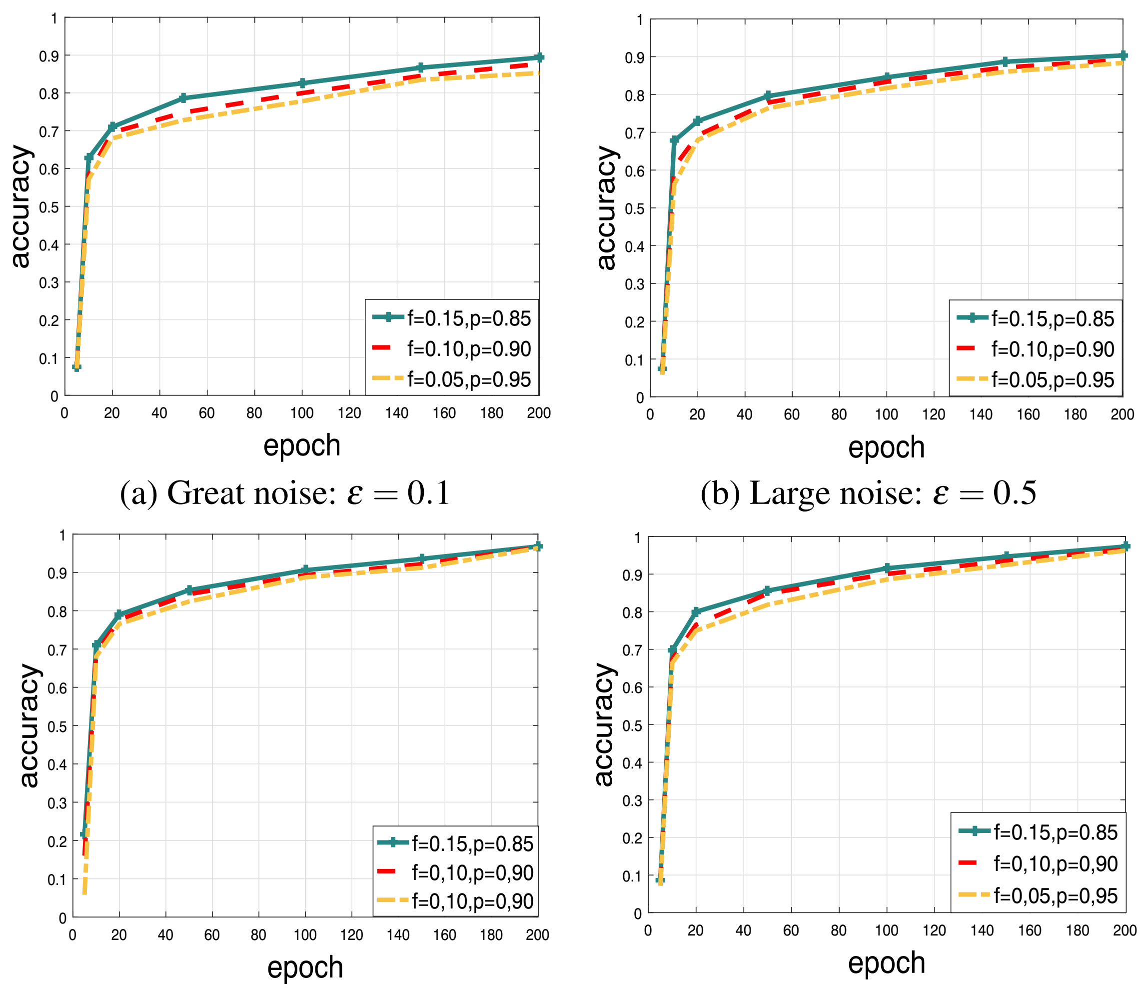
5.3 The loss function

Zhang et al. [30] have proved that the operation to the loss function satisfies ϵ_f-differential privacy. According to the sequential composition, the operation satisfies $(\epsilon_c+\epsilon_l+\epsilon_f)$-differential privacy in our APFL. Since the calculation of system parameters ω^* does not require more original data, system parameters ω^* also satisfy $(\epsilon_c+\epsilon_l+\epsilon_f)$-differential privacy.

6 Performance evaluation

6.1 Dataset and neural network architectures

We evaluate our APFL based on the MNIST database, which is a classic entry-level demo for deep learning. It



(a) Great noise: $\varepsilon = 0.1$

(b) Large noise: $\varepsilon = 0.5$

(c) Moderate noise: $\varepsilon = 2.0$

(d) Small noise: $\varepsilon = 8.0$

Fig. 5 Accuracy of the APFL

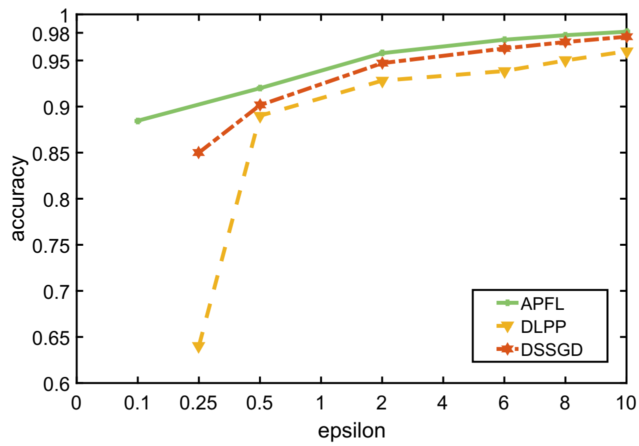


Fig. 6 Accuracy of different privacy budget

consists of 60,000 training pictures and 10,000 test pictures. Each image in the MNIST consists of 28x28 pixels.

We use Tensorflow which is a popular library for deep learning. The experiments run in Lenovo server which is Ubuntu 18.04 system and has Intel(R) Xeon(R) E5-2620 2.10 GHz CPU and 16GB RAM.

Our neural network architecture is shown as Fig. 4.

6.2 Accuracy evaluation

We implement RPAT for improving the performance of APFL. Then, we further limit the adjustment factors to a smaller range to ensure the privacy level of APFL. i.e., $0 \leq f \leq 0.15$ and $0.85 \leq p \leq 1$.

We compare the accuracy of the APFL with different privacy budget ($\epsilon_1 = 0.1$, $\epsilon_2 = 0.5$, $\epsilon_3 = 2.0$, $\epsilon_4 = 8.0$). The smaller privacy budget ϵ, the greater noise. We also choose three different adjustment factors for each privacy budget ((a): $f = 0.15$, $p = 0.85$, (b): $f = 0.10$, $p = 0.90$, (c): $f = 0.05$, $p = 0.95$). It is certain that the setting of ($f = 0.15$, $p = 0.85$) can guarantee the privacy level of the system. In addition, it is worth noting that the value of privacy budget ϵ in the experiment is the sum of ϵ_c, ϵ_l and ϵ_f. We evenly divide ϵ into the following three steps: the calculation of contribution, the linear transformation and the calculation of loss function. ie,. $\epsilon_c = \epsilon_l = \epsilon_f = \frac{\epsilon}{3}$.

As shown in Fig. 5, with the privacy budget ϵ increasing, the accuracy of our system maintains a steady growth trend. With the range of adjustment factors continue to shrink, the accuracy of the APFL is gradually reducing, but still remains high level. For instance, when the privacy budget ϵ is setting to 8.0, the accuracy of APFL is as high as 97.34% in the setting of $f = 0.15$ and $p = 0.85$, while 96.57% in the setting of $f = 0.10$ and $p = 0.90$, as well as 96.25% in the setting of $f = 0.05$ and $p = 0.95$.

We also compare with the works using DP mechanism to protect privacy of deep learning model in recent years, such as DLPP in [1] and DSSGD in [18]. In Fig. 6, we can clearly get a message that our work performs well even under a strong privacy guarantee ($\epsilon = 0.1$). When the adjustment factors are setting to $f = 0.15$ and $p = 0.85$, the accuracy of model reaches 88.46% after 200 epochs. In addition, the adjustment factors are taken as $f = 0.05$ and $p = 0.95$, the accuracy of APFL is 86.79%. However, the accuracy of DSSGD only reaches 79.63% under the same privacy budget, and the accuracy of the DLPP model is less than 65.00%.

6.3 Efficiency evaluation

The additional overhead of our system comes mainly from the pre-training process on the server-side, and users-side calculating and perturbing the contributions before starting training. We use 20 epochs to train an initialized network for the cloud server, which takes an average of 68.22 seconds.

Before the independent and asynchronous training process, the user needs to calculate the contribution with the layer-wise relevance propagation algorithm. This process only needs the forward-propagation process in the training, without calculating the gradients and loss penalty in the back-propagation process. Its average time is 4.35 milliseconds.

To mitigate privacy threats, our solution is to inject Laplace noise to the contributions, the original data in the linear transformation function, and the coefficients of the loss function. The step of injecting noise into the contributions can be synchronized with calculating the contributions, which need extra 2.67 milliseconds. The operations of injecting adaptive noise to original data in the linear transformation and the coefficients of the loss function can be completed before training, the computations of which for every epoch is similar to perturbing the contributions. In short, our APFL is outstanding in terms of efficiency.

7 Conclusion

In this paper, we propose an Adaptive Privacy-preserving Federated Learning (APFL) framework with differential privacy. In order to achieve the best trade-off between accuracy and privacy, we exploit the layer-wise relevance propagation algorithm to calculate the contributions of the attributes to model outputs. Moreover, we creatively propose the Randomized Privacy-preserving Adjustment Technology which can further improve the accuracy of APFL. The experiments present the superior performance of the APFL in terms of accuracy, computation and communication overhead.

Acknowledgements This work is supported by the National Key R&D Program of China under Grants 2017YFB0802300 and 2017YFB0802000, the National Natural Science Foundation of China under Grants 61802051, 61772121, 61728102, and 61472065, the Peng Cheng Laboratory Project of Guangdong Province PCL2018KP004, the Guangxi Key Laboratory of Cryptography and Information Security under Grant GCIS201804.

References

1. Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, Zhang L (2016) Deep learning with differential privacy. In: Proceedings of ACM CCS, pp 308–318
2. Aono Y, Hayashi T, Wang L, Moriai S et al (2018) Privacy-preserving deep learning via additively homomorphic encryption. IEEE Trans Inform Forensics Secur 13(5):1333–1345
3. Dwork C, Rothblum GN (2016) Concentrated differential privacy. arXiv:1603.01887
4. Erlingsson Ú, Pihur V, Korolova A (2014) Rappor: randomized aggregatable privacy-preserving ordinal response. In: Proceedings of ACM CCS, pp 1054–1067
5. Hao M, Li H, Luo X, Xu G, Yang H, Liu S (2019) Efficient and privacy-enhanced federated learning for industrial artificial intelligence. IEEE Trans Indust Inform
6. Hard A, Rao K, Mathews R, Beaufays F, Augenstein S, Eichner H, Kiddon C, Ramage D (2018) Federated learning for mobile keyboard prediction. arXiv:1811.03604
7. Jiang Q, Ma J, Yang C, Ma X, Shen J, Chaudhry SA (2017) Efficient end-to-end authentication protocol for wearable health monitoring systems. Comput Electric Eng 63:182–195
8. Jiang W, Li H, Xu G, Wen M, Dong G, Lin X (2019) Ptas: privacy-preserving thin-client authentication scheme in blockchain-based pki. Future Gen Comput Sys 96:185–195
9. Li H, Liu D, Dai Y, Luan TH, Yu S (2018) Personalized search over encrypted data with efficient and secure updates in mobile clouds. IEEE Trans Emerg Topics Comput 6(1):97–109
10. Li H, Yang Y, Dai Y, Yu S, Xiang Y (2017) Achieving secure and efficient dynamic searchable symmetric encryption over medical cloud data. IEEE Trans Cloud Comput. https://doi.org/10.1109/TCC.2017.2769645
11. Liu X, Zhu H, Lu R, Li H (2018) Efficient privacy-preserving online medical primary diagnosis scheme on naive bayesian classification. Peer-to-Peer Netw Appl 11(2):334–347
12. Maqueda AI, Loquercio A, Gallego G, García N, Scaramuzza D (2018) Event-based vision meets deep learning on steering prediction for self-driving cars. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5419–5427
13. Nasr M, Shokri R, Houmansadr A (2018) Comprehensive privacy analysis of deep learning: stand-alone and federated learning under passive and active white-box inference attacks. arXiv:1812.00910
14. Papernot N, Song S, Mironov I, Raghunathan A, Talwar K, Erlingsson Ú (2018) Scalable private learning with pate. arXiv:1802.08908
15. Phan N, Wu X, Hu H, Dou D (2017) Adaptive laplace mechanism: differential privacy preservation in deep learning. In: Proceedings of IEEE ICDM, pp 385–394
16. Ren H, Li H, Dai Y, Yang K, Lin X (2018) Querying in internet of things with privacy preserving: challenges, solutions and opportunities. IEEE Netw 32(6):144–151
17. Ren H, Li H, Liang X, He S, Dai Y, Zhao L (2016) Privacy-enhanced and multifunctional health data aggregation under differential privacy guarantees. Sensors 16(9):1463
18. Shokri R, Shmatikov V (2015) Privacy-preserving deep learning. In: Proceedings of ACM CCS, pp 1310–1321
19. Sivaprasad A, Ghawalkar N, Hodge S, Sanghavi M, Shinde V (2018) Machine learning based traffic classification using statistical analysis. Int J Recent Innov Trends Comput Commun 6(3):187–191
20. Song C, Ristenpart T, Shmatikov V (2017) Machine learning models that remember too much. In: Proceedings of ACM CCS, pp 587–601
21. de Vos BD, Berendsen FF, Viergever MA, Sokooti H, Staring M, Išgum I (2019) A deep learning framework for unsupervised affine and deformable image registration. Medical Image Anal 52:128–143
22. Wang H, Dong X, Cao Z (2019) Secure and efficient encrypted keyword search for multi-user setting in cloud computing. Peer-to-Peer Netw Appl 12(1):32–42
23. Xu G, Li H, Dai Y, Yang K, Lin X (2019) Enabling efficient and geometric range query with access control over encrypted spatial data. IEEE Trans Inform Forensics Secur 14(4):870–885
24. Xu G, Li H, Liu S, Wen M, Lu R (2019) Efficient and privacy-preserving truth discovery in mobile crowd sensing systems. IEEE Trans Vehicular Technol 68(4):3854–3865
25. Xu G, Li H, Liu S, Yang K, Lin X (2020) Verifynet: secure and verifiable federated learning. IEEE Trans Inform Forensics Secur 15(1):911–926
26. Xu G, Li H, Ren H, Yang K, Deng RH (2019) Data security issues in deep learning: attacks, countermeasures and opportunities. IEEE Commun Magazine 57(11):116–122. https://doi.org/10.1109/MCOM.001.1900091
27. Yang Y, Niu X, Li L, Peng H, Ren J, Qi H (2018) General theory of security and a study of hacker!-s behavior in big data era. Peer-to-Peer Netw Appl 11(2):210–219
28. Young T, Hazarika D, Poria S, Cambria E (2018) Recent trends in deep learning based natural language processing. IEEE Comput Intell Magazine 13(3):55–75
29. Zhang H, Xiao X, Mercaldo F, Ni S, Martinelli F, Sangaiah AK (2019) Classification of ransomware families with machine learning based on n-gram of opcodes. Future Gen Comput Sys 90:211–221
30. Zhang J, Zhang Z, Xiao X, Yang Y, Winslett M (2012) Functional mechanism: regression analysis under differential privacy. Proceedings of the VLDB Endowment 5(11):1364–1375
31. Zhang S, Li H, Dai Y, Li J, He M, Lu R (2018) Verifiable outsourcing computation for matrix multiplication with improved efficiency and applicability. IEEE Internet of Things Journal 5(6):5076–5088
32. Zhang X, Zhao J, Xu C, Li H, Wang H, Zhang Y (2019) Cipppa: conditional identity privacy-preserving public auditing for cloud-based wbans against malicious auditors. IEEE Trans Cloud Comput: 1–1. https://doi.org/10.1109/ TCC.2019.2927219
33. Zhang Y, Xu C, Ni J, Li H, Shen X (2019) Blockchain-assisted public-key encryption with keyword search against keyword guessing attacks for cloud storage. IEEE Trans Cloud Comput: 1–1. https://doi.org/10.1109/TCC.2019.2923222
34. Zhao C, Zhao S, Zhao M, Chen Z, Gao CZ, Li H, Tan YA (2019) Secure multi-party computation: theory, practice and applications. Inform Sci 476:357–372

Xiaoyuan Liu (S'18) received his B.S. degree from the school of control and computer enjineering, North China Electric Power University (NCEPU) in 2016. Currently, she is a master student at the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), China. Her research interests include cryptography, and the privacy issues in Deep learning.

Hongwei Li (M'12-SM'18) is currently the Head and a Professor at Department of Information Security, School of Computer Science and Engineering, University of Electronic Science and Technology of China. He received the Ph.D. degree from University of Electronic Science and Technology of China in June 2008. He worked as a Postdoctoral Fellow at the University of Waterloo from October 2011 to October 2012 under the supervision of Prof. Sherman Shen. His research interests include network security and applied cryptography. His research is supported by National Science Foundation of China, and Ministry of Science and Technology of China, and Ministry of Industry and Information Technology, and China Unicom. Dr. Li has published more than 80 technical papers. He is the sole author of a book, Enabling Secure and Privacy Preserving Communications in Smart Grids (Springer, 2014). Dr. Li serves as the Associate Editor of IEEE Internet of Things Journal, and Peer-to-Peer Networking and Applications, the Guest Editor of IEEE Network and IEEE Internet of Things Journal. He also serves/served the technical symposium co-chair of ACM TUR-C 2019, IEEE ICCC 2016, IEEE GLOBECOM 2015 and IEEE BigDataService 2015, and many technical program committees for international conferences, such as IEEE INFOCOM, IEEE ICC, IEEE GLOBECOM, IEEE WCNC, IEEE SmartGridComm, BODYNETS and IEEE DASC. He won the Best Paper Award from IEEE MASS 2018 and IEEE HELTHCOM 2015. He is the Senior Member of IEEE, Distinguished Lecturer of IEEE Vehicular Technology Society.

Guowen Xu (S'15) received his B.S. degree in information and computing science from Anhui University of Architecture (AUA) in 2014. Currently, he is a Ph.D. student at the School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC), China. His research interests include cryptography, Searchable encryption, and the privacy issues in Deep learning.

Rongxing Lu (S'09-M'11-SM'15) has been an assistant professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada, since August 2016. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor Generals Gold Medal", when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is presently a senior member of IEEE Communications Society. Dr. Lu currently serves as the Vice-Chair (Publication) of IEEE ComSoc CIS-TC. Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Miao He received the B.E. degree from Zhejiang University, China, in 2011, and the M.S. degree from University of Waterloo, Canada, in 2014. He is currently working as a R&D engineer in Fortinet Technologies (Canada) ULC. His research interests include network security and applied cryptography.

Affiliations

Xiaoyuan Liu[1,2] · **Hongwei Li**[1,2] · **Guowen Xu**[1,3] · **Rongxing Lu**[4] · **Miao He**[5]

Xiaoyuan Liu
xiaoyuan.l@foxmail.com

Guowen Xu
guowen.xu@foxmail.com

Rongxing Lu
rlu1@unb.ca

Miao He
miaohe@fortinet.com

1 The School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

2 Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China

3 CETC Big Data Research Institute Co., Ltd., Guiyang 550022, China

4 The Faculty of Computer Science, University of New Brunswick, Fredericton, NB, E3B 5A3, Canada

5 The Fortinet Technologies (Canada) ULC, Ottawa, ON, Canada