Practical Private Aggregation in Federated Learning Against Inference Attack

Ping Zhao, *Member, IEEE*, Zhikui Cao, Jin Jiang, and Fei Gao, *Member, IEEE*

***Abstract*—Federated learning (FL) enables multiple worker devices share local models trained on their private data to collaboratively train a machine learning model. However, local models are proved to imply the information about the private data and, thus, introduce much vulnerabilities to inference attacks where the adversary reconstructs or infers the sensitive information about the private data (e.g., labels, memberships, etc.) from the local models. To address this issue, existing works proposed homomorphic encryption, secure multiparty computation (SMC), and differential privacy methods. Nevertheless, the homomorphic encryption and SMC-based approaches are not applicable to large-scale FL scenarios as they incur substantial additional communication and computation costs and require secure channels to delivery keys. Moreover, differential privacy brings a substantial tradeoff between privacy budget and model performance. In this article, we propose a novel FL framework, which can protect the data privacy of worker devices against the inference attacks with minimal accuracy cost and low computation and communication cost, and does not rely on the secure pairwise communication channels. The main idea is to generate the lightweight keys based on computational Diffie–Hellman (CDH) problem to encrypt the local models, and the FL server can only get the sum of the local models of all worker devices without knowing the exact local model of any specific worker device. The extensive experimental results on three real-world data sets validate that the proposed FL framework can protect the data privacy of worker devices, and only incurs a small constant of computation and communication cost and a drop in test accuracy of no more than 1%.**

***Index Terms*—Computational Diffie–Hellman (CDH), data privacy, federated learning (FL), inference attacks.**

I. Introduction

TO PROTECT the data privacy of worker devices in centralized machine learning, federated learning (FL) was proposed to enable multiple worker devices to share local models trained on their private data to the FL server [1], [2], [3], [4]. However, existing works show that these local models imply the information about the private data and, thus, introduce much vulnerabilities to inference attacks.

Although these local models contain significantly less information about the private data, the adversary can still reconstruct worker devices private data or infer the sensitive information about the private data, e.g., labels, memberships, properties, etc., and these are called inference attacks [5], [6]. Specifically, in FL system, worker devices train the local models on their private data, and then share these local models with the honest-but-curious FL server that aggregates the local models to obtain the global model [7]. Although the private data remains well-preserved privacy, the local models entail the privacy disclosure of the private data. Moreover, the recent works [8], [9], [10], [11] have shown that FL is susceptible to inference attacks where the adversary (i.e., the honest-but-curious FL server) can reconstruct the private data or infer sensitive information about the private data. Furthermore, worker devices are vulnerable to serious attacks, i.e., spams, even blackmails and physical violence, in the event of private data being inferred. Therefore, it is desirable to design a privacy-preserving algorithm to prevent the adversary from getting the exact local models while collaboratively training the global model [12].

Existing works concerning privacy-preserving FL can be largely classified into homomorphic encryption, secure multiparty computation (SMC), and differential privacy methods. Specifically, the works [13], [14], [15] designed homomorphic encryption schemes that allow the FL server to perform arithmetic operations on ciphertexts. Other kinds of works [16], [17], [18], [19] apply SMC to FL settings to enable worker devices to encrypt and/or secret-share their local models to the honest-but-curious FL server. In addition, the studies [6], [20], [21], [22], [23], [24], [25], [26], [27] designed local differential privacy schemes that perturb local models via injecting noise before sending the local models to the honest-but-curious FL server. Nevertheless, the homomorphic encryption and SMC-based approaches are not applicable to large-scale FL scenarios as they incur substantial additional communication and computation cost [28]. Moreover, they require the secure channels to delivery keys to worker devices. In addition, differential privacy brings a substantial tradeoff between privacy budget and model performance.

To this end, we propose a novel FL framework which can protect the data privacy of worker devices against the inference attacks with minimal accuracy cost and low computation and

Manuscript received 1 March 2022; revised 6 May 2022, 2 July 2022, and 26 July 2022; accepted 15 August 2022. Date of publication 24 August 2022; date of current version 22 December 2022. This work was supported in part by the Open Foundation of State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications under Grant SKLNST-2021-1-06, and in part by the National Natural Science Foundation of China under Grant 61902060. *(Corresponding author: Fei Gao.)*

Ping Zhao is with the College of Information Science and Technology, Donghua University, Shanghai 200051, China, and also with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: pingzhao2018ph@dhu.edu.cn).

Zhikui Cao and Jin Jiang are with the College of Information Science and Technology, Donghua University, Shanghai 200051, China (e-mail: 2201876@mail.dhu.edu.cn; 2201776@mail.dhu.edu.cn).

Fei Gao is with the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: gaof@bupt.edu.cn).

Digital Object Identifier 10.1109/JIOT.2022.3201231

communication cost, and does not rely on the secure pairwise communication channels. Motivated by the existing work [29], our main idea is to generate the lightweight keys based on the computational Diffie–Hellman (CDH) problem to encrypt the local models, and the FL server can only get the sum of the local models of all worker devices without knowing the exact local model of any a specific worker device. Finally, we conduct extensive experiments on three real-world data sets, CIFAR-10, MNIST, and Fashion-MNIST, and compare our work to the existing work federated averaging (FedAvg) [30]. The experimental results validate that our proposed FL framework is desirable as it protects the local models with a small constant of computation and communication cost and a drop in test accuracy of no more than 1% at most. Overall, the main contributions of this article are as follows.

1) The proposed FL framework can protect worker devices' local models from being disclosed to the honest-but-curious worker devices, the honest-but-curious FL sever, and any untrusted third party, and does not rely on the secure channels and any trusted party in FL.
2) The proposed FL framework can obtain the test accuracy which is almost equal to that of the initial FL in most cases, and more importantly, only incurs a small constant of computation and communication cost for each worker device, making our work quite lightweight than the existing works.
3) We conduct extensive experiments, and the results validate that the proposed FL framework is applicable to the practical FL scenarios where a larger number of worker devices participant in FL, and their local data are highly nonindependent and identically distributed (Non-IID).

The remainder of this article is organized as follows. Section II reviews the related work. Section III introduces the preliminary knowledge. Section IV describes the design of the proposed FL framework in detail, followed by the theoretical analysis in Section V. Thereafter, Section VI evaluates the proposed FL framework. Finally, Section VII concludes this article.

II. Related Work

We first review existing works concerning the inference attacks on FL and then review the corresponding privacy-preserving mechanisms in FL.

A. Inference Attacks on FL

Inferring Class Representative: The existing work [8] exploits generative adversarial network (GAN) to generate imitated data which appears to come from the same distribution with the training data. The adversary performs like a normal worker device participating in training. But it used the real-time feature of FL to train a GAN. The GAN aims at class representative. Only when all class members are similar, the GAN is effective. However, training a GAN needs substantial amount of computational resources. Another work [5] proposed a new class of model inversion attack, which is based on the confidence values and, moreover, applied the attack to both the decision trees for lifestyle surveys and neural networks for facial recognition.

Inferring Membership: Member inference attack can infer whether a specific data is involved in training. Such membership information can reveal more sensitive information of users. For example, if the adversary inferred a specific user's data participating in an FL that trains a certain disease diagnosis model, then the adversary can know the health status. The existing work [31] infers which words are in the training batched of benign worker device at a deep natural language processing (NLP). The work [32] proposed a gradient ascent attack to trick the FL server to expose more information about the local data to infer the membership information of users.

Inferring Properties: Attribute inference attack includes passive attack and active attack [31]. Both of them assume the adversary has an extra training data set labeled with correct target property. Passive attack sets a binary classifier by observing the training process. Active attack tricks the FL server to extract more information, it is stronger than passive attack. The existing work [6] designed passive and active inference attacks that utilize the updates and can infer when a specific user first appears in the training.

Inferring Training Inputs and Labels: The existing work [9] proposed deep leakage from gradients (DLGs) which can easily "steal" training data with only 20 lines codes. They recovered the image at pixel level, and the acquired text was matched. Different from previous attack methods, this attack is more threatening. Furthermore, the work [10] greatly improved the accuracy of label inference based on shared gradients, and proposed the correlation between the tag and the gradient symbol.

B. Privacy-Preserving FL

The privacy-preserving mechanism in FL can be largely classified into homomorphic encryption, SMC, and differential privacy.

1) Homomorphic Encryption: Many works tried to apply homomorphic encryption to FL, aiming to protect the local models of worker devices from the honest-but-curious FL server which may actively or passively infer the users' information from the received local models. For instance, the existing works [13], [14], [15], [33], [34] proposed privacy-preserving machine learning via Paillier encryption. The studies [14], [35], [36] designed a privacy-preserving machine learning system that combines LWE-based encryption. Likewise, the literatures [15], [37], [38] proposed privacy-preserving machine learning via ring-LWE-based encryption. Those works use their framework combined with encryption algorithms, among which commonly used are Paillier encryption, LWE-based encryption and ring-LWE-based encryption. However, the privacy preservation provided in these works spend extra communication cost and computation cost. We have summarized the communication cost and the computation cost in Table I.

In order to reduce the cost of homomorphic encryption, many researches have designed lightweight homomorphic encryption algorithms. For example, the study [33] proposed

TABLE I
COMMUNICATION AND COMPUTATION COST OF OUR WORK AND SEVERAL EXISTING WORKS

	Our work	Paillier encryption	LWE-based encryption	Ring-LWE-based encryption
Communication cost	n_d	$2n_d(pred + pad)$	$(n_{lwe} + n_d * prec) \log_2 q$	$2n_d * prec * \log_2 q$
Computation cost	N_{data}	$N_{data} * \frac{n_d}{t} * T_{PaiAdd}$	$N_{data} * T_{lweAdd}$	$N_{data} * \frac{n_d}{t} * T_{rlweAdd}$

TABLE II
PROS AND CONS OF THE PROPOSAL AS COMPARED WITH THE EXISTING RELATED SCHEMES

	Our work	Homomorphic Encryption	Secure Multiparty Computation	Differential Privacy
Communication cost	medium	high	high	low
Computation cost	medium	high	high	medium
Accuracy	high	high	high	low
Level of privacy	high	high	high	medium

that parameters are encrypted after gradient compression and shearing, and literature [34] used an improved Paillier algorithm which can speed up the training by 25%–28%. Moreover, work [39] deployed a subset of the model weights in plaintext to improve the overall performance. Another study [40] used leveled homomorphic encryption to reduce the computational cost. Similarly, work [41] proposed applying partial homomorphic encryption to improved the overall efficiency. However, the encryption key needs to be distributed to users through a secure channel in these works. In contrast, in this article, we proposed a novel FL framework which can protect the data privacy of worker devices against the inference attacks with the minimal accuracy cost and lowest computation and communication cost, and more importantly does not rely on the secure pairwise communication channel.

2) Secure Multiparty Computation: SMC enables different participants with private inputs to perform a joint computation on their inputs without revealing them to each other. The existing work [16] designed secure two-party computation for linear regression, logistic regression, and stochastic gradient descent (SGD) method. The study [17] applied SMC to FL setting to against the passive and active adversary. Another literature [18] focused on the secure two-party computation and additive secret sharing, and proposed a hybrid framework. Nevertheless, SMC provides a high level of privacy and accuracy at the expense of large computation and communication cost.

3) Differential Privacy: Differential privacy can be classified into centralized differential privacy and local differential privacy. But only local differential privacy can be applicable to the FL settings. The existing work [20] used differential privacy to provide strong privacy guarantees for training data in a black-box fashion. The follow-up work [21] designed a local differential privacy mechanism for the SGD algorithm. Another work [6] designed the locally differentially private mechanism for statistical learning problems to prevent the adversary from reconstructing the users' local data. Thereafter, the existing works [22], [23] proposed new locally differentially private mechanism for FL, and work [24] combined locally differentially private and homomorphic encryption. The latest works [25], [26], [27] focused on applying locally differentially privacy to the high-dimensional local models with continuous values. However, in locally differentially private, a lot of noise is injected into the local models or local data, thus incurring huge utility degradation.

We have summarized the pros and cons of the proposal and the existing related schemes in Table II.

III. PRELIMINARIES

A. Federated Learning

FL was proposed to solve the privacy security problem and data isolated island problem [11], [42], [43]. FL enables worker devices to train the model locally with their private data and only upload their local model parameters to the FL server to update the global model. It is proved that the global model can still converge and maintain high accuracy.

In every global iteration, each worker device c_i downloads the global parameters W_g from the FL server. Next, it updates its local model parameters w_i using its private data set D_i and, e.g., SGD, $w_i = w_i - \alpha \Delta w$. $\Delta w = ([\partial f(w_i, D_i)]/\partial w_i)$ denotes the gradient, and α denotes the learning rate. Then, each c_i uploads its local model parameters w_i to the FL server. Finally, the FL server aggregates the local models of all worker devices to update the global parameters W_g for the next global iteration.

B. Disclosure of Data Privacy in FL

When updating the local model parameters, the neural network needs to go through two processes: 1) forward propagation and 2) backpropagation. The forward propagation obtains the gradient parameters, and the backpropagation updates the parameters according to the gradient. So, it can be seen that the gradient w is the differential of the model to the parameters, and the parameter updating is the linear change of the gradient. In other words, the local model parameters w_i are a specific mapping of exact local data, and the information of local data of worker devices may be leaked from these local model parameters. As shown in Fig. 1, in FL, the honest-but-curious FL server and insecure transmission channels are easy to cause the leakage of local model parameters. What is more, a large number of inference attacks have been proposed, such as member inference, attribute inference, model inversion, etc., which have brought great challenges to FL. For example, Zhu *et al.* [9] proposed DLG, which can easily reconstruct training data with only 20 lines codes. In summary, it is desirable to design privacy-preserving FL

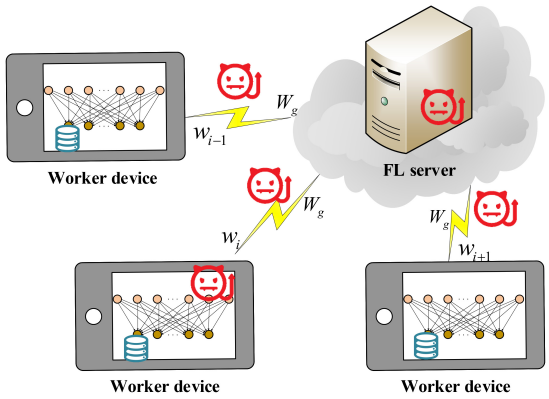

Fig. 1. Illustration of the data privacy disclosure in FL, where w_i is the local model parameters and W_g is the global parameter.

framework to prevent the privacy disclosure from the local model parameters.

C. Threat Model and Problem Definition

Threat Model: We assume that each worker device and the FL server are honest-but-curious. All worker devices and the FL server are trying to infer the training data of other worker devices (i.e., the local data). They honestly complete the training process according to the FL protocol. In addition, the communication channels among worker devices and the FL server are not secure. Other organizations, e.g., hacker may steal the transmitted data from the insecure channels.

Problem Definition: In this article, we consider the federated average algorithm, which means the FL server updates the global parameters by calculating $W_g = (w_1 + w_2 + \cdots + w_n)/n$. It can be seen from the formula that if the FL server can calculate the sum of local model parameters $\sum_{k=1}^{n} w_i$ directly without knowing the exact local model parameters of any a specific worker device, the worker devices do not need to send the exact local model parameters to the FL server. On this basis, even the communication channels are not secure, the adversary cannot infer any information about the local data sets, as the worker devices do not upload the exact local model parameters. Therefore, we only need to guarantee that the FL server can compute the accurate sum of local model parameters when worker devices send the encrypted local model parameters to the FL sever, and that it is computationally expensive for any adversary (e.g., the honest-but-curious FL server, hacker, or the honest-but-curious worker devices) to obtain the exact local parameters of a specific worker device with the help of the encrypted local model parameters.

D. Security Model

Specially, we assume that all the communication channels in our algorithm are insecure. Anyone (the FL server, worker devices) can steal the public parameters and ciphertext being transferred. In the worst case, we need to overcome the challenge of the collusion attack. We take advantage of the computational difficulty of the CDH problem. Any probabilistic polynomial time adversary (PPTA) is computational expensive to solve the following problem [29].

TABLE III
FREQUENTLY USED NOTATIONS AND THE CORRESPONDING EXPLANATIONS

$\mathbb{G}$, g	Multiplicative group, and its generator
W_g	Global model parameter
c_i	Client
w_i	Local model parameter
d_i	Mapping of local model parameter
X_i	Public parameter
K_i	Secret-Key
L_i	Ciphertext

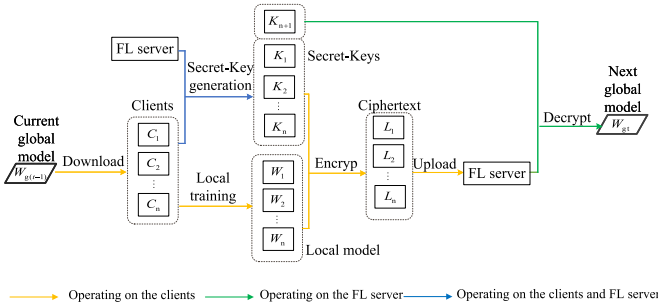

Fig. 2. Overview of our system.

We consider the following problems in the multiplicative group $\mathbb{G}$ with generator g.

Definition 1 [Decisional Diffie–Hellman (DDH) Problem in $\mathbb{G}$]: The DDH problem is defined as follows: given $g, g^a, g^b, g^c \in \mathbb{G}$ where $a, b, c \in Z$, decide if $g^{ab} = g^c$.

Definition 2 (CDH Problem in $\mathbb{G}$): The CDH problem is defined as follows: given $g, g^a, g^b \in \mathbb{G}$ where $a, b \in Z$, compute g^{ab} without knowing a or b.

CDH is harder than DDH. Anyone who can solve the CDH problem can compute g^{ab} to solve the DDH problem. Our algorithm is based on the assumption that it is computational expensive to solve the CDH problem.

IV. Design of the Proposed FL Framework

Our main idea is to enable the worker devices to send the encrypted local model parameters to the FL server and the FL server can get the accurate sum of all worker devices' local model parameters at the same time. Even the channels are not secure and parties in FL are honest-but-curious, the adversary cannot infer the exact local parameters and, thus, cannot further infer the local data of worker devices. In this section, we will introduce the design of the proposed FL framework in detail. Fig. 2 shows an overview of our system framework. Table III summarizes the notations frequently used in this article.

A. Overview

We need to design a series of keys $K_i \in \mathbb{G}$ to encrypt the local model parameters w_i. Then, we need to calculate $\sum_{i=1}^{n} w_i$ in FL server. In the encryption process, the ciphertext is obtained by multiplying the plaintext and the secret key K_i, as shown in (3). While in the decryption process, the ciphertext are multiplied, as shown in (4). Therefore, these secret keys K_i should satisfy $\prod_{i=1}^{n} K_i = 1 \bmod q^2$, where q is a big

positive integer. The FL server can obtain the sum of local model parameters by (1) and (2). Privacy security is reflected in that worker devices can easily get their own key K_i, but it is computationally expensive for any adversary to calculate the keys. Overall, the proposed FL framework mainly consists of three steps in each global iteration, which is to be introduced in Section IV-C. Our work is motivated by the existing work [29], and we use the similar idea to design our work. The difference is that we concentrate on the FL and, thus, we are facing different challenges. Interested readers can refer to the work [29] for more details.

B. Definitions

For calculating $\sum_{i=1}^{n} w_i$, we introduce some formulas

$$(1+p)^n = 1 + np \bmod p^2. \tag{1}$$

From the above equation, we can get that

$$\prod_{i=1}^{n}(1+p)^{w_i} = \prod_{i=1}^{n}(1+pw_i) = \left(1 + p\sum_{i=1}^{n} w_i\right) \bmod p^2. \tag{2}$$

From the above formula, we can calculate $\sum_{i=1}^{n} w_i$. We choose a q-order cyclic multiplicative group $\mathbb{G}$ defined as $<g>$, where g is generator. g is a prime number, and p should larger than the sum of parameters. Then, we divided worker devices and the FL server into a group (as shown in Fig. 1). In each epoch, the FL server calculates the sum of all local model parameters $\sum_{i=1}^{n} w_i$ in this group to update the global model.

C. Methodology

In this part, we describe more details about the three steps.

Step 1: Each worker device c_i downloads the global parameters W_g from the FL server. Next, they update their local parameters w_i by their local data sets D_i, respectively. In local training, we use SGD to update the next local model parameter $w_{i(t+1)} = w_{it} - \alpha([\partial f(w_{it}, D_i)]/\partial w_{it})$, where t donates current training epoch and α donates the learning rate. In order to support algorithms over floating-point numbers, the model parameters w_i should be mapped to integers. To address the challenge, we have designed the following strategy: first, add a large enough positive integer to make all parameters positive, $w_i + q$, $q \in N^+$. Second, in order to ensure the accuracy, we will retain m decimal places, then expanding it to integer, $d_i = 10^m * [w_i + q]$. Parameter m donates the number of decimal places reserved, and d_i donates the quantized parameter. After decrypting the ciphertext, the FL server will calculate $\sum_{i=1}^{n} w_i$ according to the following strategy.

Step 2: 1) Each worker device randomly generates a number $k_i \in \mathbb{Z}_q$. The FL server generates k_{n+1} as worker device c_{n+1}. Then, they calculate $X_i = g^{k_i} \in \mathbb{G}$ as public parameters, where g is a prime number and 2) each worker device c_i sends $X_i \in \mathbb{G}$ to her neighbor worker devices After receiving the public parameters, each worker device c_i calculates $K_i = ([X_{i+1}]/[X_{i-1}])^{k_i} \in \mathbb{G}$ as a secret parameter. Particularly, $K_{n+1} = (X_1/X_n)^{k_{n+1}}$ and $K_1 = (X_2/X_{n+1})^{k_1}$. In order to encrypt d_i, each worker device c_i first computes $(1+d_ip)$. Then, the worker devices get the ciphertext (i.e., the encrypted local model parameters) by multiplying their own secret parameter K_i

$$L_i = (1 + d_ip) * K_i \tag{3}$$

where L_i donates the ciphertext. The FL server keeps its own secret parameters K_{n+1}, and it does not need to do the above calculation. In fact, the FL server does not have parameters d_{n+1} for calculation. Finally, worker devices can upload the ciphertexts L_i to the FL server.

Step 3: Upon receiving the ciphertexts L_i from worker devices, the FL server calculates the following L:

$$\begin{aligned} L &= K_{n+1}\prod_{i=1}^{n} L_i \bmod p^2 \\ &= \left(g^{k_1}/g^{k_n}\right)^{k_{n+1}} \prod_{i=1}^{n}(1+d_ip)\left(g^{k_{i+1}}/g^{k_{i-1}}\right)^{k_i} \bmod p^2 \\ &= \left(1 + p\sum_{i=1}^{n} d_i\right) g^{\sum_{i=1}^{n} k_{i+1}k_i - k_ik_{i-1}} \bmod p^2 \\ &= \left(1 + p\sum_{i=1}^{n} d_i\right) \bmod p^2. \end{aligned} \tag{4}$$

Then, the FL server calculates $(L-1)/p = \sum_{i=1}^{n} d_i \bmod p^2$ to recover $\sum_{i=1}^{n} d_i$. Thereafter, the FL server does the inverse transformation of the linear transformation mentioned in Step 1 to get $\sum_{i=1}^{n} w_i$. Finally, the FL server updates the global parameters W_g according to the aggregation rules (i.e., FedAvg).

In a training epoch, we have $n+1$ devices, including the FL server and n worker devices. When a certain worker device drop calls, We only need to rearrange the worker devices and repeat step 2, which will not affect the training process.

V. Theoretical Analysis of Performance

In this section, we analyze the data privacy this work provides and the computation and communication cost this work incurs.

A. Security Analysis

In our algorithm, public parameters need to be exchanged between work devices. So, we need at least two worker devices to participate in the training. Since only the FL server knows the sum of parameters, as long as there are two participants, the FL server is not able to infer any worker devices' local data set.

Theorem 1: Our practical private aggregation is secure based on CDH in $\mathbb{G}$.

Proof: For any ciphertext in the process of our algorithm, we have

$$L_i = (1 + d_ip) * K_i = (1 + d_ip) * g^{(k_{i+1}-k_{i-1})k_i}. \tag{5}$$

Given $(1 + d_ip) * K_i$ to infer d_i, any adversary has to obtain the secret parameter $K_i = g^{(k_{i+1}-k_{i-1})k_i}$. Note that

TABLE IV
COMPUTATION AND COMMUNICATION COSTS

Parties	Our work				FedAvg	
	Computation cost		communication cost		Computation cost	Communication cost
FL sever	$O(n)$		$O(n \mid p \mid)$		$O(n)$	$O(n \mid \omega_i \mid)$
Each worker device	Setup	Encrypt	Setup	Encrypt	$O(1)$	$O(\mid \omega_i \mid)$
	$O(1)$	$O(1)$	$O(\mid p \mid)$	$O(\mid p \mid)$		

any adversary can only obtain $X_{i-1} = g^{k_{i-1}}$, $X_i = g^{k_i}$, and $X_{i+1} = g^{k_{i+1}}$ from the insecure communication channel or honest-but-curious FL server. Then, a PPTA can have $X_{i+1}/X_{i-1} = g^{k_{i+1}-k_{i-1}}$ and $X_i = g^{k_i}$. Obviously, to solve $g^{(k_{i+1}-k_{i-1})k_i}$, it is a CDH problem defined in $\mathbb{G}$. Therefore, inferring worker device's private data during our training process is at least as hard as a CDH problem in $\mathbb{G}$ for any PPTA and, thus, our work is secure based on CDH in $\mathbb{G}$. ■

Next, we analyze the security of our work tolerant to the collusion attacks of t adversaries, and we get the following results.

Theorem 2: Our practical private aggregation tolerant to collusion attacks of t adversaries is secure based on CDH in $\mathbb{G}$.

Proof: Given $(1 + d_i p) * K_i$ to infer d_i, any adversary has to solve the secret parameter

$$K_i = \left(g^{k_{i+t+1}}/g^{k_{i-1}}\right)^{k_{i+t}k_{i+t-1},\ldots k_{i+2}k_{i+1}k_i}. \tag{6}$$

Collusive adversaries can manipulate the value $k_{i+t+1} - k_{i-1}$. For a certain adversary, he can collude with $t-1$ adversarial worker devices to get $t-1$ random numbers k_i's in the exponent. Although, there are $t+1$ random numbers k_i's in the exponent, at least two random numbers remain unknown to the adversaries. Therefore, the following exponent:

$$(k_{i+t+1} - k_{i-1})k_{i+t}k_{i+t-1}, \ldots k_{i+2}k_{i+1}k_i \tag{7}$$

remains unknown. The adversaries still cannot solve K_i with g^{k_i}.

Let A_{i-1} be a certain adversary and let $A_{i+1}, \cdots, A_{i+t+1}$ be $t-1$ remaining adversarial worker devices. Then, adversarial worker devices can obtain

$$\left(g^{k_{i+t+1}}/g^{k_{i-1}}\right)^{k_{i+t}k_{i+t-1},\ldots k_{i+2}k_{i+1}} \tag{8}$$

from the $t+1$th round of the exchanges and g^{r_i} from the first round of the exchanges, and they have to solve K_i. However, this is a CDH problem.

Therefore, launching collusion attack in our work is at least as hard as a CDH problem in $\mathbb{G}$ for any PPTA. Thus, our work is secure based on CDH in $\mathbb{G}$. ■

B. Complexity Analysis

We analyze the computation and communication cost of the proposed FL framework, which is shown in Table IV. For an overview of the comparative work, we briefly summarize its cost in Table IV at the same time. The computation complexity of Setup, Encrypt, and aggregation is $O(1)$, $O(1)$, and $O(n)$. Since the Setup and Encrypt are performed by the worker devices and, thus, computation cost of each worker device is $O(1)$. The aggregation is conducted by the FL server and, thus, computation cost of the FL server is $O(n)$. In each global iteration, Setup, Encrypt, and aggregation are conducted one time and, thus, incur the communication cost $O(\mid p \mid)$, $O(\mid p \mid)$, and $O(n \mid p \mid)$, respectively, where $\mid p \mid$ is the length of p. The computation and communication costs of the FL server in FedAvg are $O(n)$ and $O(n \mid \omega_i \mid)$, where $\mid \omega_i \mid$ is the length of local model ω_i. The computation and communication costs of each worker device in FedAvg are $O(1)$ and $O(\mid \omega_i \mid)$.

TABLE V
CNN ARCHITECTURES FOR MNIST AND FASHION-MNIST

Layer	Size
Convolution	$1 \times 5 \times 5$
Max Pooling + ReLU	2×2
Convolution	$32 \times 5 \times 5$
Max Pooling + ReLU	2×2
Fully Connected + ReLU	3136
Fully Connected	512
Log Softmax	10

VI. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed FL framework.

A. Experimental Setup

Data Set: We use three classical data sets: 1) CIFAR-10; 2) MNIST; and 3) Fashion-MNIST. The MNIST data set is from the National Institute of Standards and Technology (MNIST). It contains 60 000 training data and 10 000 testing data. All of the data are 28 28 grayscale images. The Fashion-MNIST data set is the same as MNIST except that it is clothes picture. It is more complex than MNIST. The CIFAR-10 data set contains 60 000 RGB color images of 32 32.

Data Set Pretreatment: First, we divide the training set into ten groups, where the parameter 10 donates the number of data classes. For the Non-IDD setting, we select a fraction d of training data with the mth label and distribute them to the mth group, where d donates the degree of Non-IDD. The remaining training data are randomly assigned to the ten groups. $d = 1$ means that each group's training data have completely different labels. In our experiment, we set $d = 0$, $d = 0.5$, $d = 0.8$, and $d = 1$, respectively. Then, we randomly select a group for each worker devices, and randomly extract the required amount of data from this group.

Definition 3 [Attack Error (AE)]: We define the mean square error between the dummy picture and the original picture as the AE.

Machine Learning Models: For MNIST and Fashion-MNIST, we use convolutions neural networks (CNNs) with the architecture described in Table V. For CIFAR-10, we also use CNN, but the CNN is more complex. Its architecture is

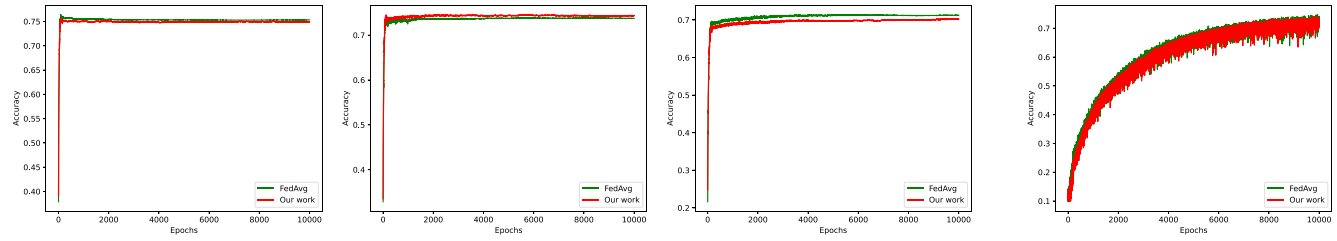

Fig. 3. Test accuracies of our work and FedAvg varying with the level of Non-IID on the data set, CIFAR-10. (a) Level of Non-IID 0. (b) Level of Non-IID 0.5. (c) Level of Non-IID 0.8. (d) Level of Non-IID 1.

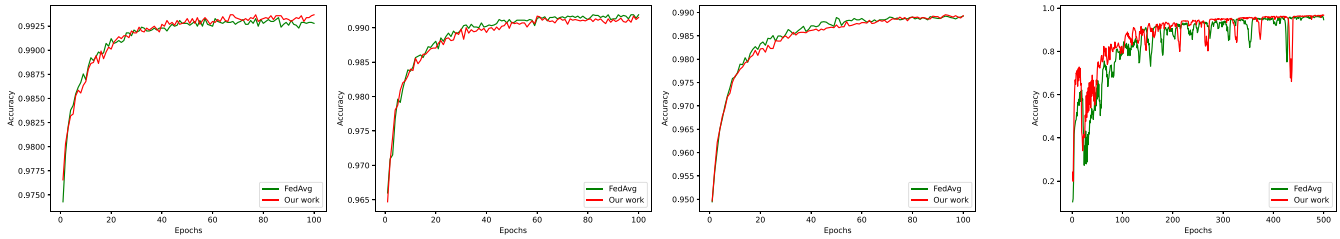

Fig. 4. Test accuracies of our work and FedAvg varying with the level of Non-IID on the data set, MNIST. (a) Level of Non-IID 0. (b) Level of Non-IID 0.5. (c) Level of Non-IID 0.8. (d) Level of Non-IID 1.

TABLE VI
CNN ARCHITECTURES FOR CIFAR-10

Layer	Size
Convolution	$3 \times 5 \times 5$
Max Pooling + ReLU	3×2
Convolution	$64 \times 5 \times 5$
Fully Connected + ReLU	1024
Fully Connected + ReLU	384
Fully Connected	192
Log Softmax	10

described in Table VI. Our work does not focus on achieving high accuracy of the global model. It is intended to protect the local models from the honest-but-curious FL server, worker devices, and any other attackers while incurring a small drop in test accuracy.

Parameter Settings: In local model training, we use SGD. We set the learning rate to 0.01, local batch size to 32, and local epoch to 1. For the global epochs, different data set and different variables will affect the convergence speed of the global model, so we set appropriate global epochs for various situations. The FL sever uses FedAvg to update the global model (i.e., parameters) W_g. In default settings, the number of worker devices is 10, the number of significant digits m is 7, and the degree of non-IID is 0.5.

Comparative Works: We compare our work with the existing work FedAvg [30], the differential privacy algorithm (hereafter DP) and Cheon *et al.* [44] (hereafter CKKS). In FedAvg, worker devices locally training local models and send local models to the FL server. The FL server updates the global model via averaging the sum of all local models. In DP, Gaussian noise is added to the local model parameters to protect the data privacy, and the accuracy of the global model will be affected by the noise. The adversary can reconstruct the local data with noisy using the uploaded parameters. CKKS is a homomorphic encryption scheme for floating-point numbers. It supports an approximate addition and multiplication of encrypted messages. The local model parameters are encoded and converted in plaintext for further encryption. The encoding process is to map complex vectors into polynomials. After encrypting the plaintext, the ciphertext is truncated into a smaller modulus, which leads to rounding of plaintext. The significant figures which contain a main message are added a noise, and the FL server decrypts the ciphertext and outputs an approximate value of plaintext with a predetermined precision.

Attack Mode: We use DLG to verify the confidentiality of the algorithm. The main idea of DLG is to assume the virtual input and label, calculate the virtual gradient, optimize the distance between the virtual gradient and the real gradient, and make the virtual data close to the original data by matching the gradient.

Metrics: We use test accuracy as the evaluation metric. We consider three factors that may affect our work, i.e., the number of worker devices for training, the degree of Non-IID of worker devices' local data, and the number of significant digits we keep when amplifying and rounding worker devices' local model parameters.

B. Experimental Results

1) Impact of the Level of Non-IID: Figs. 3–5 show the impact of the level of Non-IID on the test accuracy in the three data sets. We can observe that with the increment of the degree of Non-IID, the global model is more difficult to converge, and the test accuracy is more fluctuating in the test data set. But, the test accuracy converges in the end. In addition, we can see that the test accuracy of our work in the three data sets are almost equal to that of FedAvg. The test

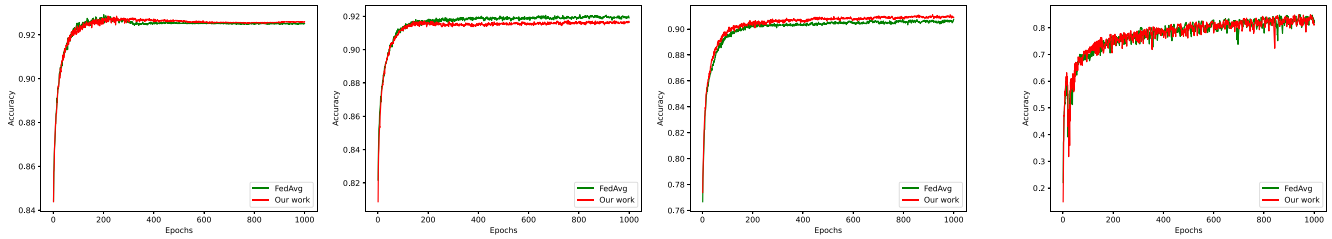

Fig. 5. Test accuracies of our work and FedAvg varying with the level of Non-IID on the data set, Fashion-MNIST. (a) Level of Non-IID 0. (b) Level of Non-IID 0.5. (c) Level of Non-IID 0.8. (d) Level of Non-IID 1.

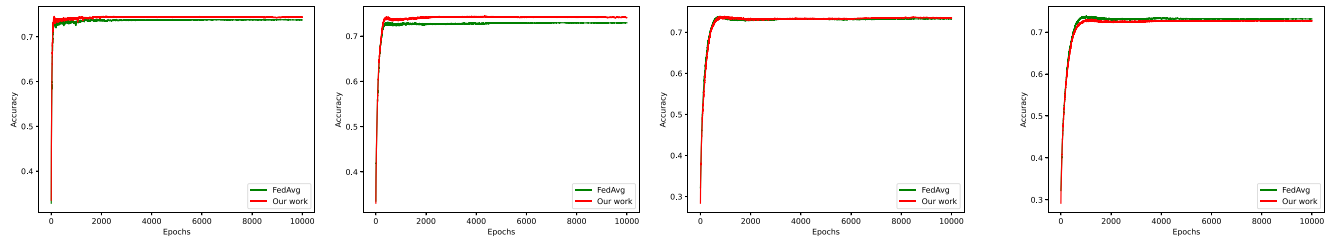

Fig. 6. Test accuracies of our work and FedAvg varying with the number of worker devices on the data set, CIFAR-10. (a) Number of worker devices 10. (b) Number of worker devices 40. (c) Number of worker devices 80. (d) Number of worker devices 100.

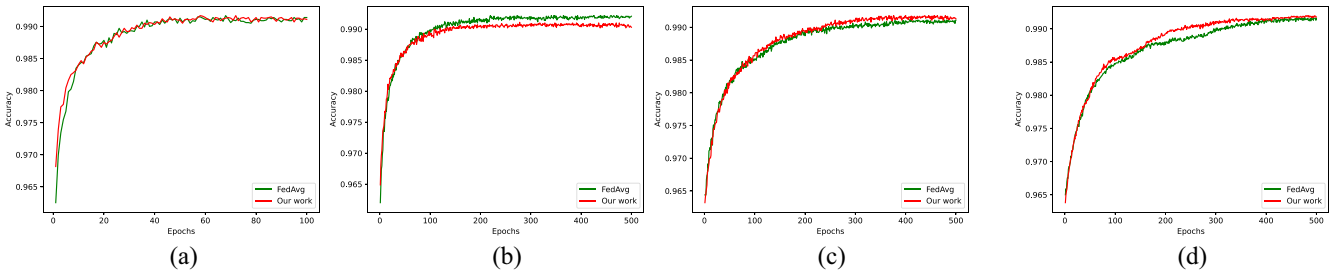

Fig. 7. Test accuracies of our work and FedAvg varying with the number of worker devices on the data set, MNIST. (a) Number of worker devices 10. (b) Number of worker devices 40. (c) Number of worker devices 80. (d) Number of worker devices 100.

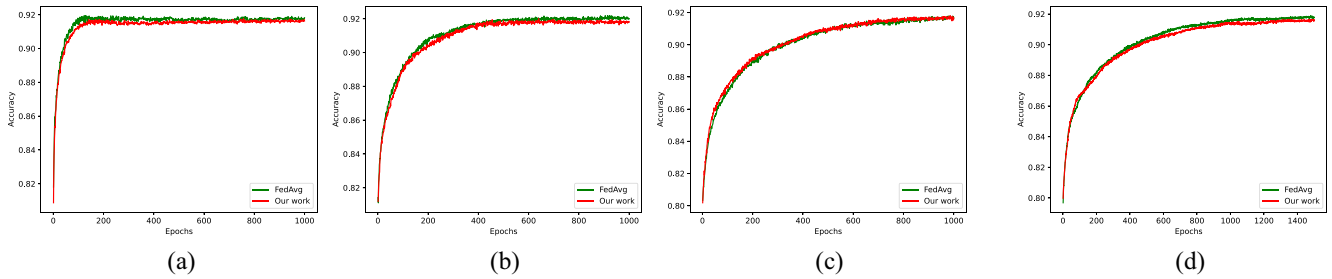

Fig. 8. Test accuracies of our work and FedAvg varying with the number of worker devices on the data set, Fashion-MNIST. (a) Number of worker devices 10. (b) Number of worker devices 40. (c) Number of worker devices 80. (d) Number of worker devices 100.

accuracy of our algorithm is at most 1% in Fig. 3(c) and 0.4% in Fig. 5(b) lower than the accuracy of FedAvg. The reason is that our work does not bring in much errors to the global models and, thus, our work can protect the data privacy of local models and guarantee the test accuracy of the global model at the same time. Moreover, it can be observed that, with the increasing level of Non-IID, the difference between the test accuracy of our algorithm and the accuracy of the FedAvg has not gradually increased. It indicates that our work is robust to the increasing level of Non-IID and, thus, is more applicable to the practical scenario where worker devices' data sets are highly Non-IID. Last, we can see that both in the three data sets, our work can obtain the desirable performance and, therefore, our work is general.

2) Impact of the Number of Worker Devices: It can be seen from Figs. 6–8 that the increase of the number of worker devices does not affect the difference between the test accuracy of our work and the FedAvg, but only affect the speed of convergence. The test accuracy of our algorithm is at most 70.2% in Fig. 3(c) and 91.6% in Fig. 5(b), 1% and 0.4% lower

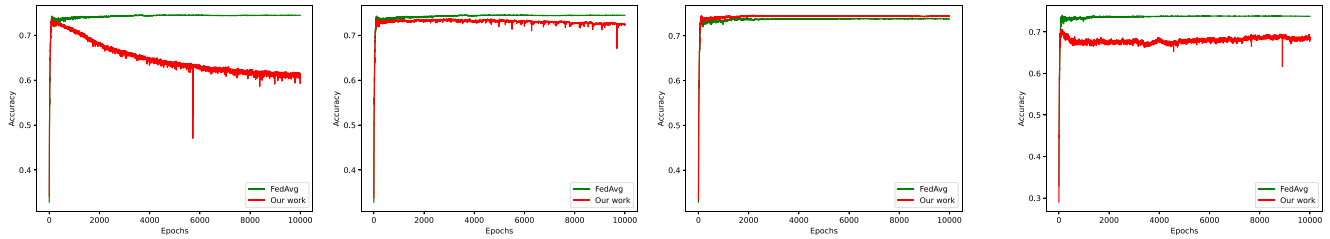

Fig. 9. Test accuracies of our work and FedAvg varying with the number of significant digits on the data set, CIFAR-10. (a) Number of significant digits 5. (b) Number of significant digits 6. (c) Number of significant digits 7. (d) Number of significant digits 8.

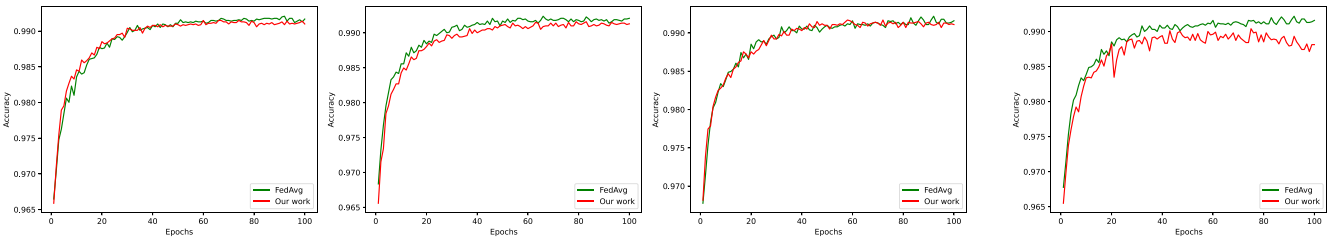

Fig. 10. Test accuracies of our work and FedAvg varying with the number of significant digits on the data set, MNIST. (a) Number of significant digits 5. (b) Number of significant digits 6. (c) Number of significant digits 7. (d) Number of significant digits 8.

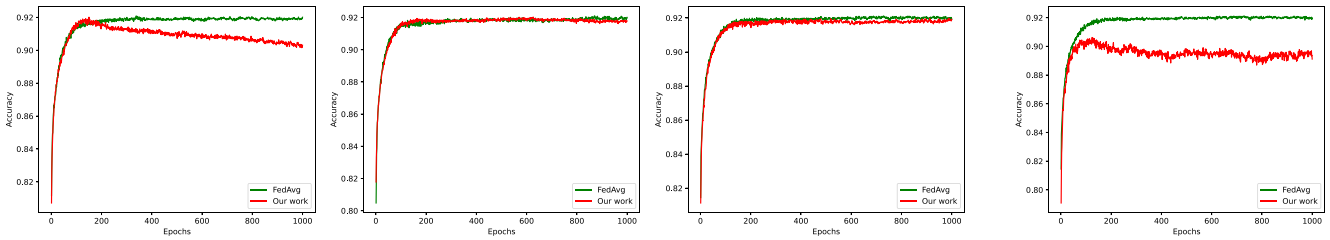

Fig. 11. Test accuracies of our work and FedAvg varying with the number of significant digits on the data set, Fashion-MNIST. (a) Number of significant digits 5. (b) Number of significant digits 6. (c) Number of significant digits 7. (d) Number of significant digits 8.

than the accuracy of FedAvg, respectively. This is because even if each parameter will generate errors when retaining the number of significant digits and encrypting, and, moreover, the more worker devices, the more errors. But when the FL server aggregates and updates the local model parameters, it will divide the sum of all local model parameters by the number of worker devices, so that the errors will also be averaged. Therefore, the average test accuracy will not change greatly regardless of the number of worker devices. Furthermore, both in the three data sets, our work can obtain the similar test accuracy with FedAvg with the increasing number of worker devices. Thus, our work is desirable, as it also protect the data privacy.

3) Impact of the Number of Significant Digits: Figs. 9–11 show the impact of the number of significant digits on test accuracy in the three data sets. As we can see, when the number of significant digits increases, the difference between the test accuracy of our work and that of FedAvg first decreases and then gradually increases. Specifically, in the data sets CIFAR-10, MNIST, and Fashion-MNIST, when the number of significant digits are 7 and 6, respectively, our work achieve the almost same test accuracy with FedAvg. In Fig. 9, in the data set CIFAR-10, the test accuracies of our work and FedAvg are 74.5% and 73.7%, respectively, with the number of significant digit is 7. Similarly, in Fig. 10, in the data set MNIST, the test accuracy of our work is 99.10% and that of FedAvg is 99.13% with the number of significant digit is 7. In Fig. 11, in the data set Fashion-MNIST, the test accuracy of our work

TABLE VII
TIME COST AND COMMUNICATION COST (100 ITERATIONS)

Dataset	Cost	FedAvg	Our work	CKKS
CIFAR-10	Total time (Sec)	1840.12	1874.78	2027.45
	Algorithm time (Sec)	–	23.56	48.21
	Accuracy	75.55%	75.11%	75.35%
	Communication cost (MB)	2.20	4.4	32.77
	Extra time cost	–	1.87%	10.9%
Fashion-MNIST	Total time (Sec)	2462.75	2537.53	2924
	Algorithm time (Sec)	–	42.39	57.59
	Accuracy	92.81%	92.79%	92.68%
	Communication cost (MB)	6.35	12.69	94.58
	Extra time cost	–	3%	18.7%
MNIST	Total time (Sec)	2331	2402.86	2791.01
	Algorithm time (Sec)	–	41.36	56.86
	Accuracy	99.33%	99.30%	99.25%
	Communication cost (MB)	6.35	12.69	94.58
	Extra time cost	–	3%	19.7%

TABLE VIII
AE OF DUMMY DATA AND GROUND-TRUTH DATA IN DIFFERENT METHODS

	FedAvg	DP with $\epsilon = 0.001$	DP with $\epsilon = 0.005$	DP with $\epsilon = 0.01$	Our work with 2 clients	Our work with 3 clients	Our work with 4 clients	Our work with 5 clients
AE	$1.16 * 10^{-8}$	$6.44 * 10^{-5}$	0.0023	0.0058	0.086	0.074	0.067	0.088
Accuracy	99%	98%	97%	95.8%	99%	99%	99%	99%

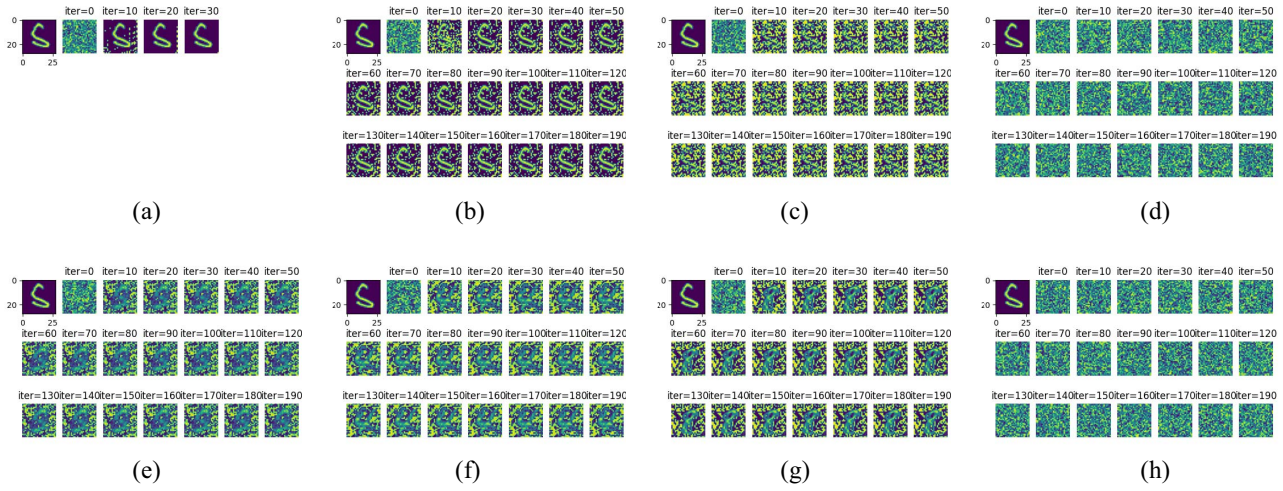

Fig. 12. Dummy pictures of DLG attacking FedAvg, DP and our work on MNIST. (a) FedAvg. (b) DP with $\epsilon = 0.001$. (c) DP with $\epsilon = 0.005$. (d) DP with $\epsilon = 0.01$. (e) Two worker devices in our work. (f) Three worker devices in our work. (g) Four worker devices in our work. (h) Five worker devices in our work.

is 91.79% and that of FedAvg is 91.96% with the number of significant digit is 6.

Therefore, the optimal number of significant digits for the data set, CIFAR-10, MNIST, and Fashion-MNIST are 7, 7, and 6. The reasons are that the more digits reserved for rounding during the linear transformation, and the fewer low-order digits to be discarded, the higher the test accuracy of the global model. However, when the number of significant digits reaches a specific value, e.g., eight significant digits in data set Fashion-MNIST, due to the errors brought in by the FL server decrypting the ciphertext, the test accuracy of the global model decreases a lot.

4) Time Cost and Communication Cost: We record the time cost of 100 iterations on a RTX3090 GPU, and results are shown in Table VII. We use CKKS as comparative work and use FedAvg as baseline. In CKKS, poly_modulus_degree is set to 8192, coeff_mod_bit_sizes is set to $[60, 40, 40]$ and scale_bits is set to 40. The total time includes the time cost of the algorithm, the local training and parameter aggregation by server. The algorithm time consists of encryption time and decryption time. The communication cost refers to the size of local model parameters or the size of encrypted local model parameters. Extra time cost means that the total time exceeds the baseline, FedAvg. We can observe that the three algorithms can achieve approximate accuracy on the three data sets, but there is a big gap in algorithm time and communication cost. On the three data sets, compared to the baseline FedAvg, communication cost in our work is doubled, and that is enlarged by about 15 times in CKKS. The algorithm time of CKKS is one to two times that of our work. The extra time cost in our work is 1.87% on CIFAR-10, 3% on Fashion-MNIST, and 3% on MNIST, while the extra time cost in CKKS is 10.9% on CIFAR-10, 18.7% on Fashion-MNIST, and 19.7% on MNIST. Generally speaking, compared to CKKS, our work only needs a little communication cost and algorithm time cost while obtaining similar privacy protection and model accuracy. In summary, our work is more desirable for the privacy-preserving FL scenarios.

5) Privacy Protection: We use DLG to verify the confidentiality of the proposed algorithm. We use DP as comparative work and use FedAvg as baseline. First, we conduct an inference attack on the gradient parameters uploaded by a specific worker device in FedAvg, and get the dummy picture and AE. Then, we use DP and our encryption algorithm, respectively. For DP, we use DLG to attack the gradient parameters with Gaussian noise. For our work, the adversary can only know the sum of parameters. After averaging the sum of parameters, we use DLG to carry out the inference attack to obtain dummy pictures and AE. We set the number of iterations to 300 and stop iterating when the loss is less than 0.000001. The dummy pictures are shown in Fig. 12. AE and test accuracy are summarized in Table VIII.

It can be seen from Table VIII that the AE of our algorithm is much higher than that of FedAvg, which is similar to DP. It means that there is a large gap between the dummy picture and the original picture. It can be seen from Fig. 12 that FedAvg is inferred within dozens of iterations when being attacked. While our algorithm converges after 300 iterations, and the adversary cannot recover the original picture. Moreover, there are more worker devices taking part in training, and the recovered picture is more blurred. Thus, our algorithm well protects the worker device's data privacy. When there is a high privacy level, for example, $\epsilon = 0.01$, DP has the same privacy protection ability, but it incurs a drop in test accuracy.

Summary of Experimental Results: The extensive results above validate that our work can protect the data privacy of

local model parameters while obtaining almost the same test accuracy with FedAvg in different settings.

VII. Conclusion

In this work, we proposed a novel FL framework which can protect the data privacy of worker devices against the inference attacks with the minimal accuracy cost and low computation and communication costs and does not rely on the secure pairwise communication channels. The main idea is to hide worker devices' exact local model parameters and send the encrypted local model parameters to the FL server. The extensive experimental results on three real-world data sets validated that the proposed FL framework can protect the data privacy of worker devices, and incurs a small constant of computation and communication cost and a drop in test accuracy of no more than 1%.

References

[1] P. Zhao, J. Tao, L. Kangjie, G. Zhang, and F. Gao, "Deep reinforcement learning-based joint optimization of delay and privacy in multiple-user MEC systems," *IEEE Trans. Cloud Comput.*, early access, Jan. 4, 2022, doi: 10.1109/TCC.2022.3140231.

[2] P. Zhao *et al.*, "Garbage in, garbage out: Poisoning attacks disguised with plausible mobility in data aggregation," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 3, pp. 2679–2693, Jul.–Sep. 2021.

[3] P. Zhao, W. Liu, G. Zhang, Z. Li, and L. Wang, "Preserving privacy in WiFi localization with plausible dummy locations," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 11909–11925, Oct. 2020.

[4] P. Zhao, J. Li, F. Zeng, F. Xiao, C. Wang, and H. Jiang, "ILLIA: Enabling k-anonymity-based privacy preserving against location injection attacks in continuous LBS queries," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 1033–1042, Apr. 2018.

[5] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2015, pp. 1322–1333.

[6] A. Bhowmick, J. C. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," 2018, *arXiv:1812.00984*.

[7] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Comput. Surveys*, vol. 54, no. 1, p. 4, 2021.

[8] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM Conf. Comput. Commun. Security*, 2017, pp. 603–618.

[9] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019, pp. 1–11.

[10] B. Zhao, K. R. Mopuri, and H. Bilen, "iDLG: Improved deep leakage from gradients," 2020, *arXiv:2001.02610*.

[11] H. Jiang *et al.*, "Fly-Navi: A novel indoor navigation system with on-the-fly map generation," *IEEE Trans. Mobile Comput.*, vol. 20, no. 9, pp. 2820–2834, Sep. 2021.

[12] H. Jiang, M. Wang, P. Zhao, Z. Xiao, and S. Dustdar, "A utility-aware general framework with quantifiable privacy preservation for destination prediction in LBSs," *IEEE/ACM Trans. Netw.*, vol. 29, no. 5, pp. 2228–2241, Oct. 2021.

[13] S. Hardy *et al.*, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," 2017, *arXiv:1711.10677*.

[14] S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," in *Proc. IEEE Symp. Comput. Arithmetic*, 2019, p. 198.

[15] Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Scalable and secure logistic regression via homomorphic encryption," IACR Cryptol. ePrint Arch., Lyon, France, Rep. 111/2016, 2016.

[16] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Security Privacy*, 2017, pp. 19–38.

[17] K. A. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM Conf. Comput. Commun. Security*, 2017, pp. 1175–1191.

[18] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in *Proc. Asia Conf. Comput. Commun. Security*, 2018, pp. 1–15.

[19] Z. Xiao *et al.*, "Vehicular task offloading via heat-aware MEC cooperation using game-theoretic method," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 2038–2052, Mar. 2020.

[20] N. Papernot, M. Abadi, Ú. Erlingsson, I. J. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–16.

[21] N. Wang *et al.*, "Collecting and analyzing multidimensional data with local differential privacy," in *Proc. IEEE Int. Conf. Data Eng.*, 2019, pp. 1–12.

[22] Y. Zhao *et al.*, "Local differential privacy-based federated learning for Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8836–8853, Jun. 2021.

[23] J. Li, M. Khodak, S. Caldas, and A. Talwalkar, "Differentially private meta-learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–18.

[24] L. Lyu *et al.*, "Towards distributed privacy-preserving prediction," 2019, *arXiv:1910.11478*.

[25] S. Truex, L. Liu, K. H. Chow, M. E. Gursoy, and W. Wei, "LDP-fed: Federated learning with local differential privacy," 2020, *arXiv:2006.03637*.

[26] L. Sun, J. Qian, X. Chen, and P. S. Yu, "LDP-FL: Practical private aggregation in federated learning with local differential privacy," 2020, *arXiv:2007.15789*.

[27] L. Sun and L. Lyu, "Federated model distillation with noise-free differential privacy," 2020, *arXiv:2009.05537*.

[28] D. Liu, Z. Cao, M. Hou, H. Rong, and H. Jiang, "Pushing the limits of transmission concurrency for low power wireless networks," *ACM Trans. Sens. Netw.*, vol. 16, no. 4, p. 40, 2020.

[29] T. Jung, X. Li, and M. Wan, "Collusion-tolerable privacy-preserving sum and product calculation without secure channel," *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 1, pp. 45–57, Jan./Feb. 2015.

[30] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1–10.

[31] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Security Privacy*, 2019, pp. 691–706.

[32] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Security Privacy*, 2019, pp. 739–753.

[33] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "BatchCrypt: Efficient homomorphic encryption for cross-silo federated learning," in *Proc. USENIX Annu. Tech. Conf.*, 2020, pp. 493–506.

[34] H. Fang and Q. Qian, "Privacy preserving machine learning with homomorphic encryption and federated learning," *Future Internet*, vol. 13, no. 4, p. 94, 2021.

[35] T. Stevens, C. Skalka, C. Vincent, J. Ring, S. Clark, and J. Near, "Efficient differentially private secure aggregation for federated learning via hardness of learning with errors," 2021, *arXiv:2112.06872*.

[36] H. Zhu, R. Wang, Y. Jin, K. Liang, and J. Ning, "Distributed additive encryption and quantization for privacy preserving federated deep learning," *Neurocomputing*, vol. 463, pp. 309–327, Nov. 2021.

[37] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *Proc. 31st Annu. Cryptol. Conf.*, Santa Barbara, CA, USA, Aug. 2011, pp. 505–524.

[38] J. Ma, S. Naas, S. Sigg, and X. Lyu, "Privacy-preserving federated learning based on multi-key homomorphic encryption," 2021, *arXiv:2104.06824*.

[39] G. Pereteanu, A. Alansary, and J. Passerat-Palmbach, "Split HE: Fast secure inference combining split learning and homomorphic encryption," 2022, *arXiv:2202.13351*.

[40] J. Suh and T. Tanaka, "Encrypted value iteration and temporal difference learning over Leveled homomorphic encryption," 2021, *arXiv:2103.11065*.

[41] C. Liu, S. Chakraborty, and D. C. Verma, "Secure model fusion for distributed learning using partial homomorphic encryption," in *Proc. Policy-Based Auton. Data Governance*, 2018, pp. 154–179.

[42] H. Jiang, Z. Xiao, Z. Li, J. Xu, F. Zeng, and D. Wang, "An energy-efficient framework for Internet of Things underlaying heterogeneous small cell networks," *IEEE Trans. Mobile Comput.*, vol. 21, no. 1, pp. 31–43, Jan. 2022.

[43] H. Jiang, Y. Zhang, Z. Xiao, P. Zhao, and A. Iyengar, "An empirical study of travel behavior using private car trajectory data," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 53–64, Jan.–Mar. 2021.
[44] J. H. Cheon, A. Kim, M. Kim, and Y. S. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2017, pp. 409–437.

Ping Zhao (Member, IEEE) received the B.E. degree from Tianjin University of Science and Technology, Tianjin, China, in 2013, and the Ph.D. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, China, in 2018.

She is currently an Associate Professor with the College of Information Science and Technology, Donghua University, Shanghai, China. Her research interests are in the area of data privacy protection in mobile networks, edge computing, and federated learning.

Zhikui Cao received the B.S. degree from the College of Post and Telecommunication, Wuhan Institute of Technology, Wuhan, China, in 2018, and the master's degree from the School of Information Science and Technology, Donghua University, Shanghai, China, in 2022.

His research interests include data privacy preserving in mobile systems.

Jin Jiang received the B.S. degree from the School of Information Science and Technology, Donghua University, Shanghai, China, in 2019, where he is currently pursuing the master's degree with the School of Information Science and Technology.

His research interests include mobile and wireless networks, especially privacy preserving in mobile systems.

Fei Gao (Member, IEEE) received the Ph.D. degree in cryptography from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2007.

He is currently a Professor with the Institute of Network Technology, BUPT. His research interests include cryptography, information security, and quantum algorithm.

Prof. Gao is a member of the Chinese Association for Cryptologic Research.