World Wide Web (2020) 23:2529–2545
https://doi.org/10.1007/s11280-020-00780-4

DP-FL: a novel differentially private federated learning framework for the unbalanced data

Xixi Huang[1] · Ye Ding[2] · Zoe L. Jiang[1,3] · Shuhan Qi[1] · Xuan Wang[1] · Qing Liao[1,3]

Received: 30 August 2019 / Revised: 23 December 2019 / Accepted: 2 January 2020 /
Published online: 30 April 2020

Abstract

Security issues of artificial intelligence attract many attention in many research fields and industries, such as face recognition, medical care, and client services. Federated learning is proposed by Google, which can prevent the leakage of data during the AI training because each enterprise only needs to exchange training parameters without data sharing. In this paper, we present a novel differentially private federated learning framework (DP-FL) for unbalanced data. In the cloud server, DP-FL framework considers the unbalanced data of different users to set different privacy budgets. In the user client, we design a novel differential private convolutional neural networks with adaptive gradient descent (DPAGD-CNN) algorithm to update each user's training parameters. Experimental results on several real-world datasets demonstrate that the DF-FL framework can protect data privacy with higher accuracy than existing works.

Keywords Federated learning · Privacy protection · Differential privacy

1 Introduction

In recent years, artificial intelligence (AI) has deeply affected industries and our lives. AI has achieved success in many fields, such as face recognition, health-care and financial fraud identification. These applications involve issues of data security and user privacy. For instance, medical records used to train AI models may reveal sensitive information of patients. Protecting user privacy is the first challenge in the AI field. To a large extent, the success of AI is due to the flourishing development of big data. Training AI models require many data. For instance, Google uses over 0.16 millions of real games to train AlphaGo [21]. However, in real life, it is difficult for enterprises or users to own such a large amount of data. Most of enterprises or users that only have small and low-quality data may want to put

This article belongs to the Topical Collection: *Special Issue on Data Science in Cyberspace 2019*
Guest Editors: Bin Zhou, Feifei Li and Jinjun Chen

✉ Qing Liao
liaoqing@hit.edu.cn

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

their data together to train a better AI model. Joint training AI model is not feasible, while data sharing can cause serious security problems. Recently, the European Union issued the General Data Protection Regulation (GDPR). The purpose of GDPR is to curb the abuse of personal information and protect personal privacy. Inevitably, the problem of "data islands" has arisen. Training efficient and robust AI models with small and low quality data are the second challenge in the AI field.

Federated learning [11] proposed by Google provides a solution to the above two challenges. Federated learning can deal with the issues of data security and data sharing. Enterprises and users need to share parameters to train AI models jointly. Federated learning solves such a problem: a server model has established through the users' model parameters updates under the encryption mechanism or the disturbance mechanism.No data exchange or merge between users. The model performance of the server model is close to the model's which gather data from each user. This collaborative learning approach does not reveal user privacy and is consistent with data security regulation.

Protection techniques and reliable security analysis provide the guarantee for the federated learning models. Common privacy protection technologies are k-anonymity [23], l-diversity [16], t-closeness [14] and differential privacy [3]. K-anonymity, l-diversity, t-closeness can not resist background knowledge attacks and cannot provide security guarantee. Differential privacy is an efficient privacy protection technology with elegant definitions. Differential privacy can resist background knowledge attacks and can adjust the degree of privacy protection based on privacy-preserving needs. Differential privacy can provide the guarantee for privacy protection of the federated learning models.

In this paper, we present a novel differentially private federated learning framework (DP-FL) for unbalanced data. Firstly, our framework mainly addresses the unbalanced data scenario. We consider that the unbalanced data scenario is more common in real life. Figure 1 shows the difference between training federated learning models with the unbalanced data (a) and the balanced data (b). In DP-FL, each user only needs to train local model parameters while performing differential privacy processing and then uploads the parameters to the cloud server for updating. There is no data sharing. Secondly, in DP-FL, we design a novel differentially private convolutional neural networks with adaptive gradient descent (DPAGD-CNN) [10] method for each user parameters updating. We can adaptively add noise (implementing differential privacy) based on the gradient descent direction for better

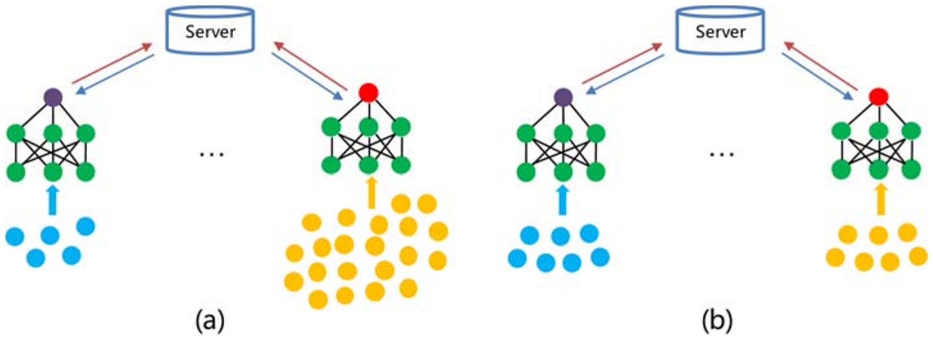

Figure 1 Training federated learning models with the unbalanced data (**a**) and the balanced data (**b**)

model performance. Thirdly, we theoretically explain that our DP-FL framework can protect user data privacy, and through experiments, we demonstrate that our federated learning framework has better model performance than the existing works.

The main contributions of our paper are the following:

1) We present a novel differentially private federated learning (DP-FL) framework for unbalanced data.
2) We design a novel differentially private convolutional neural networks with adaptive gradient descent (DPAGD-CNN) method for each user's model parameters updating.
3) We theoretically and experimentally explain that our DP-FL framework has better model performance while protecting user data privacy.

2 Related works

In this section, we introduce the existing works related to federated learning in recent years. In a traditional centralized deep learning model, the users pass their data that may contain sensitive information to a machine learning company (an untrusted third party). For this data, users can not delete it, and they do not know how these companies use their data, so their data may be at risk of privacy breaches. In 2015, Shokri et al. [20] proposed a multi-participation privacy protection collaborative deep learning model. In this model, each participant can independently train their models locally and then selectively share some of the model parameters of their local models to a central parameter server. In this way, the sensitive data of the participants can be protected from being leaked, and on the other hand, the parameters shared by other participants can be used to improve the accuracy of the model trained by the participants.

Google [11] first proposed the concept of federated learning based on Shokri's work. In federated learning, participants store all training data locally and train the model locally, and then upload the updates to the server in the server. Other participants can download updates to their own devices to improve the accuracy of the local model. In this way, federated learning solves the problem that users can only upload data to the server and cannot train the model locally. The process that users interact with the server is called communication. Konecny et al. [12] proposed a communication-efficient federated learning model. The authors propose two ways: structured updates and sketched updates to reduce the cost of communication.

McMahan et al. [18] proposed the Federated Averaging algorithm on data from many mobile devices. Many federated learning applications are based on this algorithm. Bonawitz et al. [4] propose a practical secure aggregation protocol in a federated learning setting for high dimensional data such as text messages. Zhao [30] proposed a federated learning framework for dealing with non-independent and identically distributed (no-IID) data. The authors propose a strategy to create small subsets of data to be shared among each user, which can improve model accuracy by about 30% in the CIFAR-10 dataset and share only 5% of the data globally.

Recently, researchers focused on the application of federated learning. Yang et al. [28] systematically introduced the concept and application of privacy protection federated learning. Yang proposed that protecting user data privacy is the foremost consideration for federated learning. The AI department of WeBank [27] advocated and proposed the open-source project: Federated AI Technology Enabler (FATE), which provides a secure computing framework for the federated AI ecosystem. Hard et al. [9] used federated learning for keyboard input predictions of mobile devices. The application of federated learning can predict the next word when a user types a word on the virtual keyboard of the mobile device. The application demonstrated the feasibility and benefits of training the language model on the mobile device while users do not be required uploading user data to the server. Federated learning can also be used in the IoT field [24]. Bonawitz et al. [3] proposed a federated learning system design for large-scale devices.

For the privacy protection of federated learning models, there are many security models or privacy protection technologies that can provide reliable privacy guarantees[25, 26], such as secure multiparty computing (MPC)[7, 19], homomorphic encryption[29], and differential privacy[5]. The first two are based on cryptography, and the latter is based on noise perturbation. Considering the wide applicability of differential privacy in deep learning models, differential privacy can also be used well for the privacy protection of federated learning. Geyer et al. [8] proposed a user-level differentially private federated learning framework. The framework provides users with differential privacy protection. The purpose of differential privacy protection is to hide users' contributions during model training and to obtain a trade-off between privacy loss and model performance. Liu et al. [15] proposed a secure federated transfer learning framework. The framework combines federated learning with transfer learning. It provides the same accuracy as the non-privacy protection method while protecting the privacy of user data. Bagdasaryan et al. [2] proved that client-level differential privacy could reduce the effectiveness of the backdoor attack on federated learning.

Our paper uses differential privacy to federated learning for privacy-preserving, and we consider the unbalanced data scenario. We theoretically and experimentally explain that our DP-FL framework has better model performance while protecting user data privacy.

3 Preliminaries

In this section, we introduce differential privacy [5] and some lemmas about differential privacy. Furthermore, we introduce how to apply differential privacy to the AI model. Then, we introduce federated learning and privacy-preserving in federated learning.

3.1 Differential privacy

Differential privacy aims to maximize the accuracy of data queries when querying from a database while minimizing the chances of identifying the records of the database. One condition for differential privacy is that datasets must be two neighbour datasets. If $\left|\left(D \backslash D'\right) \bigcup \left(D' \backslash D\right)\right| = 1$, the two datasets D and D' are neighbour datasets. In other words, D and D' differ by up to one record. Here is the definition of differential privacy.

Definition 3.1 Given a random function K, if the output S $(S \in Range(K))$ of function K on a given neighbor datasets D and D' satisfies the following inequality:

$$\Pr[K(D)] \leq \exp(\epsilon)\Pr[K(D') \in S] + \delta \tag{1}$$

Then the function K satisfies $(\epsilon, \delta)-$differential privacy. In (1), δ is a relaxation factor. If $\delta = 0$, the random function K gives pure differential privacy. If $\delta > 0$, K gives appreciate differential privacy. The former provides a stronger privacy guarantee than the latter. ϵ is used to balance privacy protection and data utility. The smaller ϵ is, the higher the privacy protection and the lower the data utility, and vice-versa. Implementing differential privacy techniques requires the injection of noise, which is closely related to the global sensitivity of the dataset. The global sensitivity is the following.

Definition 3.2 For any query function (function Q maps dataset D to a d-dim real space), the global sensitivity of function Q is the following.

$$\Delta Q = \max_{D,D'} \|Q(D) - Q(D')\|_1 \tag{2}$$

D and D' are neighbour datasets. $\|\cdot\|_1$ represents the $1-$norm. The global sensitivity measures the maximum difference obtained by querying adjacent datasets, such as the maximum query difference value obtained after a record is inserted or deleted into a dataset.

Noise mechanism is the conventional way to achieve differential privacy. The Gaussian mechanism [21] is commonly used for differential privacy protection in deep learning, which generates noise through the Gaussian distribution to perturb the output value generated by the query operation so that the attacker cannot distinguish between the real value and the perturbed value. For any query function Q and the global sensitivity ΔQ, the random algorithm $K(D) = Q(D) + N(0, \sigma^2)$, with

$$\sigma \geq \frac{\Delta Q}{\epsilon}\sqrt{2\ln(1.25/\delta)} \tag{3}$$

satisfies $(\epsilon, \delta)-$differential privacy, where the Gaussian noise is a Gaussian distribution with a mean of 0 and a covariance of σ^2. The noise level is proportional to the global sensitivity ΔQ and inversely proportional to the privacy budget ϵ. That is, the larger ΔQ is, the smaller ϵ is, the larger the injected noise is, the better the privacy protection effect is, and the lower the data utility is.

Differential privacy has some lemmas. In the differentially private deep learning [6], we can take advantage of the following lemmas:

Lemma 1 *Post-processing. Any calculation of output under differential privacy does not increase privacy loss.*

Lemma 2 *Serialized combination theorem. Serialized combination of differential privacy mechanisms still satisfies differential privacy protection.*

Next, we introduce how to apply differential privacy to the AI model. Minimizing the empirical risk function $L(w)$ to make the gradient update, this is the conventional step of training the AI model to obtain weight parameters w. At each step of the optimization procedure, we compute the gradient $g(\mathrm{x}_i)$ of the random sample and clip each gradient in l_2 norm, then we add noise and calculate the average gradients. The corresponding parameters are updated using the average gradient is the following.

$$\tilde{g}_t \leftarrow \frac{1}{L}\sum_i \left(\frac{\mathrm{g}_t(\mathrm{x}_i)}{\max(1, \|\mathrm{g}_t(\mathrm{x}_i)\|_2/C)} + N(0, \sigma^2)\right) \tag{4}$$

Then updates w by $\theta_{t+1} \leftarrow \theta_t - \eta\tilde{g}_t$. η is a learning rate. The clipping of the gradient here is to limit the impact of a single data on the whole, and it is also convenient to calculate the global sensitivity. Differential privacy protection of the AI model is achieved by injecting noise into the gradient during the optimization process.

3.2 Federated learning

Federated Learning is a machine learning framework that helps different users perform joint modeling without data sharing while meeting privacy protection, data security, and government regulations. Figure 2 shows the conventional process of federated learning proposed by Google [1].

Federated learning shown in Figure 2 has three main steps: A. The users train the local model and upload the model update; B. The users' model updates are gathered together in a server; C. The server optimizes the shared model, and each user gets a new model update. Repeat this process.

Yang [28] defines federated learning from the perspective of model performance: If there are N users $\{F_1, F_2, \ldots, F_N\}$, each user has its own data $\{D_1, D_2, \ldots, D_N\}$. The traditional approach is to gather the data together, $D = D_1 \cup D_2 \cup \ldots \cup D_N$ to train a model M_{sum}. The model performance of the model is V_{sum}. In federal learning, the users collaborate to train a model M_{fed}, in which no user will share his data with others, and the model performance is V_{fed}. If

$$|V_{sum} - V_{fed}| < \delta \tag{5}$$

then the federated learning model has a δ-accuracy loss.

We protect the privacy of federal learning through differential privacy. Figure 3 shows a typical federated learning framework with differential privacy updates.

Each user performs a model update locally and uploads parameters to the server in the cloud. The server averages the received parameters and performs differential privacy processing. The approach Averaging proposed by McMahan [17] is widely used in federal learning. Then the server distributes the averaged parameters to each user. The steps as follows.

– Suppose that there are K users, each user has its own datasets $\{B_1, B_2, \ldots, B_K\}$. Each user trains the model locally.

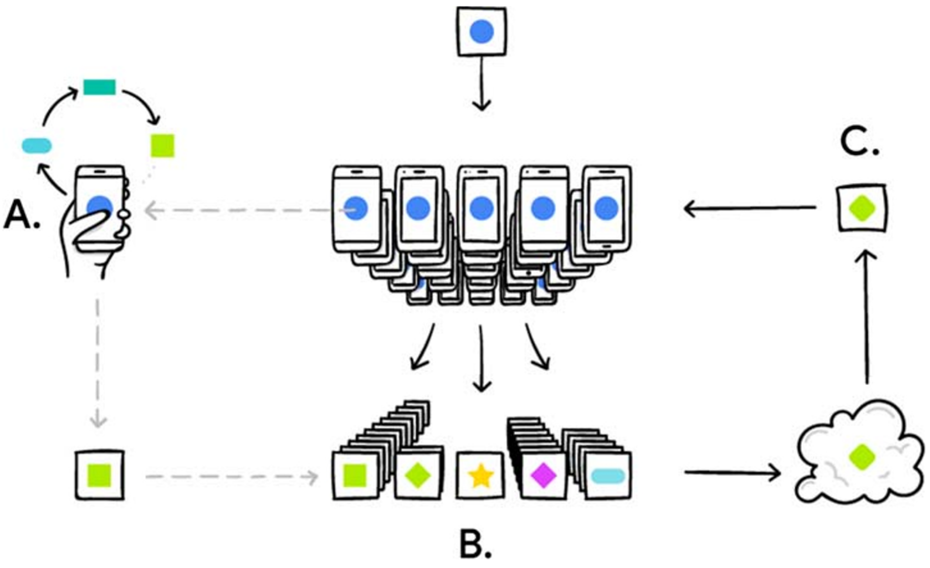

Figure 2 The main process of federal learning

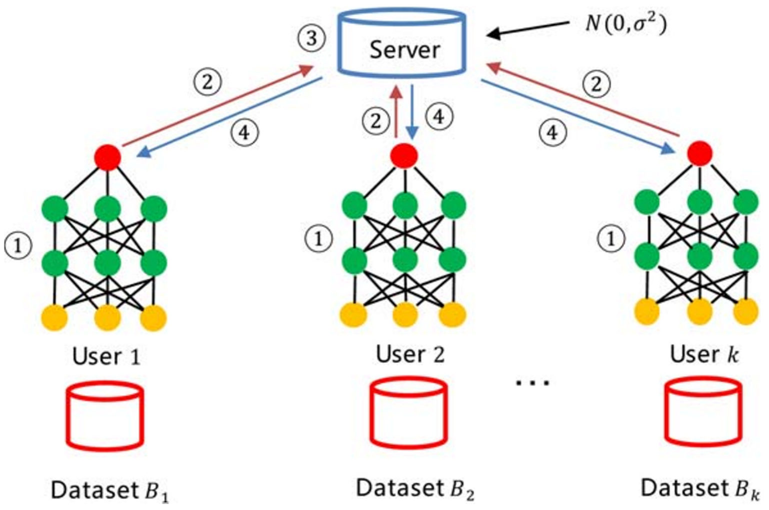

Figure 3 Federated learning framework with differential privacy update

- In the step $t+1$, each user uploads the parameters updates $\Delta w_{t+1}^k = w_{t+1}^k - w_t^k$ to the server, where w_{t+1}^k are the parameters trained locally by user k in the step $t+1$, and w_t^k are the parameters returned from the cloud in the step t.
- The server injects Gaussian noise into the parameters uploaded by each user and averages them: $w_{t+1} \leftarrow w_t + \frac{1}{K}\left(\sum_{k=1}^{K} \Delta w_{t+1}^k + N(0,\sigma^2)\right)$, where $N(0,\sigma^2)$ is a Gaussian noise with a mean of 0 and a variance of σ^2. Differential privacy protection is achieved through noise mechanism.
- The server sends the averaged parameters to each user for the next step of training.

Perform the above four steps of the loop until the preset privacy budget ϵ is exhausted, then the federated learning model training is completed. The above processes show a complete federated learning framework that protects the privacy of user data.

4 The DP-FL framework for the unbalanced data

This section introduces our framework: differentially private federated learning (DP-FL) for the unbalanced data. The traditional federated learning framework is mainly aimed at the balanced data; that is, each user has the same amount of data. DP-FL has the following two contributions:

- Different from the traditional method, our DP-FL framework set different privacy parameters for each users based on their amount of data;
- In DP-FL, we perform differential privacy protection on each user's model locally and then upload noise-disturbed parameters to the server. Therefore, DP-FL can address the unbalanced data scenario, which is superior to traditional methods.

Figure 4 shows the DP-FL framework. The model training has four steps. DP-FL mainly consists of two parts: server and users. Steps two and four are the processes of uploading

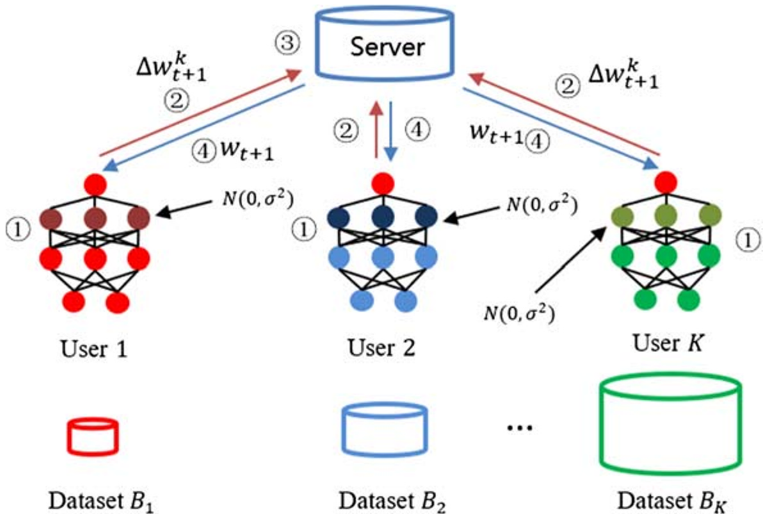

Figure 4 The DP-FL framework for the unbalanced data

and downloading parameters. Below we mainly introduce steps one and three of user and server operations.

4.1 The users' model parameters update

Step one. In the user's local model update, we use noise perturbation to achieve differential privacy protection. We use the Gaussian mechanism to perturb the user's local model gradient. First, we calculate the gradient,

$$g_t(x; b) \leftarrow \nabla L(w; b) \tag{6}$$

where g represents the gradient of x in step t, and b is the batch size. Then we need to clip the gradient,

$$g_t(x; b) = g_t(x; b)/\max(1, \|g_t(x; b)\|_2/S) \tag{7}$$

gradient clipping ensures that the second norm of the gradient is limited to the range of S. S is the global sensitivity. Next, we add Gaussian noise to disturb the gradient,

$$\tilde{g}_t(x; b) \leftarrow \frac{1}{b}\sum(g_t(x; b)/\max(1, \|g_t(x; b)\|_2/S) + N(0, S^2\sigma^2)) \tag{8}$$

where $N(0, S^2\sigma^2)$ is the gaussian noise. Then we can update the weight parameters,

$$w \leftarrow w - \eta\tilde{g}_t(x; b) \tag{9}$$

where η is the learning rate. Finally, the model update of user k can be obtained,

$$\Delta w^k_{t+1} = w^k_{t+1} - w^k_t \tag{10}$$

In each user's differential privacy model update, we use a novel algorithm: differentially private convolutional neural network with adaptive gradient descent (DPAGD-CNN) [10] to update the parameters. The advantage of DPAGD-CNN is that we can adaptively allocate the noise injected into the gradient. At the very beginning of the optimization process, the noise does not affect the correct direction of the gradient descent too much [13]. However, as the optimization progresses, the direction of the gradient descent becomes accurate. Small noise will affect the gradient decent direction. We divide the privacy budget ϵ into two parts. In each iteration, we use part of $\epsilon(\epsilon_g)$ allocated to compute the "noisy" gradient, and we use the remaining part of $\epsilon(\epsilon_n)$ allocated to select the optimal step size. We adaptively adjust the size of these two parts.

Algorithm 1 Differentially Private Convolutional Neural Networks with Adaptive Gradient Descent (DPAGD-CNN).

```
Input:
    Loss function $L(\theta)$, privacy parameters $\epsilon$ and $\delta$, gradient norm clipping $S$, budget
    increase rate $\gamma$, local batch size $b$.
Output:
    Weight parameters $w_t$.
1:  Initialize $\theta_0$ randomly
2:  $t \leftarrow 0$, $\epsilon_g$ and $\epsilon_n \leftarrow \epsilon$ randomly
3:  while $\epsilon > 0$ do
4:      $i \leftarrow 0$
5:      $g_t(\mathrm{x}) \leftarrow \nabla_{\theta_t} L(\theta_t, \mathrm{x})$
6:      $g_t(\mathrm{x}) \leftarrow g_t(\mathrm{x}) / \max(1, \|\mathrm{g}_t(\mathrm{x})\|_2 / S)$
7:      $\tilde{g}_t(\mathrm{x}) \leftarrow \frac{1}{N} \sum (g_t(\mathrm{x}) + N(0, S^2 \epsilon_g))$
8:      $\epsilon \leftarrow \epsilon - \epsilon_g$
9:      while $i = 0$ do
10:         $\Theta = \{L(w_t - \eta \tilde{g}_t(\mathrm{x})) : \eta \in \Phi\}$
11:         $\epsilon \leftarrow \epsilon - \epsilon_n$
12:         $i \leftarrow \mathrm{LapNoise}(\Theta, C, \sqrt{2\epsilon_n})$
13:         if $i > 0$ then
14:             if $\epsilon > 0$ then
15:                 $w_{t+1} \leftarrow w_t - \eta_i \tilde{g}_t$
16:             end if
17:         else
18:             $\epsilon_n \leftarrow (1+\gamma)\epsilon_n$, $\epsilon_g \leftarrow (1-\gamma)\epsilon_g$
19:         end if
20:     end while
21:     $t \leftarrow t + 1$
22: end while
23: return $w_t$
```

Algorithm 2 LapNoise(Θ, Δ, ϵ).

```
Input:
    a set of step size candidates $\Theta$,
    global sensitivity $\Delta f$,
    privacy budget $\epsilon$.
Output:
    The index $i$ of the best step size.
1:  $\tilde{\Theta} = \left\{ \tilde{v}_i = v + \mathrm{Lap}(\frac{\Delta f}{\epsilon}) : v \in \Theta \right\}$    //Lap is the Laplace noise.
2:  return $\arg\max_{i \in [|\Theta|]} \tilde{v}_i$
```

Algorithm 1 shows the detailed steps of DPAGD-CNN. In line 3, the loop is performed when the privacy budget ϵ is not exhausted. Line 5-7 compute the gradient, clip the gradient, and inject noise into the gradient. In line 8, part of the privacy budget is used to compute the gradient. Lines 9 to 20 describe the steps for adaptive adjusting ϵ_g and ϵ_n. We customize a collection of step sizes, called Θ. Each element of Θ is a loss function value. Step sizes can be set in advance. Then we use LapNoise function (Algorithm 2) to select the best step size. Executing a gradient descent when the optimal step size is obtained, otherwise, we reduce ϵ_g and increase ϵ_n.

Algorithm 2 introduces the LapNoise function. Taking candidates $\varTheta$, global sensitivity Δf, and privacy budget ϵ as input, the algorithm returns the index i of the best step size. The function Lap $\left(\frac{\Delta f}{\epsilon}\right)$ represents a Laplace distribution, whose the mean is 0 and the scale parameter is $\frac{\Delta f}{\epsilon}$.

Theorem 4.1 *Algorithm 1 satisfies (ϵ, δ)-differential privacy.*

Proof Privacy loss accumulates in each iteration. We use advanced differential privacy composition theorems [5] to track the privacy loss of each step of gradient updates. The composition theorem can provide tight bound for privacy losses. So according to Lemma2, we only need to ensure that the privacy budget is not exhausted. In each iteration, two operations incur privacy loss: "noisy" gradient (Line 8), LapNoise operation (Line 12). In line 3 and line 14, we make sure that the privacy budget $\epsilon > 0$. At each gradient update, we check if this update will cause ϵ to be less than 0 (Line 14). If a gradient update causes $\epsilon < 0$, we will not make this update. Line 3 controls the overall algorithm to satisfy differential privacy. So we proved that Algorithm 1 satisfies (ϵ, δ)-differential privacy. □

4.2 Parameters average in the server

Step three. Parameters average process of the server. Algorithm 3 details the entire process of DP-FL framework. Suppose that there are a total of K users participating in federated learning model training. In each round (communication between the server and the users), there is a random subset Z of size n $(n) \leq K$ that is sampled. Only the users of Z upload model parameters to the server,

$$\varDelta w_{t+1}^{k} \leftarrow \text{UserUpdate}(\epsilon/T, w_t) \tag{11}$$

The function UserUpdate is the users' parameters updating. T is the rounds of communication. The server averages the parameters uploaded by the users of Z,

$$w_{t+1} \leftarrow w_t + 1/n\left(\sum_{k=1}^{K} \varDelta w_{t+1}^{k}\right) \tag{12}$$

Then the server only sends the model parameter w_t (round t) to the users of Z. The randomization process can reduce the model training time and increase the robustness of the model. Our approach to addressing the unbalanced data is to set different ϵ (privacy parameters) for each user. The noise variance σ can be calculated from ϵ and δ. We set the same δ for each user. The more data the user own, the less the effect of noise on the gradient during training.

Algorithm 3 Differentially private federated learning algorithm for the unbalanced data.

```
Input:
    Number of users participating in federated learning $K$, privacy parameters set $\{\epsilon\}_{k=0}^{K}$
    and $\delta$, rounds of communication $T$, number of the users participating in each epoch of
    communication $n$.
Output:
    The weight parameters $w_T$ of the server model.
1:  //Server
2:  $w_0$ randomly    //Initialize the weights.
3:  for $t = 0, 1, 2, \ldots, T$ do
4:      $Z \leftarrow$ random set of $n$ users
5:      for $k \in Z$ in parallel do
6:          $\Delta w_{t+1}^{k} \leftarrow$ UserUpdate($\epsilon/T, w_t$)
7:      end for
8:      $w_{t+1} \leftarrow w_t + 1/n(\sum_{k=1}^{K} \Delta w_{t+1}^{k})$    //Update the parameters.
9:  end for
10: return $w_{t+1}$    //The last server parameters.
11:
12: //Communication between server and users
13:
14: //User
15: function UserUpdate($\epsilon, w_t$)
16:     $\hat{w} \leftarrow w_t$
17:     $w =$ DPAGD-CNN($\epsilon, w_t$)
18:     $\Delta w_{t+1} = w - \hat{w}$
19: return $\Delta w_{t+1}$
```

Each user performs a model update locally and uploads parameters to the server. The server averages the received parameters and sends the parameters to the user subset Z. For each user, we use the DPAGD-CNN method to get the local parameters. When a round of privacy budget is exhausted, the risk of the user data leakage has increased to a critical point. The user quits federated learning training, and other users continue to train locally. We can guarantee that our DP-FL framework meets differential privacy protection.

5 Experiments

5.1 Experiment setup

In this paper, we use three standard datasets, MNIST, CIFAR-10 and Iris for the experiments. The MNIST dataset consists of 70,000 handwritten digital grey-level images, including 60,000 training images and 10,000 test images. Each image is 28×28 in size. The labels are digits from 0 to 9. The CIFAR-10 dataset consists of 10 categories of RGB color images. The image size is 32×32. These ten categories include: airplane, car, bird, cat, deer, dog, frog, horse, boat, truck. There are a total of 50,000 training examples and 10,000 test examples. The Iris flower dataset is a classic dataset that is often used as an example

Figure 5 The datasets used in our experiments. (left: MNIST, middle: CIFAR-10, right: Iris)

in statistical learning or machine learning. The dataset contains a total of 150 records in 3 categories, each with 50 records. Each record has four features: Sepal length, Sepal width, Petal length and Petal width. We can use these four features to predict which species of iris flower (iris-setosa, iris-versicolour or iris-virginica) belongs to. Figure 5 shows these three datasets.

Since CNN has good feature extraction ability, we use CNN as the neural network architecture. Our DPAGD-CNN architecture for each users' model updating on the MNIST dataset has two convolution layers and two fully-connected layers. The feature maps of the two convolution layers are 32 and 64 with five kernels and one stride. In the fully-connected layer, we use dropout to prevent overfitting. The last layer is the output layer with ten digits. The batch size is 600. In each layer, we inject noise into the gradient for differential privacy protection. The DPAGD-CNN architecture for each users' model updating on the CIFAR-10 dataset is similar to that on the MNIST dataset. The neural networks has two convolution layers and three fully-connected layers. In each convolution layer, we use batch normalization to speed up training. In each layer, we inject noise into the gradient for differential privacy protection. We also use DPAGD-CNN architecture in the Iris dataset. Since the Iris dataset is small, we only use one layer of fully-connected layer. Before injecting noise into the gradient, we need to clip the gradient norm. The gradient norm clip threshold we set is 0.01 for the three datasets.

We compare our DP-FL framework with a state-of-the-art framework [8]: a client level differentially private federated learning (CL-FL). We use classification accuracy as the standard for model evaluation. We use Google's TensorFlow to build CNN architectures. The python version we use is 3.6.4. Also, We use a single GPU, i.e., NVIDIA Tesla P100, 16 GB with 3,584 CUDA cores for hardware acceleration.

5.2 Performance

This subsection shows our experimental performance. Table 1 shows the comparison of the experimental performance of our DP-FL framework with the client level federated learning

Table 1 Comparison of DP-FL framework with the client level federated learning (CL-FL) framework [8] on the MNIST dataset

	# of users	δ	Accuracy of DP-FL	Accuracy of CL-FL
No-DP	10	–	0.976	
DP	10	e-2	0.928	0.76
	100	e-3	0.939	0.78
	1000	e-5	0.957	0.92

Table 2 The accuracy of DP-FL framework on the unbalanced data and the balanced data on the three datasets

	Users	Acc on MNIST	Acc on CIFAR-10	Acc on Iris
Unbalanced data	100	0.939	0.718	0.945
Balanced data	100	0.942	0.729	0.955

(CL-FL) framework [8] on the MNIST dataset. The number of communication rounds we set is 20. Privacy parameter δ is set to e-2, e-3, e-5 for 10, 100, 1000 users. Firstly, we compare the effect of differential privacy on accuracy on both frameworks for 10 users. In the non-differentially private setting (noDP), both frameworks can achieve a classification accuracy of 97.6%. In the differentially private setting, Our DP-FL framework classification accuracy is 92.8%, compared with 76% of the client level FL framework. It is necessary to sacrifice a little model performance for differential privacy protection. The classification accuracy of DP-FL framework also exceeds the client level FL framework for 100, 1000 users.

Table 2 shows the classification accuracy of DP-FL framework on the unbalanced data and the balanced data for 100 users on the three datasets. We hope that DP-FL framework can achieve the same accuracy as on the balanced data since we mainly target the unbalanced data scenario. From Table 2 we can see that the classification accuracy on the unbalanced data and the balanced data are almost the same on all datasets. This demonstrates the effectiveness of DP-FL framework for different data scenarios.

Figure 6 shows the trend of the classification accuracy of DP-FL framework along with communication round for the unbalanced data on the three datasets. In each subfigure, we compare the settings of the two scenarios: the non-differentially private setting (noDP-FL) and the differentially private setting (DP-FL). For example, in the left subfigure, the blue line indicates the accuracy of federated learning framework without differential privacy protection on the MNIST dataset. The accuracy does not fluctuate greatly with the growth of communication round since there is no noise to disturb the gradient. The orange line indicates the accuracy of our DP-FL framework. The accuracy varies slightly with the growth of communication round due to noise disturbance. The final accuracy of DP-FL is almost the same as that of noDP-FL. Our DP-FL framework has good model performance while protecting user data privacy. The middle subfigure shows the trend of the classification accuracy of DP-FL framework along with communication round for the unbalanced data on the CIFAR-10 dataset. The red line indicates the accuracy of federated learning framework without differential privacy protection. The green line indicates the accuracy of our DP-FL framework. We can see that there is a slight accuracy fluctuation in the DP-FL framework,

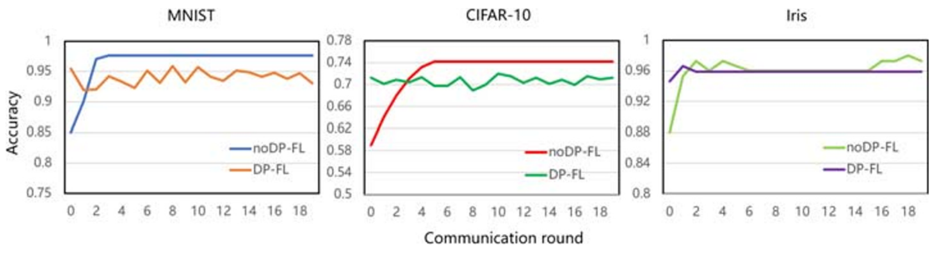

Figure 6 The DP-FL framework for the unbalanced data on the three datasets. (left: MNIST, middle: CIFAR-10, right: Iris)

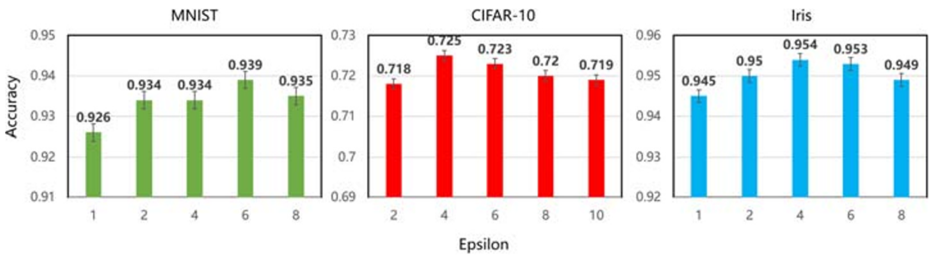

Figure 7 The accuracy of each user model in DP-FL framework on the three datasets. (left: MNIST, middle: CIFAR-10, right: Iris)

which is within acceptable limits. Accuracy is maintained at a high level in the DP-FL framework. In the right subfigure, the accuacies of DP-FL and noDP-FL are basically the same on the Iris dataset. The Iris dataset is too small, differential privacy has little effect on federated learning framework.

Figure 7 shows the accuracy of each user model in DP-FL framework on the three datasets. Suppose that there are 100 users, we divide them into five categories for the convenience of experiments. Each category has an equal amount of data. We set different privacy parameters ϵ from 1 to 8 for each category of user based on their amount of data on the MNIST and Iris dataset. The CIFAR-10 dataset is more difficult to train, so we set larger privacy parameters, from 2 to 10. After 20 communication rounds of training, all category of user achieves a good classification accuracy on the three datasets. For example, We achieve the accuracies from 92.6% to 93.9% on the MNIST dataset. By uploading and downloading parameters, each user can benefit from the parameters of other users without data sharing. This explains why users with different degrees of privacy protection have similar model performance.

To support the superiority of our DPAGD-CNN method used in the single user's model parameters update of DP-FL framework, we compare DPAGD-CNN with two state-of-the-art models on MNIST and CIFAR-10. The first one is the differentially private stochastic gradient descent (**DP-SGD**) CNN model-based proposed by Abadi et al. [22]. He uses "moment accountant" to track privacy loss in the optimization process. The second one is the adaptive Laplace mechanism (**AdLM**) proposed by Phan et al. [1]. He adaptively injects noise into the average relevance of input features, coefficients of the differentially private layer and coefficients of the approximated loss function to preserve $(\epsilon_1 + \epsilon_2 + \epsilon_3)$-differential privacy.

Figure 8a shows the classification accuracy of the three models under different privacy budget ϵ. ϵ changes from 0.2 to 8. We can see that our model, DPAGD-CNN, performs

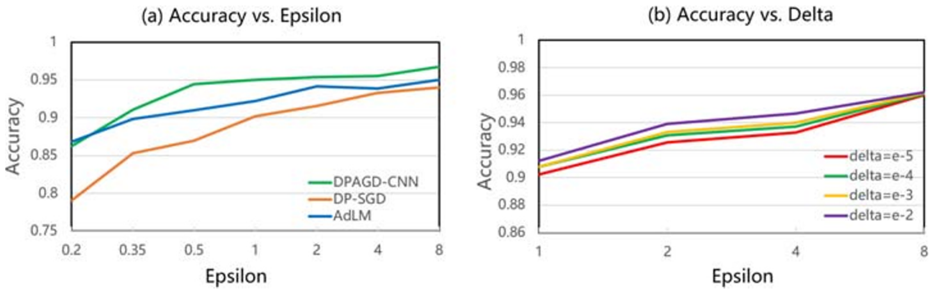

Figure 8 Effects of privacy budget on accuracy on the MNIST dataset. (left: ϵ, right: δ)

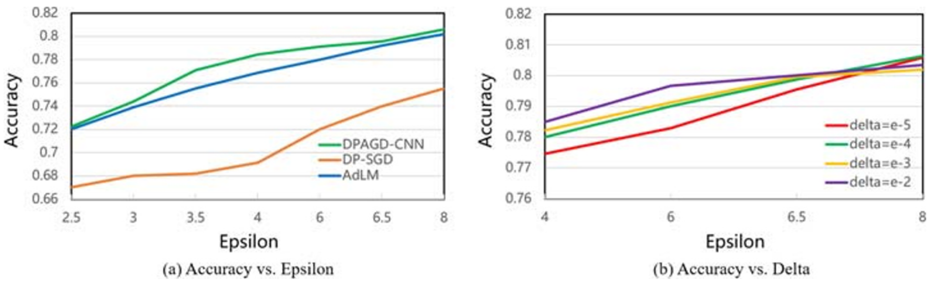

Figure 9 Effects of privacy budget on accuracy on the CIFAR-10 dataset. (left: ϵ, right: δ)

better than DP-SGD and AdLM. When privacy budget $\epsilon = 0.5$, our model classification accuracy is 94.41%, compared with 86.98% of DP-SGD and 90.96% of AdLM. This is a surprising performance. Because small privacy budget ϵ means a strong privacy guarantee, which inevitably reduces the prediction accuracy. When ϵ is large, e.g., $\epsilon = 8$, our model achieves 96.71%. This is close to the prediction accuracy without privacy protection. The curves in Figure 8b portray the effect of relaxation factor δ on the prediction accuracy under different privacy budget, e.g., ϵ =1, 2, 4 and 8. δ changes from 10^{-5} to 10^{-2}. No matter how big ϵ is, δ has little effect on the prediction accuracy of the model. Privacy budget ϵ is the main factor affecting the prediction accuracy.

Similar to the experimental results on MNIST, Figure 9a portrays that our model, DPAGD-CNN, has batter performance better than DP-SGD and AdLM on CIFAR-10. For instance, when privacy budget $\epsilon = 4$, our model classification accuracy is 78.46%, compared with 69.12% of DP-SGD and 76.88% of AdLM. The curves in Figure 9b portray the effect of relaxation factor on the prediction accuracy under privacy budget $\epsilon = 4, 6, 6.5$ and 8. δ changes from 10^{-5} to 10^{-2}. δ has little effect on the accuracy.

6 Conclusion

In this paper, we present a novel federated learning framework based on differential privacy technique (DP-FL) to protect the data privacy for multiple users. Furthermore, we design a novel differential private convolutional neural networks with adaptive gradient descent (DPAGD-CNN) algorithm to protect the privacy of each user's data. DF-FL framework provides a two-level protections in cloud server and user client, simultaneously. Experimental results on several real-world datasets verify the effectiveness of DP-FL under the unbalanced data scenario.

Acknowledgements This work is partly supported by the National Key Research and Development Program of China under grand No. 2017YFB0802204. National Natural Science Foundation of China under the grand No.61976051 and Basic Research Project of Shenzhen under grant No.JCYJ20180306174743727 and National and provincial program supporting projects of Shenzhen, China No. GJHS20170313113617970.

References

1. Abadi, M., Chu, A., Goodfellow I., McMahan, H., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 23th ACM Conference on Computer and Communications Security. ACM (2016)
2. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning (2018)

3. Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konecny, J., Mazzocchi, S., McMahan, H., et al: Towards federated learning at scale: system design. arXiv:1902.01046 (2019)
4. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy preserving machine learning. In: ACM Conference on Computer and Communications Security (ACM CCS) (2016)
5. Dwork, C.: Differential privacy. In: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming, Venice, Italy, pp. 1–12 (2006)
6. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science **9**(3–4), 211–407 (2014)
7. Du, W., Han, Y., Chen, S.: Privacy-preserving Multivariate Statistical Analysis: Linear Regression and Classification. In: Proceedings of the 2004 SIAM international conference on data mining. pp. 222–233 (2004)
8. Geyer, R., Klein, T., Nabi, M.: Differentially private federated learning: a client level perspective. NIPS Workshop: Machine Learning on the Phone and other Consumer Devices (2017)
9. Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., Ramage, D.: Federated learning for mobile keyboard prediction. arXiv:1811.03604 (2018)
10. Huang, X., Liao, Q., Qi, S., Guan, J., Jiang, Z., Wang, X.: Differentially Private Convolutional Neural Networks with Adaptive Gradient Descent. IEEE International Conference on Data Science in Cyberspace (DSC). Hangzhou, China (2018)
11. Konečny, J., McMahan, H., Ramage, D., Richtarik P.: Federated optimization: distributed machine learning for on-device intelligence. arXiv:1610.02527 (2016)
12. Konečny, J., McMahan, H., Yu, F., Richtárik, P., Bacon, D.: Federated learning: strategies for improving communication efficiency. arXiv:1610.05492 (2016)
13. Lee, J., Kifer, D.: Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1656–1665 (2018)
14. Li, N., Li, T., Venkatasubramanian S.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: Proceedings of the IEEE International Conference on Data Engineering(ICDE). Istanbul, Turkey, pp. 106–115 (2007)
15. Liu, Y., Chen, T., Yang Q.: Secure federated transfer learning. arXiv:1812.03337 (2018)
16. Machnavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: l-diversity: Privacy beyond k-anonymity. In: Proceedings of the 22nd International Conference on Data engineering(ICDE). Atlanta, Georgia, USA, pp. 24–35 (2006)
17. McMahan, B., Ramage, D.: Federated learning: Collaborative machine learning without centralized training data. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html, Accessed 04 Oct 2018 (2018)
18. McMahan, H., Moore, E., Ramage, D., Hampson, S., Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) (2017)
19. Mohassel, P., Zhang, Y.: Secureml: A System for Scalable Privacy-preserving Machine Learning. In: Proceedings of the 2017 IEEE Symposium on Security and Privacy(SP), pp. 19–38 (2017)
20. Shokri, R., Shmatikov, V.: Privacy-preserving deep learning. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS), New York, USA, pp. 1310–1321 (2015)
21. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., Hassabis, D.: Mastering the game of go with deep neural networks and tree search. Nature **529**, 484–503 (2016)
22. Smith, V., Chiang, C., Sanjabi, M., Talwalkar, A.: Federated multi-task learning. In: Advances in Neural Information Processing Systems, pp. 4427–4437 (2017)
23. Sweeney, L.: k-anonymity: a model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based System **10**(5), 24–35 (2002)
24. Wang, W., He, S., Sun, L., Jiang, T., Zhang, Q.: Cross-technology Communications for Heterogeneous IoT Devices Through Artificial Doppler Shifts. IEEE Trans. Wirel. Commun. **18**(2), 796–806 (2019)
25. Wang, W., Zhang, Q.: Privacy-preserving collaborative spectrum sensing with multiple service providers. IEEE Trans. Wirel. Commun. **14**(2), 1011–1019 (2014)
26. Wang, W., Chen, L., Zhang, Q.: Outsourcing high-dimensional healthcare data to cloud with personalized privacy preservation. Comput. Netw. **88**, 136–148 (2015)
27. WeBank: https://github.com/WeBankFinTech/FATE (2020)

28. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated machine learning: concept and applications. ACM Trans. Intell. Syst. Technol. **10**(2), Article 12 (2019)
29. Yuan, J., Yu, S.: Privacy Preserving Back-propagation Neural Network Learning Made Practical with Cloud Computing. IEEE Trans. Parallel Distrib. Syst. **25**(1), 212–221 (2013)
30. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. arXiv:1806.00582 (2018)

Affiliations

Xixi Huang[1] · Ye Ding[2] · Zoe L. Jiang[1,3] · Shuhan Qi[1] · Xuan Wang[1] · Qing Liao[1,3]

Xixi Huang
huangxixi@stu.hit.edu.cn

Ye Ding
dingye@dgut.edu.cn

Zoe L. Jiang
zoeljiang@hit.edu.cn

Shuhan Qi
shuhanqi@cs.hitsz.edu.cn

Xuan Wang
wangxuan@cs.hitsz.edu.cn

[1] School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen, 518000, China

[2] School of Computer Science and Network Security, Dongguan University of Technology, Dongguan, 523000, China

[3] Peng Cheng Laboratory, Shenzhen, 518000, China