Privacy-Preserving Asynchronous Federated Learning under Non-IID settings

Yinbin Miao, Da Kuang, Xinghua Li, Shujiang Xu, Hongwei Li, *Fellow, IEEE*, Kim-Kwang Raymond Choo, *Senior Member, IEEE*, and Robert H. Deng, *Fellow, IEEE*

***Abstract*—To address the challenges posed by data silos and heterogeneity in distributed machine learning, privacy-preserving asynchronous Federated Learning (FL) has been extensively explored in academic and industrial fields. However, existing privacy-preserving asynchronous FL schemes still suffer from the problem of low model accuracy caused by inconsistency between delayed model updates and current model updates, and even cannot adapt well to Non-Independent and Identically Distributed (Non-IID) settings. To address these issues, we propose a Privacy-preserving Asynchronous Federated Learning based on the alternating direction multiplier method (PAFed), which is able to achieve high-accuracy models in Non-IID settings. Specifically, we utilize vector projection techniques to correct the inconsistency between delayed model updates and current model updates, thereby reducing the impact of delayed model updates on the aggregation of current model updates. Additionally, we employ an optimization method based on alternating direction multipliers to adapt the Non-IID settings to further enhance the global model accuracy. Finally, through extensive experiments, we demonstrate that our scheme improves the model accuracy by up to 12.53% when compared with current state-of-the-art solution FedADMM.**

***Index Terms*—Federated learning, Privacy-preserving, asynchronous, Non-IID.**

I. Introduction

FEDERATED Learning (FL) [1], [2] is an emerging distributed learning paradigm, which solves the problem of data islands by jointly learning the global model through distributed client devices. However, even if FL does not leak client-side local data, the model may still be vulnerable to inference attacks. Moreover, compared with traditional central learning, FL has many client groups and limited communication resources, which often causes asynchronous update, resulting in a decrease in model accuracy [3]. Therefore, privacy-preserving asynchronous FL has been extensively explored in both academic and industrial fields. However, existing privacy-preserving asynchronous FL solutions still face many problems in practical applications.

The first problem is that existing privacy-preserving asynchronous FL still has the problem of conflicts between delayed model updates and current model updates, resulting in a decrease in global model accuracy. Specifically, in an asynchronous environment, the server may receive model updates from different clients in different periods when performing parameter aggregation. As the number of delayed rounds increases, there can be a significant difference in the direction of model updates for delayed clients, thereby affecting the global model accuracy. Existing asynchronous FL solutions [4]–[10] have supported asynchronous aggregation to effectively alleviate the problems of asynchronous aggregation. In particular, Miao *et al.* [11] proposed a method assigning different weight values to delayed model updates based on the timestamps uploaded by clients to reduce the impact of outdated models on the learning process. However, the fundamental issue of reduced model accuracy is that inconsistent directions between delayed model updates and current model updates remains unresolved [12]. Specifically, when there is a conflict between the delayed model update and the current model update and the size of the two model updates is significantly different, the aggregated gradient will be disproportionately affected by the larger gradient value, greatly deviating from the global optimal value. Thus, addressing the inconsistent direction of delayed model updates is crucial for alleviating conflicts between delayed and current updates and ultimately enhancing the overall efficacy of asynchronous FL systems.

The second problem is that the existing privacy-preserving asynchronous FL still cannot adapt well to Non-IID settings, thereby affecting the model accuracy. In actual scenarios, a large number of client groups causes local data distribution to often appear Non-IID settings, and there may be large differences in local model parameters between clients. And the differences between client model updates may be more prominent in an asynchronous environment, resulting in reduced accuracy of the aggregated global model. Personalized FL [13], [14] can train a model for each participant that adapts

Y. Miao is with the School of Cyber Engineering, Xidian University, Xi'an 710071, China, and also with the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Qilu University of Technology (Shandong Academy of Sciences), Jinan, 250014, China. E-mail: ybmiao@xidian.edu.cn.

D. Kuang is with the School of Cyber Engineering, Xidian University, Xi'an 710071, China. E-mail: DaKuang@stu.xidian.edu.cn.

X. Li is with the State Key Laboratory of Integrated Service Networks, School of Cyber Engineering, Xidian University, Xi'an 710071, China, and also with Engineering Research Center of Big data Security, Ministry of Education, Xi'an 710071, China. Email: xhli1@mail.xidian.edu.cn.

S. Xu is with the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology (Shandong Academy of Sciences), Jinan, 250014, China, and also with the Shandong Provincial Key Laboratory of Computer Networks, Shandong Fundamental Research Center for Computer Science, Jinan, 250014, China. E-mail: xushj@sdas.org.

H. Li is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610051, China. E-mail: hongweili@uestc.edu.cn.

K.-K. R. Choo is with the Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249 USA. Email: raymond.choo@fulbrightmail.org.

R. H. Deng is with the School of Information Systems, Singapore Management University, 80 Stamford Road 178902, Singapore. Email: robertdeng@smu.edu.sg.

to its specific data distribution, thereby adapting to the characteristics of inconsistent data distribution and improving model accuracy. Other solutions [15]–[17] have also been proposed to address data heterogeneity by optimizing the loss function, but these methods do not support asynchronous environments. In addition, current solutions [18] combine optimization loss functions and methods supporting asynchronous aggregation to address issues caused by system heterogeneity. Nevertheless, these schemes fail to effectively address the conflict between delayed and current model updates in asynchronous systems. Therefore, it is necessary to adapt well to inconsistent data distribution in asynchronous environments.

Homomorphic encryption (HE), secure multi-party computation (SMC) and differential privacy (DP) are widely utilized in asynchronous FL. SMC-based asynchronous FL [19] enables secure aggregation, but it incurs high communication costs due to multiple interactions between clients. DP-based asynchronous FL [20]–[22] forestalls attackers from obtaining local private information of the client by introducing noise, but this approach may reduce model accuracy due to the noise injection. Compared to the aforementioned encryption mechanisms [23]–[26], employing HE encryption is effectively sufficient to reduce communication overhead and prevent model accuracy loss. Nevertheless, most HE-based FL solutions adopt a multi-client single-key privacy protection mechanism, which are not secure because obtaining the key of a certain client will harm the interests of all other clients.

Through the above discussion, we propose a high-precision privacy-preserving asynchronous FL in a Non-IID settings (PAFed). To address the conflict between delayed model updates and current model updates, we utilize the gradient direction to identify delayed updates and employ vector projection technology to adjust the direction of delayed updates, thereby bringing them closer to the current average update. Second, we optimize the loss function based on the alternating direction multiplier method to accommodate data heterogeneity. Through these methods, we have effectively solved the problems of low accuracy of global model in asynchronous FL under Non-IID settings. Additionally, we adopt a multi-key privacy protection framework to effectively prevent local data leakage and address the issue of low privacy security caused by single-key HE mechanisms.

Our contributions can be summarized as follows:

1) We propose a high-precision asynchronous FL, which effectively resolves the conflict between delayed model updates and current model updates through the normal plane projection technique, and adopt the alternating direction multiplier method to solve the problem of inconsistent data distribution, thereby improving the accuracy of global model.
2) We propose a dynamic classification mechanism under ciphertext to finely classify and process delayed model updates to solve the conflict between delayed model updates and current round model updates, then dynamically adjust multi-level learning rates to adapt to non-IID settings and asynchronous environments to further improve model performance. In addition, we propose a multi-key HE-based privacy protection framework that can effectively protect client privacy and prevent local data leakage.
3) Formal analysis shows that our scheme has no data leakage during the aggregation process, and a large number of experiments demonstrate that the model accuracy of our scheme is higher than that of FedADMM by up to 12.53%.

The rest of this paper is organized as follows: Section II reviews recent work on APFed, and Section III describes preliminaries. Section IV introduces problem formulationof APFed. Section VI and Section VII analyze the discussion of security and performance respectively. Section VIII concludes the paper.

II. Related Work

To prevent the accuracy of global model from being reduced in an asynchronous environment, Xie *et al.* [5] proposed to dynamically adjust the weight of each client based on the time difference of parameter upload for asynchronous aggregation. Asynchronous updates can reduce the overhead of synchronization between clients, minimize communication costs, and enhance system throughput. However, the weighted aggregation strategy adopted cannot mitigate the model accuracy degradation caused by discrepancies between delayed model updates and current model updates. To further mitigate the impact of delayed model update inconsistencies, Wang *et al.* [12] proposed a federated fair averaging algorithm. They employ cosine similarity to detect potential conflicts between client gradients and then rectify the conflicting gradients. Although this method can alleviate gradient conflict issues to a certain extent, it requires modification of each conflicting gradient, significantly increasing high computational overhead.

Furthermore, all the above solutions do not support Non-IID settings, and cannot effectively solve the problems caused by inconsistent data distribution. Sattler *et al.* [13] proposed a cluster-based FL scheme, which groups clients with similar data into the same cluster by calculating the cosine value, so as to reduce the impact of Non-IID data and improve model accuracy. However, this approach necessitates supplementary clustering processes, thereby escalating computational and communication demands. Li *et al.* [17] counteracted the divergence caused by data heterogeneity by adding proximal terms to the locally trained loss function. While this technique avoids augmenting computational and communication burdens, it exhibits pronounced sensitivity to parameter selection. Furthermore, Karimireddy *et al.* [27] corrected client-drift in their client-side local updates by using a variance reduction. Despite its efficacy, this approach incurs a substantial increase in communication costs and lacks the flexibility to handle device diversity. Kong *et al.* [18] introduced a FedAdmm algorithm using dual variables to store the changes in local gradients, which allows for automatic adaptation to data heterogeneity and device heterogeneity by tolerating variable workloads performed by clients. Although this method accommodates asynchronous updates, it does not effectively address the model accuracy issues caused by the inconsistency between delayed and current model updates. Additionally, all

the mentioned approaches expose model updates directly to the server during the aggregation process, potentially leading to privacy risks.

To solve the above problems, Lu *et al.* [28] proposed a dual-weighted approach based on sample size and timestamps in asynchronous FL, employing adaptive threshold compression on gradients to reduce the effective information an attacker can obtain. While this method diminishes the gradient information available to an attacker, it still poses a risk of partial information leakage. Gu *et al.* [19] introduced an asynchronous decentralized federated stochastic gradient algorithm, which uses a tree-structured communication method for asynchronous updates to tackle the asynchrony issue. Although this approach effectively prevents local data leakage, the communication overhead becomes significantly costly as the number of communications between clients increases due to the tree-based communication strategy. Yu *et al.* [20] introduced a stochastic peer-to-peer update mechanism to adapt to an asynchronous environment, employing DP during the gradient descent local training process to protect client privacy. This method effectively safeguards client privacy, but it compromises model accuracy. Miao *et al.* [11] proposed a time-weighted asynchronous PPFL. Specifically, this method assigns weights to outdated model updates during the asynchronous aggregation process based on the number of delayed rounds, allowing for the integration of stale models. However, the method of allocating weights solely based on the number of model update delay rounds in asynchronous aggregation is not sufficiently accurate, as the delay rounds do not accurately reflect the conflict between delayed model updates and current model updates. In addition, all clients in this solution use the same key to locally encrypt the local model, which incurs additional involves additional privacy risks. That is, when the key of one client is leaked, the data privacy of all other clients will face the risk of privacy leakage. Finally, we give a summary of the above schemes in TABLE I.

TABLE I
SUMMARY OF SCHEMES

Schemes	Support IID or Non-IID	asynchronous or synchronous	Core Technology	Privacy-preserving method
[5]	IID	Asynchronous	Async average	✗
[10]	IID	Asynchronous	consensus mechanism	DP
[13]	Non-IID	Synchronous	Clustering	✗
[17]	Non-IID	Synchronous	proximal terms	✗
[19]	IID	Asynchronous	Tree structure	MPC
[20]	IID	Asynchronous	Peer-to-peer	DP
[12]	IID	Asynchronous	Adust updates	✗
[28]	IID	Asynchronous	dual-weights	✗
[27]	Non-IID	Synchronous	control variables	✗
[18]	Non-IID	Asynchronous	Primal-dual	✗
[11]	IID	Asynchronous	time-weighting	HE
PAFed	Non-IID	Asynchronous	Projection&ADMM	HE

III. PRELIMINARIES

In this section, we introduce the goals of FL, and then introduce the principles of the Alternating Direction Method of Multipliers (ADMM) and Cheon-Kim-Kim-Song (CKKS) homomorphic encryption.

A. Federated Learning (FL)

FL aims to find the global model w that minimize the weighted average loss of all clients.

$$\min_{\theta\in\mathbb{R}^d} F(\mathbf{w}) = \sum_{i=1}^{m} \frac{n_i}{n} f_i(\mathbf{w}), \tag{1}$$

where $f_i(\mathbf{w}) = \frac{1}{n_i}\sum_{\xi_k\in D_i}\mathcal{L}(\mathbf{w},\xi_k)$, D_i denotes the local dataset of the i-th client, ξ_k is the k-th sample in D_i, n_i is the size of D_i and $n=\sum_{i=1}^{m} n_i$, $\mathcal{L}(\mathbf{w},\xi_k)$ is the global loss function of training model.

B. Alternating Direction Method of Multipliers

Alternating Direction Method of Multipliers (ADMM) [29], [30] is a distributed optimization algorithm derived from the combination of the Douglas-Rachford and Bregman algorithms, primarily utilized to solve constrained optimization problems by decomposing complex challenges into manageable sub-problems for distributed computing and parallel processing.

Suppose we need to solve an optimization problem shown by Eq. 2:

$$\min f(\mathbf{x}) + g(\mathbf{z}), \quad s.t., \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}=\mathbf{c}, \tag{2}$$

where $\mathbf{x}\in\mathbb{R}^n, \mathbf{z}\in\mathbb{R}^m$ are optimization vectors, $f(\cdot)$, $g(\cdot)$ are convex functions, $\mathbf{A}\in\mathbb{R}^{p\times n}, \mathbf{B}\in\mathbb{R}^{p\times m}$ are known matrices and $\mathbf{c}\in\mathbb{R}^p$ is a known vector. To solve such convex optimization problems, the augmented Lagrangian function is defined:

$$\begin{aligned}\mathcal{L}(\mathbf{x},\mathbf{y},\mathbf{z}) =& f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^\top(\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}-\mathbf{c}) + \\ & \frac{\rho}{2}\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}-\mathbf{c}\|^2,\end{aligned} \tag{3}$$

where $\mathcal{L}(\cdot)$ is loss function, ρ is the parameter and $\mathbf{y}$ is the Lagrange multiplier vector. The goal is to find $\mathbf{x}$ and $\mathbf{z}$ that minimize the objective function. In the k-th iteration, ADMM starts with fixed $\mathbf{z}^k$ and $\mathbf{y}^k$ and updates $\mathbf{x}^{k+1}$ using Eq. 4:

$$\mathbf{x}^{k+1} = \arg\min_x \left(f(\mathbf{x}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{z}^k-\mathbf{c}+\mathbf{y}^k\right\|^2\right), \tag{4}$$

Next, ADMM updates $\mathbf{z}^{k+1}$ in a similar manner:

$$\mathbf{z}^{k+1} = \arg\min_z \left(g(\mathbf{z}) + \frac{\rho}{2}\left\|\mathbf{A}\mathbf{x}^{k+1}+\mathbf{B}\mathbf{z}^k-\mathbf{c}+\mathbf{y}^k\right\|^2\right). \tag{5}$$

Finally, $\mathbf{y}^{k+1}$ is updated as follows:

$$\mathbf{y}^{k+1} = \mathbf{y}^k + \mathbf{A}\mathbf{x}^{k+1} + \mathbf{B}\mathbf{z}^{k+1} - \mathbf{c}. \tag{6}$$

C. Full Homomorphic Encryption (CKKS)

CKKS [31] is a fully HE scheme based on approximate minimal coding, which allows addition and multiplication operations on encrypted data. CKKS achieves high encryption efficiency and security while ensuring calculation accuracy. We introduce the key components and operations of CKKS in detail, including Initialization, Encoding,

Decoding, KeyGeneration, Encryption, Decryption, Addition, and Multiplication.

CKKS.Initialization$(1^\lambda) \rightarrow (N, p, q)$: Given a security parameter λ, generate modulus N, p and q satisfying safety level λ, where N is the degree of the polynomial ring.

CKKS.Encoding$(\mathbf{z}) \rightarrow \Phi(x)$: Given a vector of complex numbers $\mathbf{z} \in \mathbb{C}^{N/2}$, output an integer polynomial $\Phi(x) \in \frac{\mathbf{z}_q[x]}{x^{N+1}}$, where $\mathbf{z}_q$ is the integer ring modulo q, and x is a variable in the polynomial ring.

CKKS.Decoding$(\Phi(x) \rightarrow \mathbf{z}$: Given the polynomial $\Phi(x)$ and the root ξ of the circular polynomial $\varphi_m = x^N + 1$, output a complex vector $\mathbf{z} = \{z_1, z_2, ..., z_{N-1}\}$, where $z_i = \Phi(\xi^i) + e_i$ and e_i represents the error term.

CKKS.KeyGen$(\chi_s, \chi_e) \rightarrow (p_k, s_k)$: Given $s \leftarrow \chi_s$, $e, e' \leftarrow \chi_e$, $a \leftarrow \mathbb{R}_q$ and $a' \leftarrow \mathbb{R}_{pq}$, output private key $sk \leftarrow (1, s)$, public key $pk \leftarrow (b, a) \in \mathbb{R}_q^2$, compute key $evk \leftarrow (b', a') \in \mathbb{R}_{pq}^2$, where χ_s is private key distribution, χ_e is error distribution, χ_r is random distribution, $b = -a \cdot s + e \bmod q$, $b' = -a' \cdot s + e' + P \cdot s^2 \bmod p \cdot q$.

CKKS.Enc$_{pk}(m) \rightarrow c$: Given the plaintext polynomial $m \in \mathbb{R}$, output the ciphertext $c \leftarrow r \cdot pk + (m + e_0, e_1) \bmod q$, where $r \leftarrow \chi_r$, $e_0, e_1 \leftarrow \chi_e$.

CKKS.Dec$_{sk}(c) \rightarrow m$: Given a ciphertext $c = (c_0, c_1) \in \mathbb{R}_q^2$ and private key sk, output its plaintext $m' \leftarrow c_0 + c_1 \cdot s \bmod q$.

CKKS.Add$(c, c') \rightarrow c_{add}$: Given two ciphertexts $c = (c_0, c_1) \in \mathbb{R}_q^2$ and $c' = (c_0', c_1') \in \mathbb{R}_q^2$, output the sum via $c_{add} \leftarrow (c + c') \bmod q$.

CKKS.Mult$_{evk}(c, c') \rightarrow c_{mult}$: Given two ciphertexts $c = (c_0, c_1)$ and $c' = (c_0', c_1')$, compute a triad as $(d_0, d_1, d_2) = (c_0, c_0', c_0c_1' + c_0'c_1, c_1c_1')$mod q, output the product as $c_{mult} \leftarrow (d_0, d_1) + \lfloor P^{-1} \cdot d_2 \cdot evk \rceil \bmod q$.

IV. Problem Formulation

In this section, we show the system model, problem definition and design goals, respectively.

A. System Model

In this section, we consider asynchronous update environments. The system model of PAFed consists of servers, online clients, delayed clients and key distribution center (KDC). The role of each entity is as follows:

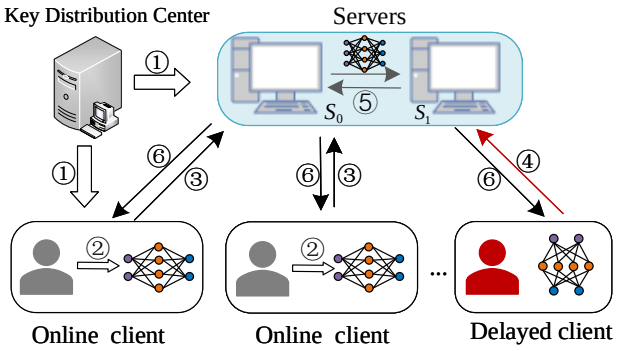


Fig. 1. System model

- KDC: KDC assigns public/secret key pairs to server S_1 and all clients.
- Servers: In the asynchronous environment, the server S_0 tracks the status of each client, dynamically adjusts the aggregation strategy, and then transmits the result to server S_1 for key conversion, finally returns aggregation result to clients.
- Online Clients: In each round, online clients receive the global model and train it locally, then upload updates to S_0.
- Delayed clients: Updates uploaded by delayed clients are not received by S_0 immediately due to network delays and other reasons, but will be received by S_0 in several delayed rounds.

Specifically, KDC assigns public/secret key pairs to each client and S_1 (Step ①). In the τ-th communication round, the online client C_i updates the local model $\mathbf{w}_i^\tau$ and encrypts it locally (Step ②), then uploads the encrypted model update $[\![\mathbf{w}_i^\tau]\!]$ to server S_0 (step ③). In addition to receiving the update sent by the online client, S_0 may also receive the delayed model update uploaded by the client $C_i^{\tau-ts}$ of the $\tau-ts$ round (Step ④). Then, the server S_0 performs secure aggregation, and then sends the aggregation result to S_1 for key conversion (Step ⑤). Finally, S_0 sends the global model $[\![\mathbf{w}^{\tau+1}]\!]$ to clients (Step ⑥).

B. Threat Model

In our system, we assume that KDC is honest and servers and clients are honest-but-curious. Specifically, all entities honestly follow the initially set learning protocol, but the servers will infer the client's private information through the client's local model, and the client may also curiously infer other clients' private information through the global model. Furthermore, the asynchronous environment of the system involves delayed clients. The potential threats caused by above entities are shown as follows:

- *Reduction of model accuracy:* Since the dataset of client presents a Non-IID distribution, and the existence of the delayed client will introduce delay in the communication process, therefore the accuracy of global model will eventually be greatly degraded.
- *Leakage of data privacy:* The central server is honest but curious, there may be a risk that the server infers private information from client local data.

C. Problem Definition

Given n clients $\mathcal{C} = \{C_1, C_2, ..., C_n\}$ with corresponding datasets $\mathcal{D} = \{D_1, D_2, ..., D_n\}$, all clients jointly update the global model $\mathbf{w}^\tau$. Considering the asynchronous environment, we assume that the first d clients in round τ may encounter different levels of noise, delay or loss. We denote delayed clients as $\mathcal{C}_d^\tau = \{C_1, C_2, ..., C_d\}$ and online clients as $\mathcal{C}_o^\tau = \{C_{d+1}, C_{d+2}, ..., C_n\}$ respectively. The server S_0 needs to wait for the delayed clients $\mathcal{C}_d^\tau$ during synchronization aggregation, causing system lag or bottleneck. To solve the problem of synchronous aggregation, S_0 does not need to wait for $\mathcal{C}_d^\tau$ but directly uses the model updates $\nabla W_o^\tau = \{\nabla \mathbf{w}_{d+1}^\tau, \nabla \mathbf{w}_{d+2}^\tau, ..., \nabla \mathbf{w}_n^\tau\}$ of

$\mathcal{C}_o^\tau$ for aggregation, and discards the delayed model updates $\nabla W_d^\tau = \{\nabla \mathbf{w}_1^\tau, \nabla \mathbf{w}_2^\tau, ..., \nabla \mathbf{w}_d^\tau\}$ of $\mathcal{C}_d^\tau$. However, this will result in the loss of effective information and affect the model accuracy.

To solve this problem, S_0 adopts an asynchronous aggregation strategy based on timestamp. Specifically, in the τ-th round, S_0 can receive both the delayed model updates $\nabla W_d^{\tau_j}$ from the delayed clients $\mathcal{C}_d^{\tau_j}$ in τ_j round and the model updates ∇W_o^τ from the online clients $\mathcal{C}_0^\tau$ in current round. Then, S_0 assigns a low weight value to $\nabla W_d^{\tau_j}$ according to the delay round number $\tau - \tau_j$ for weighted aggregation to make up for the missing gradient. However, the asynchronous aggregation strategy based on timestamp also has flaws. When $\tau - \tau_j$ is too large, the versions of $\nabla W_d^{\tau_j}$ and ∇W_o^τ will be quite different, resulting in possible large deviations and conflicts between the gradient directions of $\nabla W_d^{\tau_j}$ and ∇W_o^τ. We can define a conflict metric function $R(\nabla \mathbf{w}_i^\tau, \nabla \mathbf{w}_j^{\tau_j})$ to quantify the conflict between two vectors, *i.e.*, $R(\nabla \mathbf{w}_i^\tau, \nabla \mathbf{w}_j^{\tau_j}) = \nabla \mathbf{w}_i^\tau \cdot \nabla \mathbf{w}_j^{\tau_j}$, where $\nabla \mathbf{w}_i^\tau \in \nabla W_o^\tau$, $\nabla \mathbf{w}_j^{\tau_j} \in \nabla W_d^{\tau_j}$. Based on the above definition, we can further give the problem definition:

Definition 1 *In the τ-th communication round, the update $\nabla \mathbf{w}_i^\tau \in \mathbb{R}^d$ of online client C_i^τ conflicts with the update $\nabla \mathbf{w}_j^{\tau_j} \in \mathbb{R}^d$ of delayed client $C_j^{\tau_j}$, if $R(\nabla \mathbf{w}_i^\tau, \nabla \mathbf{w}_j^{\tau_j}) < 0$.*

Beside, the data of different clients C_i may present different distribution characteristics under Non-IID settings. For example, D_i only contains samples of a certain category, while $D_{j \neq i}$ may contain samples of other categories. This means that local model updates may be optimized for the data distribution on a specific device rather than the global data distribution. Moreover, the local model $\mathbf{w}_i^\tau$ trained by each client C_i may be very different, resulting in a decrease in the performance of $\mathbf{w}^\tau$ after the aggregation of the central server. As shown in Fig. 2, $\mathbf{w}_c^\tau$ refers to the model trained on all data using SGD, and $\mathbf{w}_f^\tau$ refers to the model after the central server in fedavg aggregates $\mathbf{w}_i^\tau$ and $\mathbf{w}_j^\tau$.

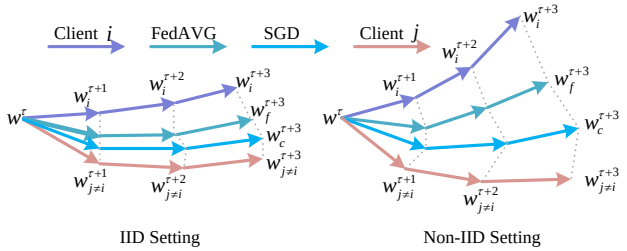


Fig. 2. Illustration of the weight divergence for FL with IID and Non-IID data.

In addition, in a single-key system, all clients share a key pair $K = (pk, sk)$. If the private key sk of a client C_i is leaked, any client $C_{j \neq i}$ may suffer from the risk of information leakage, as shown in Eq.7, where $\mathbf{w}_j'$ is the decrypted model containing local information of client j.

$$\forall j \neq i, \mathbf{w}_j' = \mathsf{Dec}(\mathsf{Enc}(\mathbf{w}_j, sk), pk). \tag{7}$$

D. Design Goals

Based on the above discussion, our scheme needs to design a FL that ensures both security and high accuracy in Non-IID settings and asynchronous update scenarios. The design goals are as follows:

- *Model accuracy:* Our scheme should achieve high model accuracy under specific Non-IID settings and asynchronous update scenarios. Specifically, our scheme should meet or outperform FedAvg's model performance, whether in a client environment with Non-IID settings or in asynchronous environment.
- *Privacy:* During the entire FL period, neither server S_0, S_1 nor the client should infer any valid information about other clients except within the scope of their own permissions.

V. Proposed Scheme

In this section, we describe how our scheme works. First, we give the technical overview of the scheme and then elaborate its concrete construction.

A. Technical Overview

Traditional FL cannot adapt well to Non-IID settings in an asynchronous environment, resulting in reduced model accuracy. To solve this problem, we introduce the ADMM algorithm in an asynchronous environment to optimize the loss function, as follows:

$$\begin{aligned}\mathcal{L}_i\left(\mathbf{w}_i^{\tau_r}, \mathbf{y}_i^{\tau_r}, \mathbf{w}^\tau\right) = f_i\left(\mathbf{w}_i^{\tau_r}\right) + \left(\mathbf{y}_i^{\tau_r}\right)^\top \left(\mathbf{w}_i^{\tau_r} - \mathbf{w}^\tau\right) \\ + \frac{\rho}{2}\|\mathbf{w}_i^{\tau_r} - \mathbf{w}^\tau\|^2,\end{aligned} \tag{8}$$

where $\mathbf{y}_i^{\tau_r} \in \mathbb{R}^d$ is the local dual variable held by client C_i, τ refers to the current round, τ_r refers to the most recent online round of the client, and $\rho > 0$ is the coefficient of the quadratic term. Specifically, we first update the local model $\mathbf{w}_i^{\tau+1}$ of the τ-th round of client C_i as shown in Eq. 9,

$$\mathbf{w}_i^{\tau+1} \approx \arg\min_{w_i^\tau} \mathcal{L}\left(\mathbf{w}_i^{\tau_r}, \mathbf{y}_i^{\tau_r}, \mathbf{w}^\tau\right), \tag{9}$$

and then update $\mathbf{y}_i^{\tau+1}$ as follow:

$$\mathbf{y}_i^{\tau+1} = \mathbf{y}_i^{\tau_r} + \rho\left(\mathbf{w}_i^{\tau+1} - \mathbf{w}^{\tau_r}\right), \tag{10}$$

finally, the server S_0 updates global model $\mathbf{w}^{\tau+1}$ as follow:

$$\mathbf{w}^{\tau+1} = \mathbf{w}^\tau + \frac{\eta}{|\mathcal{C}^\tau|} \sum_{i \in \mathcal{C}^\tau} \left(\left(\mathbf{w}_i^{\tau+1} - \mathbf{w}_i^{\tau_r}\right) + \frac{1}{\rho}\left(\mathbf{y}_i^{\tau+1} - \mathbf{y}_i^{\tau_r}\right)\right), \tag{11}$$

where $\mathcal{C}^\tau$ is the set of clients selected to participate in training in round τ, and η is the global learning rate.

In addition, to better adapt to the asynchronous environment and solve the potential conflict issue between the current round of model update $\nabla \mathbf{w}_i^\tau \in \nabla W_o^\tau$ and the delayed model update $\nabla \mathbf{w}_{j \neq i}^{\tau_j} \in \nabla W_d^\tau$ (*i.e.*, $\nabla \mathbf{w}_i^\tau \cdot \nabla \mathbf{w}_{j \neq i}^{\tau_j} < 0$). We use the projection method to correct the direction of the delayed average update so that it is close to the current average update. Specifically, the server S_0 receives the updates $\nabla \mathbf{w}_i^\tau$ sent by the current round of online clients and calculates the average value $\nabla \mathbf{w}_m^\tau$. Then, S_0 records all delayed updates $\nabla W_d^\tau = (\nabla \mathbf{w}_0^{\tau_0}, ..., \nabla \mathbf{w}_d^{\tau_d})$ received in this round. For delayed updates, we adopt the idea of multi-layer classification processing. First, S_0 judges whether $\nabla \mathbf{w}_j^{\tau_j}$ conflicts with $\nabla \mathbf{w}_m^\tau$ (*i.e.*,

$\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j} > 0$). Then, S_0 calculates $\nabla \mathbf{w}_{m_1}^\tau$ by averaging the model updates $\nabla \mathbf{w}_j^{\tau_j}$ satisfying $\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j} < 0$. Similarly, S_0 first calculates the average of all $\nabla \mathbf{w}_j^{\tau_j}$ satisfying $\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j} > 0$, denoted as $\nabla \mathbf{w}_{m_2}^\tau$, and then projects $\nabla \mathbf{w}_{m_2}^\tau$ onto the normal plane of $\nabla \mathbf{w}_m^\tau$ to correct the direction of $\nabla \mathbf{w}_{m_2}^\tau$, as shown in Eq. 12.

$$\nabla \mathbf{w}_{m_2}^\tau \leftarrow \nabla \mathbf{w}_{m_2}^\tau - \frac{\nabla \mathbf{w}_{m_2}^\tau \cdot \nabla \mathbf{w}_m^\tau}{\left\| \nabla \mathbf{w}_m^\tau \right\|^2} \nabla \mathbf{w}_m^\tau. \tag{12}$$

As shown in Fig. 3, we project the delayed model updates $\nabla \mathbf{w}_i^\tau$ and $\nabla \mathbf{w}_j^\tau$ onto the normal plane of the average update $\nabla \mathbf{w}^\tau$ respectively, so that the corrected model updates $\nabla \mathbf{w}_i^{\tau'}$, $\nabla \mathbf{w}_j^{\tau'}$ become more consistent with the direction of $\nabla \mathbf{w}^\tau$. In addition, Fig. 3 shows that the direction of global model $\nabla \mathbf{w}^{\tau+1'}$ aggregated from $\nabla \mathbf{w}_i^{\tau'}$ and $\nabla \mathbf{w}_j^{\tau'}$ is more beneficial for the performance of the global model when comparing with $\nabla \mathbf{w}^{\tau+1}$ aggregated from $\nabla \mathbf{w}_i^\tau$ and $\nabla \mathbf{w}_j^\tau$.

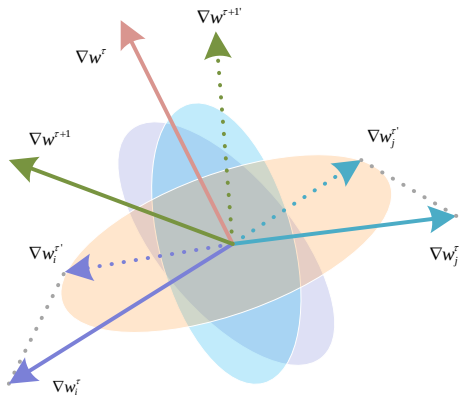


Fig. 3. Schematic diagram of the result of projecting delayed model updates to the normal plane

To protect the privacy of C_i, we use CKKS to encrypt the model update uploaded by C_i and ensure that servers perform aggregations under the ciphertext, thus preventing local data leakage. Before encrypting the model updates, we first need to quantify the model updates and classify the models, and then calculate the average of each category. Since CKKS supports additive homomorphism and multiplicative homomorphism, we can simply calculate the average value $[\![\nabla \mathbf{w}_m^\tau]\!]$. However, it should be noted that CKKS cannot directly support the operation of classification processing, that is, it cannot directly determine whether there is a conflict between $\nabla \mathbf{w}_j^{\tau_j}$ and $\nabla \mathbf{w}_m^\tau$ under the ciphertext, because $[\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!]$ is a ciphertext value and cannot be directly decrypted by servers.

To solve this problem, we introduce the algorithm $Min(\cdot,\cdot)$ [32] for computing the minimum value of two ciphertexts, as shown in Algorithm 1. Given two ciphertexts $[\![a]\!]$ and $[\![b]\!]$ encrypted by CKKS, $Min(\cdot,\cdot)$ returns a ciphertext whose decrypted value approximates to the value of a when $a \le b$, otherwise $Min(\cdot,\cdot)$ returns a ciphertext whose decrypted value approximates to the value of b, where $a, b \in [0,1]$.

To achieve the classification of delayed model updates and then average them, we should determine the relationship between $\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j}$ and 0 under the ciphertext based on

Algorithm 1 *Min* $(\cdot,\cdot)$

Input: $[\![a]\!]$, $[\![b]\!]$, where $a, b \in [0,1]$.
Output: a ciphertext that approximate to the value $\mathrm{Min}(a,b)$.
1: Initial d, $[\![x]\!] \leftarrow ([\![a]\!] + [\![b]\!])/2$, $[\![y]\!] \leftarrow ([\![a]\!] - [\![b]\!])/2$;
2: $[\![m_0]\!] \leftarrow [\![y]\!]^2$, $[\![n_0]\!] \leftarrow [\![y]\!]^2 - 1$;
3: **for** $i = 0, 1, ..., d-1$ **do**
4: $\quad [\![m_{i+1}]\!] \leftarrow [\![m_i]\!]\,([\![1]\!] - ([\![n_i]\!]/2))$;
5: $\quad [\![n_{i+1}]\!] \leftarrow [\![n_i]\!]^2\,(([\![n_i]\!] - [\![3]\!])/4)$;
6: **end for**
7: **return** $[\![x]\!] - [\![m_d]\!]$.

Algorithm 2 ClientUpdate

Input: Local learning rate η, local epoch number E_i, batches B, $\mathbf{w}_i^{\tau_r}$, $\mathbf{y}_i^{\tau_r}$.
Output: Local model $\mathbf{w}_i^\tau + 1$, $\mathbf{y}_i^\tau + 1$.
1: **for** $t = 0, 1, ..., E_i - 1$ **do**
2: $\quad$ **for** batch $b \in B$ **do**
3: $\quad\quad \mathbf{w}_i^{\tau+1} \approx \arg\min_{\mathbf{w}_i^{\tau_r}} \mathcal{L}(\mathbf{w}_i^{\tau_r}, \mathbf{y}_i^{\tau_r}, \mathbf{w}^\tau) = f_i(\mathbf{w}_i^{\tau_r}) + (\mathbf{y}_i^{\tau_r})^\top (\mathbf{w}_i^{\tau_r} - \mathbf{w}^\tau) + \frac{\rho}{2} \|\mathbf{w}_i^{\tau_r} - \mathbf{w}^\tau\|^2$;
4: $\quad$ **end for**
5: **end for**
6: $\mathbf{y}_i^{\tau+1} = \mathbf{y}_i^{\tau_r} + \rho\left(\mathbf{w}_i^{\tau+1} - \mathbf{w}^\tau\right)$;
7: **return** $\mathbf{w}_i^{\tau+1}$, $\mathbf{y}_i^{\tau+1}$.

Algorithm 4. We first normalize $\nabla \mathbf{w}_m^\tau$ and $\nabla \mathbf{w}_j^{\tau_j}$ and then input $\left(\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j} + 1\right)/2$ and $1/2$ into $Min(\cdot,\cdot)$ to indirectly obtain the relationship between the sizes of $\nabla \mathbf{w}_m^\tau \cdot \nabla \mathbf{w}_j^{\tau_j}$ and 0. In addition, we set two variables $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ and initialize them to 0. For each delayed model update $\nabla \mathbf{w}_j^{\tau_j}$, we perform an accumulation on $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ once.

$$\begin{aligned}
[\nabla \mathbf{w}_{m_1}^\tau] &\leftarrow [\nabla \mathbf{w}_{m_1}^\tau] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] \left(\frac{[\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + 1}{2} - [\![x]\!] \right) \\
[\nabla \mathbf{w}_{m_2}^\tau] &\leftarrow [\nabla \mathbf{w}_{m_2}^\tau] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * ([\![x]\!] - [\![1/2]\!]) \\
[\![x]\!] &= Min\left(([\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + [\![1]\!])/2, [\![1/2]\!] \right).
\end{aligned} \tag{13}$$

Finally, S_0 calculates the mean of $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ and then corrects $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ according to the Eq. 14.

$$[\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow [\![\nabla \mathbf{w}_{m_2}^\tau]\!] - \left([\![\nabla \mathbf{w}_{m_2}^\tau]\!] \cdot [\![\nabla \mathbf{w}_m^\tau]\!] \right) [\![\nabla \mathbf{w}_m^\tau]\!]. \tag{14}$$

B. Concrete Construction of PAFed

We divide the privacy framework into five parts, system initialization, local update and processing, delayed model update classification and correction, model aggregation, and key transformation, as shown in Fig. 4. Next, we describe its specific process in Algorithm 3.

System initialization. First, KDC selects a security parameter λ, then initializes like CKKS.Initialization(1^λ), and then assigns a public/secret key pair (pk_i, sk_i) to each client C_i by calling CKKS.KeyGen(χ_s;χ_e). Similarly, KDC also assigns a public/secret key pair (pk_s, sk_s) to S_1.

Local update and processing. The server S_0 initializes the global model $\mathbf{w}^0$, and each client initializes a dual variable $\mathbf{y}_i^0$. For the t-th iteration, S_0 selects the client subset $\mathcal{C}^\tau$ of the τ-th round, and sends the global model $\mathbf{w}^\tau$ to the client $C_i \in \mathcal{C}_0^\tau$, where $\mathcal{C}_o^\tau$ refers to a subset of online clients in $\mathcal{C}^\tau$.

Algorithm 3 Main process

Input: Global learning rate $\alpha_0, \alpha_1, \alpha_2$, number of clients K, number of communication rounds T.
Output: Global model $[\![\mathbf{w}^{\tau+1}]\!]$

1: KDC performs the system initialization process, and assigns different keys pk_s and pk_i to S_1 and each client $\{C_i | 0 \leq i \leq K-1\}$ respectively.
2: Each C_i initials $\mathbf{w}_i^0$, $\mathbf{y}_i^0$ and server S_0 initials $\mathbf{w}^0$;
3: **for** $\tau = 0, ...T-1$ **do**
4: S_0 samples a subset $\mathcal{C}^\tau$ of N clients with the prob and Server send the global model $\mathbf{w}^0$ to $\mathcal{C}^\tau$;//$\mathcal{C}_o^\tau$ is the collection of online clients in $\mathcal{C}^\tau$;
5: **for** each client $C_i \in \mathcal{C}_0^\tau$ in parallel **do**
6: $\mathbf{w}_i^{\tau+1}, \mathbf{y}_i^{\tau+1} \leftarrow ClientUpadate(\mathbf{w}_i^{\tau_r}, i)$;
7: $\nabla \mathbf{w}_i^\tau \leftarrow (\mathbf{w}_i^{\tau+1} - \mathbf{w}_i^{\tau_r}) + \frac{1}{\rho}(\mathbf{y}_i^{\tau+1} - \mathbf{y}_i^{\tau_r})$;
8: $\nabla \mathbf{w}_i^\tau \leftarrow \nabla \mathbf{w}_i^\tau / |\nabla \mathbf{w}_i^\tau|$;
9: $[\![\nabla \mathbf{w}_i^\tau]\!] \leftarrow \mathsf{CKKS.Enc}_{pk_s}(\nabla \mathbf{w}_i^\tau)$;
10: c_i uploads $[\![\nabla \mathbf{w}_i^\tau]\!]$ to S_0;
11: **end for**
12: $[\![\nabla \mathbf{w}_m^\tau]\!], [\![\nabla \mathbf{w}_{m_1}^\tau]\!], [\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow DUCC([\![\nabla \mathbf{w}_i^\tau]\!])$;
13: $[\![\mathbf{w}^{\tau+1}]\!] \leftarrow [\![\mathbf{w}^\tau]\!] + \alpha_0 [\![\nabla \mathbf{w}_m^\tau]\!] + \alpha_1 [\![\nabla \mathbf{w}_{m_1}^\tau]\!] + \alpha_2 [\![\nabla \mathbf{w}_{m_2}^\tau]\!]$;
14: $[\![\mathbf{w}^{\tau+1}]\!] \leftarrow Keytransformation([\![\mathbf{w}^{\tau+1}]\!])$;
15: **end for**
16: **return** $[\![\mathbf{w}^{\tau+1}]\!]$.

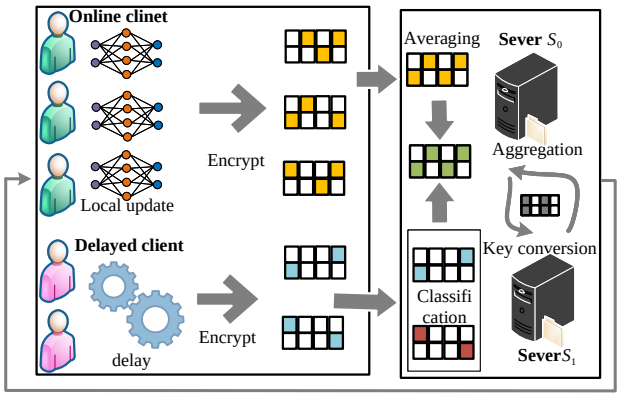


Fig. 4. The main process of PAFed.

Then, the online client $C_i \in \mathcal{C}_0^\tau$ performs E_i rounds of local training, selects the smallest batch of data samples $b \in B$ and the objective function as follow:

$$\min_{w, w_i \in \mathbb{R}^d} \sum_{i=1}^{m} \frac{n_i}{n} f_i(\mathbf{w}_i), \quad \text{s.t.}, \mathbf{w}_i = \mathbf{w}. \tag{15}$$

To solve the objective function, the method of alternating direction multipliers is used to calculate $\mathbf{w}_i^{\tau+1}$ by Eq. 9. As shown in Algorithm 2, after doing E_i rounds local training, the client C_i updates $\mathbf{y}_i^{\tau+1}$ by Eq. 10, then calculates the model update $\nabla \mathbf{w}_i^\tau$ as show in Eq. 16.

$$\nabla \mathbf{w}_i^\tau = \left(\mathbf{w}_i^{\tau+1} - \mathbf{w}_i^{\tau_r}\right) + \left(\mathbf{y}_i^{\tau+1} - \mathbf{y}_i^{\tau_r}\right)/\rho. \tag{16}$$

Furthermore, to achieve correction of delayed model updates in the ciphertext case, C_i standardizes $\nabla \mathbf{w}_i^\tau$ as shown in line 8 of Algorithm 3. Although the quantization process will affect the model accuracy to a certain extent, the impact is still within the acceptable range, and the quantization process also reduces the computing and communication overheads to a certain extent. Then, C_i uses the public key pk_s of S_1 to encrypt locally, namely $\mathsf{CKKS.Enc}_{pk_s}(\nabla \mathbf{w}_i^\tau)$. Finally, C_i sends the encrypted result $[\![\nabla \mathbf{w}_i^\tau]\!]$ to S_0.

Delayed Update Classification and Correction (DUCC). As shown in Algorithm 4, the server S_0 receives encrypted model updates $[\![\nabla \mathbf{w}_i^\tau]\!]$ sent by online clients C_i, and calculates the average value $[\![\nabla \mathbf{w}_m^\tau]\!]$. Then, S_0 records all delayed updates received in this round, as denoted $\nabla W_d = ([\![\nabla \mathbf{w}_0^{\tau_0}]\!], ..., [\![\nabla \mathbf{w}_d^{\tau_d}]\!])$, and initializes $[\![\nabla \mathbf{w}_{m_1}^\tau]\!], [\![\nabla \mathbf{w}_{m_2}^\tau]\!]$, n_1, n_2 to 0. For each delayed model update $[\![\nabla \mathbf{w}_j^{\tau_j}]\!] \in H$, S_0 executes Eq. 13 once. In addition, S_0 also accumulates n_1 and n_2, where n_1 and n_2 respectively record the total accumulated weights of $[\![\nabla \mathbf{w}_{m_1}^\tau]\!], [\![\nabla \mathbf{w}_{m_2}^\tau]\!]$.

Finally, S_0 calculates the average value of $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$ by dividing $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$ by n_1. Similarly, $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ is divided by n_2 to obtain the average value of $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$. Then, S_0 corrects $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$ by Eq. 14.

Algorithm 4 DUCC(·)

Input: Encrypted local model update $[\![\nabla \mathbf{w}_i^\tau]\!]$.
Output: Model updates $[\![\nabla \mathbf{w}_m^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$.

1: **// Execution** S_0
2: S_0 records delayed updates as $\nabla W_d = ([\![\nabla \mathbf{w}_0^{\tau_0}]\!], ..., [\![\nabla \mathbf{w}_d^{\tau_d}]\!])$;
3: $[\![\nabla \mathbf{w}_m^\tau]\!] \leftarrow \sum_{\mathcal{C}_o^\tau} [\![\nabla \mathbf{w}_i^\tau]\!] \Big/ |\mathcal{C}_o^\tau|$ and set $n_1 \leftarrow 0, n_2 \leftarrow 0, [\![\nabla \mathbf{w}_{m_1}^\tau]\!] \leftarrow 0, [\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow 0$;
4: **for** $[\![\nabla \mathbf{w}_d^{\tau_d}]\!] \in H$ **do**
5: $[\![x]\!] = Min\left(\left([\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + [\![1]\!]\right)/2, [\![1/2]\!]\right)$;
6: $[\![\nabla \mathbf{w}_{m_1}^\tau]\!] \leftarrow [\![\nabla \mathbf{w}_{m_1}^\tau]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] \left(\frac{[\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + 1}{2} - [\![x]\!]\right)$;
7: $n_1 \leftarrow n_1 + \left(\frac{[\![\nabla \mathbf{w}_m^\tau]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + 1}{2} - [\![x]\!]\right)$;
8: $[\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow [\![\nabla \mathbf{w}_{m_2}^\tau]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * ([\![x]\!] - [\![1/2]\!])$;
9: $n_2 \leftarrow n_2 + ([\![x]\!] - [\![1/2]\!])$;
10: **end for**
11: $[\![\nabla \mathbf{w}_{m_1}^\tau]\!] \leftarrow [\nabla \mathbf{w}_{m_1}^\tau]\!]/n_1$ and $[\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow [\![\nabla \mathbf{w}_{m_2}^\tau]\!]/n_2$;
12: $[\![\nabla \mathbf{w}_{m_2}^\tau]\!] \leftarrow [\![\nabla \mathbf{w}_{m_2}^\tau]\!] - ([\![\nabla \mathbf{w}_{m_2}^\tau]\!] \cdot [\![\nabla \mathbf{w}_m^\tau]\!])\,[\![\nabla \mathbf{w}_m^\tau]\!]$;
13: **return** $[\![\nabla \mathbf{w}_m^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_1}^\tau]\!]$, $[\![\nabla \mathbf{w}_{m_2}^\tau]\!]$.

Model aggregation. As in Eq. 17, S_0 updates the global model $[\![\mathbf{w}^\tau]\!]$.

$$[\![\mathbf{w}^{\tau+1}]\!] \leftarrow [\![\mathbf{w}^\tau]\!] + \alpha_0 [\![\nabla \mathbf{w}_m^\tau]\!] + \alpha_1 [\![\nabla \mathbf{w}_{m_1}^\tau]\!] + \alpha_2 [\![\nabla \mathbf{w}_{m_2}^\tau]\!], \tag{17}$$

where $\alpha_0, \alpha_1, \alpha_2$ are the weight values of non-delayed model updates, non-conflicted delayed model updates and conflicted delayed model updates, respectively.

Key transformation. Since $[\![\mathbf{w}^{\tau+1}]\!]$ is encrypted by the public key pk_s and cannot be directly decrypted by the clients, we need to perform key conversion. As shown in Algorithm 5, S_0 first randomly generates a non-zero vector r as a mask vector, where the size of r is consistent with the size of $[\![\mathbf{w}^{\tau+1}]\!]$. Then, S_0 locally encrypts r by calling $\mathsf{CKKS.Enc}_{pk_s}(r)$. Then, S_0 confuses $[\![\mathbf{w}^{\tau+1}]\!]$ as $[\![\mathbf{w}^{\tau+1}]\!] + [\![r]\!]$. Finally, S_0 sends $[\![\mathbf{w}^{\tau+1} + r]\!]$ to S_1. After receiving $[\![\mathbf{w}^{\tau+1} + r]\!]$, S_1 first decrypts it with its own private key sk_s to get $\mathbf{w}^{\tau+1} + r$, then encrypts $\mathbf{w}^{\tau+1} + r$ with each client's public key by calling $\mathsf{CKKS.Enc}_{pk_i}(\mathbf{w}^{\tau+1} + r)$, finally sends the encrypted result to S_0. S_0 first encrypts r with the public key of each client to obtain $[\![r]\!]$, then removes the mask by $[\![\mathbf{w}^{\tau+1} + r]\!] - [\![r]\!]$,

finally sends the corresponding $[\![\mathbf{w}^{\tau+1}]\!]$ to the next round of clients.

Algorithm 5 Key transformation $(\cdot)$

Input: Global model $\left[\!\left[\nabla \mathbf{w}_{pk_s}^{\tau+1}\right]\!\right]$.
Output: Global model $\left[\!\left[\nabla \mathbf{w}_{pk_i}^{\tau+1}\right]\!\right]$.
// **Execution** S_0
S_0 randomly selects a vector r as a mask.
$[\![r]\!] \leftarrow \mathsf{CKKS.Enc}_{pk_s}(r)$;
$[\![\mathbf{w}^{\tau+1}+r]\!] = [\![\mathbf{w}^{\tau+1}]\!] + [\![r]\!]$;
S_0 sends $[\![\mathbf{w}^{\tau+1}+r]\!]$ to S_1;
// **Execution** S_1
$\mathbf{w}^{\tau+1}+r \leftarrow CKKS.Dec_{sk_s}([\![\mathbf{w}^{\tau+1}+r]\!])$;
for $i = 0, ...K-1$ **do**
  $[\![r]\!] \leftarrow CKKS.Dec_{sk_i}([\![\mathbf{w}^{\tau+1}+r]\!])$;
  $[\![\mathbf{w}^{\tau+1}+r]\!] \leftarrow \mathsf{CKKS.Enc}_{pk_i}(\mathbf{w}^{\tau+1}+r)$;
end for
S_1 sends $[\![\mathbf{w}^{\tau+1}+r]\!]$ to S_0;
// **Execution** S_0
for $i = 0, ...K-1$ **do**
  $[\![r]\!] \leftarrow \mathsf{CKKS.Enc}_{pk_i}(r)$;
  $[\![\mathbf{w}^{\tau+1}]\!] \leftarrow [\![\mathbf{w}^{\tau+1}+r]\!] - [\![r]\!]$;
end for
S_0 sends the corresponding $[\![\mathbf{w}^{\tau+1}]\!]$ to clients;
return $[\![\mathbf{w}^{\tau+1}]\!]$.

VI. Theoretical Analysis

In this section, we provide comprehensive proof of the correctness and security of PAFed.

Theorem 1 *The theorem on the correctness of delayed model updates under ciphertext demonstrates that, model updates can be accurately classified and accumulated under encryption. Eq.13 ensures that even with delays, the updates can still be effectively classified and accumulated in an encrypted context.*

Proof 1 *Since* $\nabla \mathbf{w}_m^{\tau}, \nabla \mathbf{w}_j^{\tau_j}$ *is a unit vector, when* $\left(\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} + 1\right)/2 < 1/2 \rightarrow \nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} < 0$ *has* $[\![x]\!] = \left([\![\nabla \mathbf{w}_m^{\tau}]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + [\![1]\!]\right)/2$*, then has*

$$\begin{aligned}
&[\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] \leftarrow [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * ([\![x]\!] - [\![1/2]\!]) \\
&\approx [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * \left(\left[\!\left[\frac{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j}+1}{2}\right]\!\right] - [\![1/2]\!]\right) \\
&= [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * \frac{[\![\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j}]\!]}{2} \\
&= [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * \frac{[\![\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j}]\!]}{2\|\nabla \mathbf{w}_m^{\tau}\|\|\nabla \mathbf{w}_j^{\tau_j}\|} \\
&= [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * [\![v_i]\!],
\end{aligned} \tag{18}$$

where $v_i = \frac{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j}}{2\|\nabla \mathbf{w}_m^{\tau}\|\|\nabla \mathbf{w}_j^{\tau_j}\|}$ *is half of the cosine of the angle between the two, which can be used as the weight value for model update, then there is*

$$\begin{aligned}
\frac{[\![\nabla \mathbf{w}_{m_2}^{\tau}]\!]}{2} &= \frac{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} < 0} ([\![\nabla \mathbf{w}_j^{\tau_j}]\!] \cdot [\![v_i]\!])}{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} < 0} [\![v_i]\!]} \\
&= \left[\!\left[\frac{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} < 0} (\nabla \mathbf{w}_j^{\tau_j} \cdot v_i)}{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} < 0} v_i}\right]\!\right].
\end{aligned} \tag{19}$$

And the value of $[\![\nabla \mathbf{w}_{m_2}^{\tau}]\!]$ *remains unchanged,*

$$\begin{aligned}
&[\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] \\
&= [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] \left(\frac{[\![\nabla \mathbf{w}_m^{\tau}]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + [\![1]\!]}{2} - [\![x]\!]\right) \\
&\approx [\nabla \mathbf{w}_{m_1}^{\tau}] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] \cdot [\![0]\!] = [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!].
\end{aligned} \tag{20}$$

When $\left(\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} + 1\right)/2 > 1/2 \rightarrow \nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} > 0$*, we have* $[\![x]\!] = [\![1/2]\!]$*,*

$$\begin{aligned}
[\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] &= [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * ([\![x]\!] - [\![1/2]\!]) \\
&\approx [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * [\![0]\!] \\
&= [\![\nabla \mathbf{w}_{m_2}^{\tau}]\!],
\end{aligned} \tag{21}$$

and then have

$$\begin{aligned}
&[\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] \\
&= [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] \left(\frac{[\![\nabla \mathbf{w}_m^{\tau}]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!] + [\![1]\!]}{2} - [\![x]\!]\right) \\
&\approx [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * \frac{[\![\nabla \mathbf{w}_m^{\tau}]\!] \cdot [\![\nabla \mathbf{w}_j^{\tau_j}]\!]}{2} \\
&= [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * \frac{[\![\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j}]\!]}{2\|\nabla \mathbf{w}_m^{\tau}\|\|\nabla \mathbf{w}_j^{\tau_j}\|} \\
&= [\![\nabla \mathbf{w}_{m_1}^{\tau}]\!] + [\![\nabla \mathbf{w}_j^{\tau_j}]\!] * [\![v'_i]\!],
\end{aligned} \tag{22}$$

finally $[\![\nabla \mathbf{w}_{m_1}^{\tau}]\!]/n_1 = \frac{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} > 0} ([\![\nabla \mathbf{w}_j^{\tau_j}]\!] \cdot [\![v'_i]\!])}{\sum_{\nabla \mathbf{w}_m^{\tau} \cdot \nabla \mathbf{w}_j^{\tau_j} > 0} [\![v'_i]\!]}$ *can be obtained.*

Based on the above, the delayed model updates can be correctly classified.

Theorem 2 *In the process of aggregation and key conversion of the global model* $\mathbf{w}^{\tau}$ *on the server side, malicious attackers cannot obtain any useful information about the local data of the client from* $[\![\mathbf{w}^{\tau}]\!]$*.*

Proof 2 *The security of our scheme relies on the security of CKKS, which in turn is founded on the hardness of the Learning With Errors (LWE) problem. Given that CKKS is CPA-secure and that its security assumption LWE is widely accepted to be hard, we extend this security to PAFed. In our scenario, the model update* $\mathbf{w}^{\tau}$ *is encrypted as* $[\![\mathbf{w}^{\tau}]\!] = \mathsf{CKKS.Enc}_{pk_s}(\mathbf{w}^{\tau})$ *and sent to server* S_0 *for aggregation. It is worth noticing that any adversary cannot gain any useful information about* $\mathbf{w}^{\tau}$ *from* $[\![\mathbf{w}^{\tau}]\!]$ *due to the security of CKKS. Specifically, for any polynomial-time adversary* $\mathcal{A}$*,*

$$\Pr[\mathcal{A}([\![\mathbf{w}^{\tau}]\!]) = \mathbf{w}^{\tau}] < \epsilon, \tag{23}$$

where ϵ *is a negligible value. This means that even if attackers can intercept* $[\![\mathbf{w}^{\tau}]\!]$*, they cannot decrypt* $\mathbf{w}^{\tau}$ *because they do not have the corresponding private key* sk_s*, and the probability of recovering plaintext from* $[\![\mathbf{w}^{\tau}]\!]$ *is almost zero.*

Lastly, in the model's key conversion process, since S_1 *receives the masked model* $[\![\mathbf{w}^{\tau}+r]\!]$*, even if* S_1 *has the private key* sk_s*,* S_1 *cannot obtain information about* $\mathbf{w}^{\tau}$*.* S_0 *does not have the global model's corresponding private keys* sk_s *and* sk_i*. Thus, throughout the training process,* S_0 *operates*

under ciphertext, making it impossible to obtain any valid information.

In summary, based on the CPA security of CKKS, we can conclude that a malicious attacker cannot obtain any useful information about the client's local data from $[\![\mathbf{w}^\tau]\!]$ during the aggregation and update process of $\mathbf{w}^\tau$.

Remark 1 *This solution aims to solve the problem of low global model accuracy caused by the conflict between delayed model update and Non-IID settings in an asynchronous environment. On this basis, a multi-key privacy protection framework is adopted to protect customers' private data. Note that Proof 1 proves that this scheme can correctly correct the direction of delayed model update under ciphertext, making it closer to the direction of the average gradient. Proof 2 explains that the privacy security of this scheme mainly depends on the security of CKKS and mask $[\![r]\!]$. In addition, the introduction of multi-key mechanism and CKKS encryption in this solution will increase the computing overhead of the server, which is a problem that needs to be improved in the future.*

VII. Performance Analysis

We first implement the main algorithms of the proposed scheme, then test the performance of the proposed scheme based on different datasets, finally analyze the experimental results.

TABLE II
MODEL DESCRIPTIONS OF MNIST(FASHION-MNIST), CIFAR-10.

MNIST	$Conv_1$	in_channels=1,out_channels=32,kernel_size=5,padding=2
	Pool	kernel_size=2, stride=2
	$Conv_2$	in_channels=32,out_channels=64,kernel_size=5,padding=2
	Pool	kernel_size=2, stride=2
	FC_1	in_features=3136,out_features=512
	FC_2	in_features=512,out_features=10
CIFAR-10	$Conv_1$	in_channels=3,out_channels=32,kernel_size=5,padding=2
	Pool	kernel_size=2, stride=2
	$Conv_2$	in_channels=32,out_channels=64,kernel_size=5,padding=2
	Pool	kernel_size=2, stride=2
	FC_1	in_features=4096,out_features=256
	FC_2	in_features=256,out_features=10

A. Experiment Parameters

This solution is implemented in Python 3.8. In the training model stage, we rely on TorchVision and Torch two libraries to implement. As shown in TABLE II and TABLE IV, we test MNIST (Fashion-MNIST), CIFAR-10 and CIFAR-100 datasets using different model structures.

TABLE III
SYSTEM PARAMETER SETTINGS.

Notation	Parameter	Notation	Parameter
N	100	σ	0.1
E	10	T	100
B	50	α	0.8
β	0.1	δ	0
SGD	0.5	ρ	0.01

We assume that a total of N clients participate in FL and the number of communication rounds is T, where α is the global model learning rate. And the participation ratio of clients in each round of communication is set to σ, including delayed clients and online clients. Among them, the dropped customers account for the δ proportion of all customers in this round. For local model training, the local model is updated using momentum SGD. In addition, the local batch size is B, the number of local updates is E, the local learning rate is β and the momentum parameter is γ. Unless otherwise specified, the experimental parameters used in this experiment are listed in TABLE III, and the experimental data of this scheme are the average of the five experimental results.

TABLE IV
MODEL DESCRIPTIONS OF CIFAR-100.

CIFAR-100	$Conv_1$	in_channels=3,out_channels=32,kernel_size=3,padding=1
	BatchNorm	num_features=32
	ReLU	inplace=True
	$Conv_2$	in_channels=32,out_channels=64,kernel_size=3
	MaxPool	kernel_size=2, stride=2
	$Conv_3$	in_channels=64,out_channels=128,kernel_size=3
	BatchNorm	num_features=128
	MaxPool	kernel_size=2, stride=2
	Dropout	p=0.05
	$Conv_4$	in_channels=128,out_channels=256,kernel_size=3
	BatchNorm	num_features=256
	MaxPool	kernel_size=2, stride=2
	Dropout	p=0.1
	FC_1	in_features=1024,out_features=512
	ReLU	inplace=True
	Dropout	p=0.1
	FC_2	in_features=512,out_features=100

Data Distribution. Suppose there are $B = \sum_{i=1}^{C} B_i$ training data, where B_i is the number of samples of the i-th class. When constructing the FL settings of IID in this experiment, all data is evenly distributed among K clients. That is, there are C categories of data on each client, and there are B_i/K samples of the i-th category on the k-th client, so that the category distribution on each client is guaranteed to be the same. When constructing the Non-IID settings, we first arrange the training data by label, ensuring that images with similar labels are grouped together. Then, each client is randomly assigned two shards, so that each client is only assigned data with a few tags.

MNIST. MNIST [33] is a handwritten digit image set consisting of 28 $\times$ 28 grayscale digit images from 0 to 9, including 60,000 training samples and 10,000 testing samples. In addition, we set the probability $p = 0.1$ in MNIST, which indicates that the locally trained data is antipodal and identically distributed among clients.

Fashion-MNIST. Fashion-MNIST [34] is an image dataset that can directly replace the MNIST handwritten digit set. The image size, dataset specification, format and division are consistent with the MNIST. But the difference is that Fashion-MNIST has more rich image categories, including 70,000 front image images of different products in 10 categories. Compared with the MNIST, Fashion-MNIST can better reflect the difference in model accuracy compared with other schemes.

CIFAR-10. CIFAR-10 [35] is a color image classification dataset containing 50,000 training samples and 10,000 testing samples. Compared with the MNIST, CIFAR-10 consists of

TABLE V
ACCURACY ON DATASETS.

Setting		FedAdmm	FedAvg	FedProx	PAFed	Setting		FedAdmm	FedAvg	FedProx	PAFed
MNIST	$\delta' = 0$	**0.990473**	0.989264	0.989291	0.990145	CIFAR-10	$\delta' = 0$	0.633473	0.606273	0.601227	**0.6365**
	$\delta = 0$	0.984509	0.978973	0.978891	**0.984918**		$\delta = 0$	0.556409	0.515364	0.512473	**0.579036**
	$\delta = 0.1$	**0.984073**	0.974291	0.977264	0.9836		$\delta = 0.1$	0.536769	0.508245	0.511881818	**0.587304**
	$\delta = 0.2$	0.982358	0.974067	0.97535	**0.983567**		$\delta = 0.2$	0.533565	0.496036	0.498927	**0.589371**
	$\delta = 0.3$	0.972509	0.973555	0.970836	**0.983545**		$\delta = 0.3$	0.520966	0.49287	0.49345	**0.587706**
	$\delta = 0.4$	0.981427	0.972964	0.968773	**0.9829**		$\delta = 0.4$	0.508752	0.47598	0.47562	**0.578112**
	$\delta = 0.5$	0.979455	0.967591	0.963509	**0.983918**		$\delta = 0.5$	0.486669	0.452309	0.457945	**0.569251**
	$\delta = 0.6$	0.977173	0.963764	0.960782	**0.982609**		$\delta = 0.6$	0.45718	0.444036	0.443982	**0.559191**
	$\delta = 0.7$	0.973327	0.946845	0.949818	**0.976918**		$\delta = 0.7$	0.414106	0.41387	0.41936	**0.526318**
FMNIST	$\delta' = 0$	**0.905864**	0.899118	0.898036	0.905055	CIFAR-100	$\delta' = 0$	0.5146	0.5006	0.5046	**0.5192**
	$\delta = 0$	0.842264	0.835291	0.835818	**0.867282**		$\delta = 0$	0.4489	0.4	0.4175	**0.4617**
	$\delta = 0.1$	0.840536	0.826382	0.828691	**0.874289**		$\delta = 0.1$	0.4419	0.3976	0.4042	**0.4662**
	$\delta = 0.2$	0.840425	0.819	0.823355	**0.871273**		$\delta = 0.2$	0.4402	0.3893	0.4083	**0.4506**
	$\delta = 0.3$	0.832653	0.811727	0.812382	**0.871589**		$\delta = 0.3$	0.43	0.378	0.3999	**0.4361**
	$\delta = 0.4$	0.823685	0.796909	0.801527	**0.866727**		$\delta = 0.4$	0.4159	0.3662	0.3827	**0.4436**
	$\delta = 0.5$	0.81222	0.786509	0.801527	**0.866622**		$\delta = 0.5$	0.3932	0.3509	0.3586	**0.424**
	$\delta = 0.6$	0.798987	0.770927	0.769182	**0.855789**		$\delta = 0.6$	0.3479	0.3252	0.3461	**0.3862**
	$\delta = 0.7$	0.780927	0.732355	0.741073	**0.839551**		$\delta = 0.7$	0.2477	0.3019	0.3122	**0.3491**

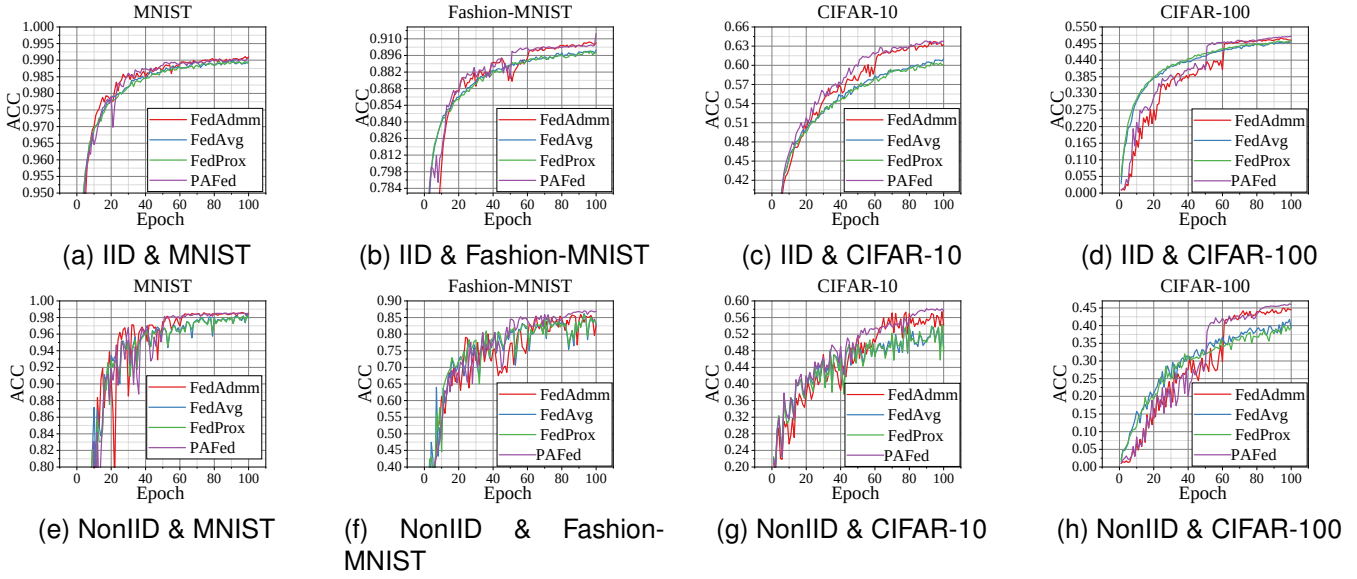


Fig. 5. Compare the performance of each scheme under the IID settings and Non-IID settings, that is, under the IID settings, Non-IID settings and $\delta = 0$, test different datasets to see how the global model accuracy changes with the increase in the number of communication rounds.

10 categories of 32×32 color images, and CIFAR-10 contains real objects in the real world, which are not only noisy but also have different characteristics and proportions of image objects, which also brings great difficulties to recognition. For example, the direct linear model Softmax dose not perform well on CIFAR-10.

CIFAR-100. CIFAR-100 [35] contains color images in 100 categories, with 600 images in each category. These images are small images of 32× 32 pixels, covering various objects such as animals, vehicles, buildings, people, etc. Compared with CIFAR-10, CIFAR-100 is usually used for more advanced research and model evaluation because it has more categories and is more challenging.

B. Experiment Results

We first test the performance of global model of PAFed under the IID settings and $\delta = 0$. As shown in Fig. 5, we notice that PAFed can maintain the same level of model performance as other baseline methods in the IID settings. Specifically, when the number of training rounds exceeds 50 rounds, the model accuracy of PAFed is slightly higher than those of other baseline schemes. Especially on CIFAR-10, the average best model accuracy of PAFed is 3% higher than that of FedAvg. The "average best model accuracy" specifically denotes the mean of the highest accuracies recorded in each of the final 10 training rounds. This approach offers a focused insight into the model's performance stability and effectiveness in its mature training stages, offering a reliable indicator of its expected model performance.

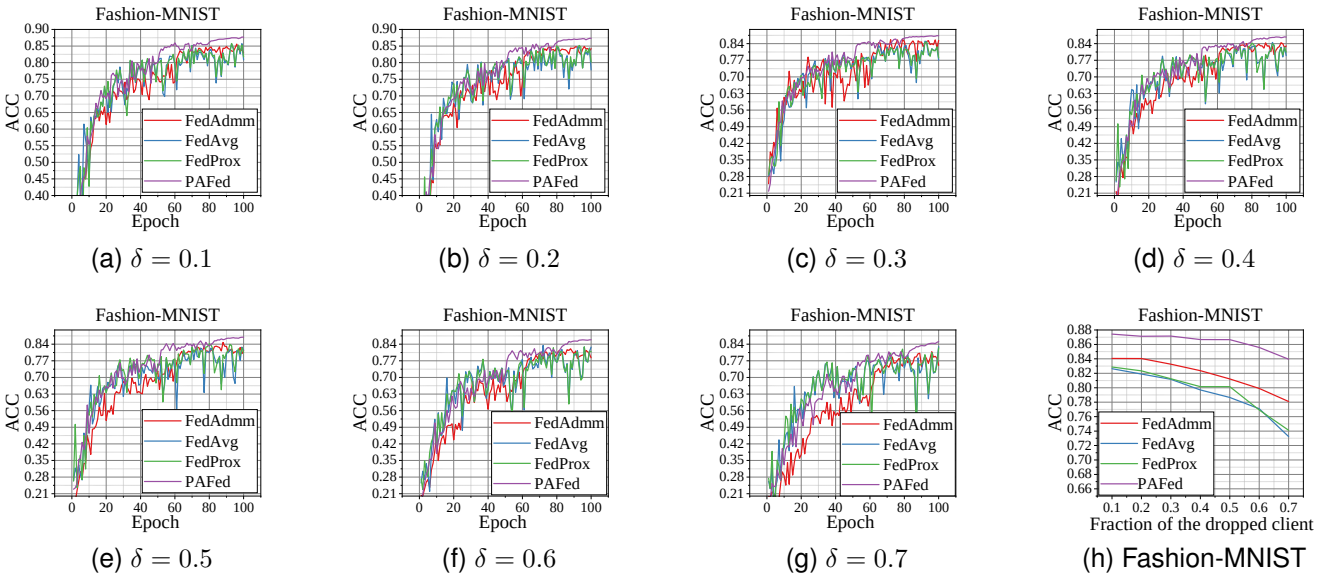


Fig. 6. Compare the performance of each scheme under Fashion-MNIST, Non-IID settings and system heterogeneous settings, that is, in the case of Non-IID settings and taking δ equal to 0.1 to 0.7 respectively, as the number of communication rounds increases, the accuracy of global model changes above. The Fig. 6 (h) shows the average accuracy of global model after 10 rounds under different δ.

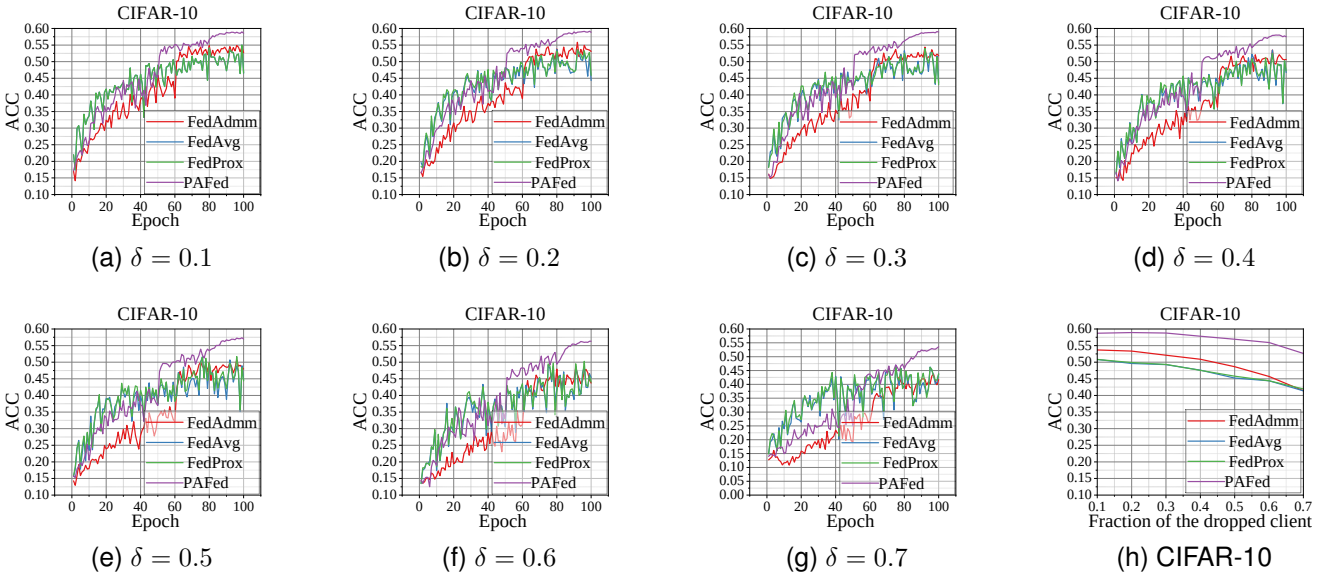


Fig. 7. Compare the performance of each scheme under CIFAR-10, Non-IID settings and system heterogeneous settings, that is, in the case of Non-IID settings and taking δ equal to 0.1 to 0.7 respectively, as the number of communication rounds increases, the accuracy of global model changes above. The Fig. 7 (h) shows the average accuracy of global model after 10 rounds under different δ.

Secondly, to better verify the performance of PAFed under Non-IID settings. For the experiment, we distribute the data according to a Non-IID settings. We also ensure that the global model aggregation process is immediate, with no gradient delay, signified by setting δ to 0. Detailed examination of Fig. 5 reveals that when the number of global rounds exceeds 60, PAFed outperforms competing methodologies in terms of accuracy and exhibits superior resilience to data heterogeneity. This superior performance of PAFed, as well as that of FedAdmm, can be attributed to the utilization of the ADMM optimization algorithm. Both methodologies employ ADMM to fine-tune the constrained loss function, significantly diminishing the discrepancies among the local models of clients in Non-IID settings, thereby enhancing the effectiveness of model aggregation. Notably, this enhancement in aggregation efficacy becomes more pronounced with an increase in the number of global rounds.

Furthermore, a comparative analysis of the average best model accuracy across four distinct datasets substantiates that PAFed's performance is consistently on par with or superior to that of FedAdmm and surpasses the performances of alternative methodologies. Specifically, in the context of the Fashion-MNIST dataset, PAFed's average best model accuracy surpasses that of FedAvg by approximately 3%. When the analysis extends to more intricate datasets, such as CIFAR-10 and CIFAR-100, the advantages of PAFed become even

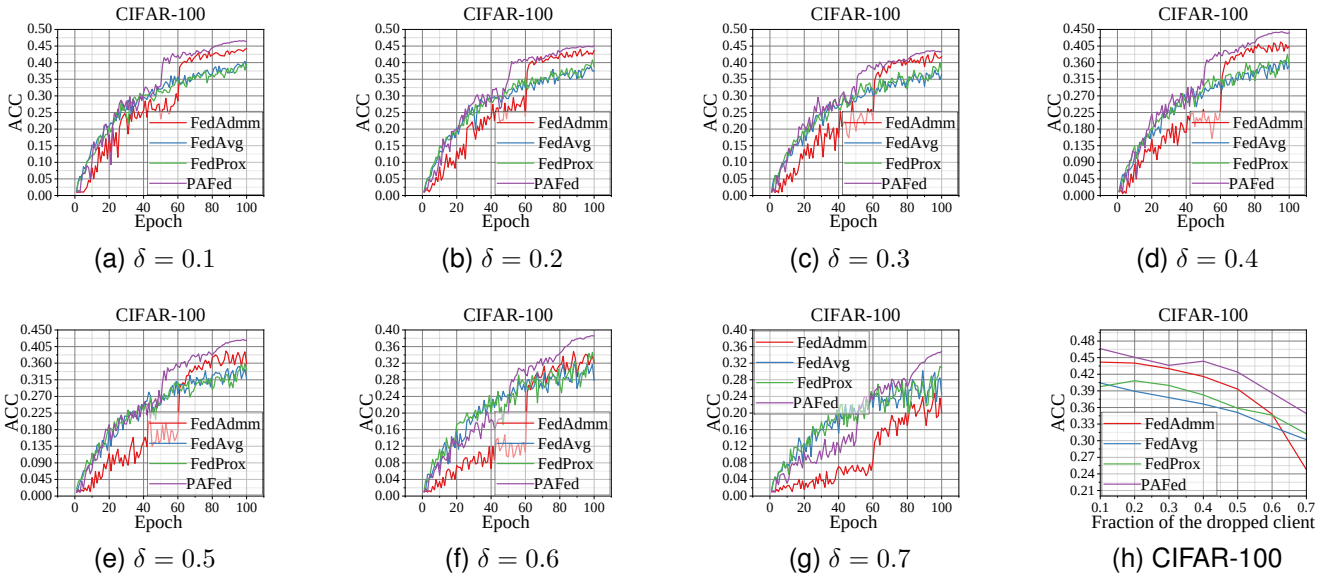


Fig. 8. Compare the performance of each scheme under CIFAR-100, Non-IID settings and system heterogeneous settings, that is, in the case of Non-IID settings and taking δ equal to 0.1 to 0.7 respectively, as the number of communication rounds increases, the accuracy of global model changes above. The Fig. 8 (h) shows the average accuracy of global model after 10 rounds under different δ.

more discernible. On the CIFAR-10 dataset, PAFed's average best model accuracy exceeds that of FedAvg by roughly 6.4%, and on CIFAR-100, the superiority margin of PAFed's average best model accuracy over FedAvg's is about 6.1%. This data underscores the robustness and adaptability of PAFed in handling diverse and complex data landscapes, showcasing its efficacy in Non-IID settings.

In particular, when the dataset is Fashion-MNIST, we observe that PAFed demonstrates distinct advantages across varying levels of δ. To elucidate, at a δ setting of 0.1, PAFed's model accuracy surpasses that of FedAdmm by a notable margin of 3% and exhibits a 5% improvement over FedAvg. This advantage becomes more pronounced as we elevate the heterogeneity level to $\delta = 0.5$, where PAFed's model accuracy advantage expands to 5% over FedAdmm and 7% over FedAvg. Remarkably, when analyzing the performance at a δ of 0.5 on Fashion-MNIST, PAFed's model accuracy is 6% higher than FedAdmm's and 10% higher than FedAvg's.

Transitioning to an analysis on the CIFAR-10 and CIFAR-100 datasets, the benefits of employing PAFed are even more conspicuous. For the CIFAR-10 dataset, at a minimal heterogeneity level ($\delta = 0.1$), PAFed's model accuracy is more than 5% higher than that of FedAdmm and 8% higher than that of FedAvg. This gap widens further when δ is adjusted to 0.5, where PAFed's model accuracy exceeds FedAdmm's by 8% and FedAvg's by 11%. Noteworthy is the scenario under $\delta = 0.7$, where PAFed's model accuracy leads by 11% compared to both FedAdmm and FedAvg.

In addition, an overarching trend is identified across all datasets, as δ increases, there is a universal decline in model accuracy across all schemes. However, PAFed exhibits a more tempered reduction in global model accuracy compared to other schemes, indicating its superior resilience to data heterogeneity. This trend is graphically represented in Fig. 6(h) and 7(h), which illustrate that the model accuracy of PAFed on the Fashion-MNIST dataset declines more gradually than that of other schemes when $0.1 < \delta < 0.6$. A similar pattern is observed in the CIFAR-10 and CIFAR-100 datasets, reinforcing the robustness and adaptability of PAFed in the face of varying data heterogeneity levels.

Furthermore, an intriguing observation emerges when the value of δ surpasses the 0.5 threshold. In such instances, FedAdmm exhibits a precipitous decline in model accuracy, outpacing the downward trends of other schemes. This is vividly illustrated in Fig. 8(g), where the model accuracy of FedAdmm notably falls below that of FedAvg, underscoring a significant performance discrepancy. This phenomenon can be partially explained by the inherent characteristics of the FedAdmm. Specifically, FedAdmm's performance is intrinsically linked to the number of participating clients, its performance escalates with more clients but faces challenges in scenarios characterized by client disconnections or limited participation. Such a trait underscores the sensitivity of FedAdmm to the network's client architecture, rendering it less robust in the face of fluctuating client availability. In stark contrast, our proposed method, PAFed, introduces a sophisticated vector projection technology as a strategic countermeasure to address and rectify the issues arising from delayed model updates. This innovative approach not only compensates for the potential shortfall in client numbers but also significantly reduces the conflicts and inconsistencies typically associated with delayed updates. By effectively harmonizing the divergent updates from various clients, our method enhances the overall robustness and consistency of the model training process under Non-IID settings. TABLE V shows the comparative analysis of the average best model accuracy achieved by PAFed against other prevalent schemes across a diverse array of datasets and settings. In this table, the notation $\delta' = 0$ explicitly denotes a scenario where $\delta = 0$ and under IID settings, while all other entries pertain to Non-IID settings. Notably, the black text

within the table is employed to highlight the best or maximum values observed in each row, offering a clear and immediate visual cue to identify the superior performance metrics.

VIII. CONCLUSION

We propose an admm-based heterogeneous PPFL scheme in a asynchronous update environment, which can effectively solve the problem of data heterogeneity and low model accuracy in a system heterogeneous environment. Specifically, our scheme reduces the difference between client's local model and current global model by settings constraints and introducing the ADMM algorithm for solution. Secondly, considering the problem of system heterogeneity, we use the average update direction to classify the delayed gradient. Then,, the vector projection technology is used to correct the conflicting delayed model update, so that the delayed model update is close to the current average update. In addition, we also introduce CKKS to this framework to protect client privacy and prevent local data leakage.

Building on PAFed, we will primarily focus on reducing computational and communication overhead in future. Given the asynchronous update environment's effectiveness in addressing data heterogeneity and enhancing model accuracy, the next steps aim to design more sophisticated algorithms that minimize the frequency of communication while ensuring robust convergence rates. Additionally, we plan to refine the vector projection technique to more accurately align delayed model updates with the current average update direction, even in highly heterogeneous systems. Further research will also explore advancements in fully homomorphic encryption like CKKS to strengthen client privacy measures without compromising computational performance. The ultimate goal is to achieve a balance between privacy, accuracy, and resource efficiency in FL.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (No. 62072361), Shaanxi Fundamental Science Research Project for Mathematics and Physics (No. 22JSY019), the National Natural Science Foundation of China (No.62125205), Fellowship pf China Postdoctoral Science Foundation (No. 2022T150507), Opening Project of Intelligent Policing Key Laboratory of Sichuan Province (No. ZNJW2023KFMS002), Open fund of Key Laboratory of Computing Power Network and Information Security (No. 2023ZD020).

REFERENCES

[1] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint*, 2016, arXiv:1610.02527.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artificial intelligence and statistics (AISTATS'17)*, 2017, pp. 1273–1282.

[3] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[4] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," *arXiv preprint*, 2020, arXiv:2002.06440.

[5] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint*, 2019, arXiv:1903.03934.

[6] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-iid data," in *Proc. International Conference on Big Data (Big Data'20)*. IEEE, 2020, pp. 15–24.

[7] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *Proc. International Conference on Machine Learning (ICML'18)*. PMLR, 2018, pp. 3043–3052.

[8] S. Zheng, Q. Meng, T. Wang, W. Chen, N. Yu, Z.-M. Ma, and T.-Y. Liu, "Asynchronous stochastic gradient descent with delay compensation," in *Proc. International Conference on Machine Learning (ICML'17)*. PMLR, 2017, pp. 4120–4129.

[9] J. Langford, A. Smola, and M. Zinkevich, "Slow learners are fast," *arXiv preprint*, 2009, arXiv:0911.0491.

[10] X. Yan, Y. Miao, X. Li, K.-K. Raymond, X. Meng, and R. H. Deng, "Privacy-preserving asynchronous federated learning framework in distributed iot," *IEEE Internet of Things Journal*, 2023, doi:10.1109/JIOT.2023.3262546.

[11] Y. Miao, Z. Liu, X. Li, M. Li, H. Li, K.-K. R. Choo, and R. H. Deng, "Robust asynchronous federated learning with time-weighted and stale model aggregation," *IEEE Transactions on Dependable and Secure Computing*, 2023, doi:10.1109/TDSC.2023.3304788.

[12] Z. Wang, X. Fan, J. Qi, C. Wen, C. Wang, and R. Yu, "Federated learning with fair averaging," *arXiv preprint*, 2021, arXiv:2104.14937.

[13] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 8, pp. 3710–3722, 2020.

[14] A. Ghosh, J. Chung, D. Yin, and K. Ramchandran, "An efficient framework for clustered federated learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19 586–19 597, 2020, doi:10.1109/TIT.2022.3192506.

[15] P. P. Liang, T. Liu, L. Ziyin, N. B. Allen, R. P. Auerbach, D. Brent, R. Salakhutdinov, and L.-P. Morency, "Think locally, act globally: Federated learning with local and global representations," *arXiv preprint*, 2020, arXiv:2001.01523.

[16] S. Vahidian, M. Morafah, and B. Lin, "Personalized federated learning by structured and unstructured pruning under data heterogeneity," in *Proc. international conference on distributed computing systems workshops (ICDCSW'21)*. IEEE, 2021, pp. 27–34.

[17] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[18] Y. Gong, Y. Li, and N. M. Freris, "Fedadmm: A robust federated deep learning framework with adaptivity to system heterogeneity," in *Proc. International Conference on Data Engineering (ICDE'22)*. IEEE, 2022, pp. 2575–2587.

[19] B. Gu, A. Xu, Z. Huo, C. Deng, and H. Huang, "Privacy-preserving asynchronous vertical federated learning algorithms for multiparty collaborative learning," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 11, pp. 6103–6115, 2021.

[20] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2134–2143, 2019.

[21] T. Zhang, A. Song, X. Dong, Y. Shen, and J. Ma, "Privacy-preserving asynchronous grouped federated learning for iot," *IEEE Internet of Things Journal*, vol. 9, no. 7, pp. 5511–5523, 2021.

[22] J. Huang, C. Xu, Z. Ji, S. Xiao, T. Liu, N. Ma, and Q. Zhou, "Aflpc: an asynchronous federated learning privacy-preserving computing model applied to 5g-v2x," *Security and Communication Networks*, vol. 2022, 2022, doi:https://doi.org/10.1155/2022/9334943.

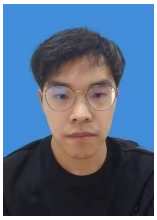
[23] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM Conference on Computer and Communications Security (CCS'17)*, 2017, pp. 1175–1191.

[24] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *Proc. Symposium on Network System Design and Implementation (NSDI'17)*, 2017, pp. 259–282.

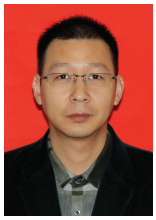
[25] Y. Aono, T. Hayashi, L. Wang, S. Moriai *et al.*, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[26] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. ACM conference on computer and communications security (CCS'15)*, 2015, pp. 1310–1321.
[27] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *Proc. International Conference on Machine Learning (ICML'20)*. PMLR, 2020, pp. 5132–5143.
[28] X. Lu, Y. Liao, P. Lio, and P. Hui, "Privacy-preserving asynchronous federated learning mechanism for edge network computing," *IEEE Access*, vol. 8, pp. 48 970–48 981, 2020, doi:10.1109/ACCESS.2020.2978082.
[29] E. Wei and A. Ozdaglar, "Distributed alternating direction method of multipliers," in *Proc. IEEE Conference on Decision and Control (CDC'12)*, 2012, pp. 5445–5450.
[30] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.
[31] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. International Conference on the Theory and Applications of Cryptology and Information Security (ASIACRYPT'17)*. Springer, 2017, pp. 409–437.
[32] J. H. Cheon, D. Kim, D. Kim, H. H. Lee, and K. Lee, "Numerical method for comparison on homomorphically encrypted numbers," in *Proc. International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'2019)*. Springer, 2019, pp. 415–445.
[33] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.
[34] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint*, 2017, arXiv:1708.07747.
[35] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

Yinbin Miao (Member, IEEE) received the B.E. degree with the Department of Telecommunication Engineering from Jilin University, Changchun, China, in 2011, and Ph.D. degree with the Department of Telecommunication Engineering from Xidian University, Xi'an, China, in 2016. He is also a postdoctor in Nanyang Technological University from September 2018 to September 2019, and a postdoctor in City University of Hong Kong from December 2019 to December 2021. He is currently a Professor with the Department of Cyber Engineering in Xidian University, Xi'an, China. His research interests include information security and applied cryptography.



Da Kuang Da Kuang received the B.S. degree with Department of Information Security from Hainan University, Hainan, China, in 2022. Now he is pursuing his M.S. degree with the Department of Cyber Engineering from Xidian University, Xi'an, China. His research interests include information security and federated learning.



Xinghua Li received the M.S. and Ph.D. degrees in computer science from Xidian University, China, in 2004 and 2007, respectively. He is currently a Professor with the School of Cyber Engineering, Xidian University. His research interests include wireless networks security, privacy protection, cloud computing, and security protocol formal methodology.

Shujiang Xu was born in Linyi, China in 1979. He has received the B.S. degree in mathematics and applied mathematics from Shandong Normal University, China in 2003, the M.S. degree in applied mathematics and Ph.D. degrees in cryptography from Beijing University of Posts and Telecommunications, China, in 2006 and 2014, respectively. He is currently a Research Professor with the Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology (Shandong Academy of Sciences), China. His main research interests include data security and privacy protection, blockchain, and multimedia information security.

Hongwei Li (Fellow, IEEE) received the PhD degree in computer software and theory from the University of Electronic Science and Technology of China, Chengdu, China, in 2008. He is currently a professor with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, China. His research interests include network security, applied cryptography, and trusted computing. He is a member of China Computer Federation.

Kim-Kwang Raymond Choo (Senior Member, IEEE) received the Ph.D. in Information Security in 2006 from Queensland University of Technology, Australia, and currently holds the Cloud Technology Endowed Professorship at The University of Texas at San Antonio (UTSA). He is the founding co-Editor-in-Chief of ACM Distributed Ledger Technologies: Research & Practice, founding Chair of IEEE TEMS Technical Committee on Blockchain and Distributed Ledger Technologies, an ACM Distinguished Speaker and IEEE Computer Society Distinguished Visitor (2021 - 2023), and a Web of Science's Highly Cited Researcher (Computer Science - 2021, Cross-Field - 2020). He is also the recipient of the 2019 IEEE Technical Committee on Scalable Computing Award for Excellence in Scalable Computing (Middle Career Researcher).

Robert H. Deng (Fellow, IEEE) has been an AXA Chair Professor in cybersecurity and a Professor in information systems with the School of Information Systems, Singapore Management University, since 2004. His research interests include data security and privacy, multimedia security, networks, and system security. He served/is serving on the editorial boards of many international journals, including IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING. He has received the Distinguished Paper Award (NDSS 2012), Best Paper Award (CMS 2012), and Best Journal Paper Award (IEEE Communications Society 2017).