Accuracy Degrading: Toward Participation-Fair Federated Learning

Xianyao You, Ximeng Liu, *Senior Member, IEEE*, Xuanwei Lin, Jianping Cai, and Shaoquan Chen

***Abstract*—Centralized learning now faces data mapping and security constraints that make it difficult to carry out. Federated learning with a distributed learning architecture has changed this situation. By restricting the training process to participants' local, federated learning addresses the model training needs of multiple data sources while better protecting data privacy. However, in real-world application scenarios, federated learning faces the need to achieve fairness in addition to privacy protection. In practice, it could happen that some federated learning participants with specific motives may short join the training process to obtain the current global model with only limited contribution to the federated learning whole, resulting in unfairness to the participants who previously participated in federated learning. We propose the FedACC framework with a server-initiated global model accuracy control method to address this issue. Besides measuring the accumulative contributions of newly joined participants and providing participants with a model with an accuracy that matches their contributions, the FedACC still guarantees the validity of participant gradients based on the accuracy-decayed model. Under the FedACC framework, users do not have access to the full version of the current global model early in their training participation. However, they must produce a certain amount of contributions before seeing the full-accuracy model. We introduce an additional differential privacy mechanism to protect clients' privacy further. Experiments demonstrate that the FedACC could obtain about 10%–20% accuracy gain compared to the state-of-the-art methods while balancing the fairness, performance, and security of federated learning.**

***Index Terms*—Data privacy, deep learning, fairness, federated learning.**

I. Introduction

With the development of IoT and the Internet, we could have more ability to train the model that breaks out the record highs, based on the vast amount of data the IoT contributing [1], [2], [3]. However, traditional centralized machine learning has difficulty utilizing data while ensuring that the privacy in the data is protected. In order to address the need, the framework of federated learning [4] was proposed to break the practical "data silo" problem, which has changed the practice paradigm of data utilization in machine learning today [5].

Manuscript received 29 September 2022; revised 27 December 2022; accepted 13 January 2023. Date of publication 19 January 2023; date of current version 7 June 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62072109 and Grant U1804263, and in part by the Natural Science Foundation of Fujian Province under Grant 2021J06013. *(Corresponding author: Ximeng Liu.)*

The authors are with the College of Computer and Data Science and the Fujian Provincial Key Laboratory of Information Security of Network Systems, Fuzhou University, Fuzhou 350108, China (e-mail: snbnix@gmail.com).

Digital Object Identifier 10.1109/JIOT.2023.3238038

However, the initially proposed federated learning framework is not perfect. With privacy, existing incentives and fairness mechanisms are not feasible for practice, e.g., [6], [7], and [8] that depend on the client quantifying its capabilities and data quality honestly. With such difficulties, it is essential to measure the client's contribution in a feasible way.

In federated learning, one key to ensuring fairness is implementing rules that match the client's contribution to model training with the actual reward received by the client. In some cases, such rules are implemented in a mandatory way, such as the reputation-based mechanism approach [9], [10] that forces clients not to persistently cheat other clients to receive rewards that do not match what they contributed. And the server could reward the clients with a monetary-based approach [11], [12], [13]. But what is often overlooked is that the global model distributed by the server in the federated learning is also the reward for the client's participation in the training process [14]. It is undeniable that the ability to handle the model reward term affects the fairness of the federated learning process [15], [16].

However, how to measure the model reward is still a problem. Under the relatively stronger assumption, e.g., the clients always are available from the training starting to the training ending [17], the fairness of model reward is more focused on the one obtained by each client at the end of final training [6], [18]. However, in practical scenarios, it is usually not guaranteed that clients can always participate in a federated learning training process for a long time. An unfair scenario is illustrated in Fig. 1. For the scenario that the cloud server designs and launches federated learning, several participants with normal behavior join the training process at the moment t_1, t_2, and t_3, respectively. Furthermore, a participant with malicious motivation also joins the training process at a later moment t_4 and exits the training process but steals the model without contributing to the federated learning process, just like a "thief." In that case, the clients can obtain the training results of other clients in the previous longer federated learning and only make limited contributions to the system, leading to unfairness to the clients previously joined in the training. Furthermore, as the training process advances, the fairness risk associated with this asymmetry is further increased, i.e., the marginal gain from the federated learning system's absorption of an honest client relative to the loss of fairness risk caused by the absorption of a malicious client becomes increasingly prominent.

To address the fairness risk shown in Fig. 1, we propose a federated adaptive accuracy controlling (FedAAC) federated

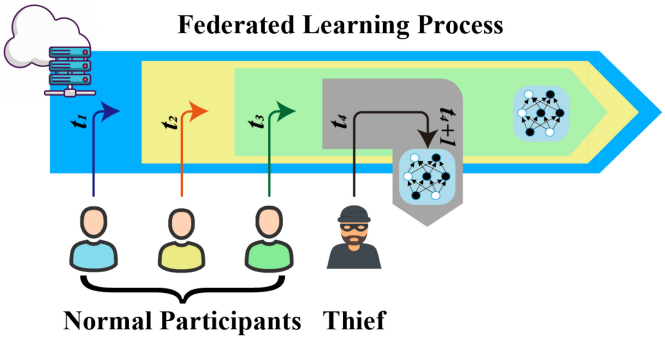


Fig. 1. Malicious participant stealing FL model.

learning framework to prevent clients from maliciously acquiring the global model. For scenarios where clients dynamically join and uncontrollably exit from federated learning, the framework restricts the global models that participating clients can acquire to match the current contributions made by the clients through an adaptive accuracy control approach. Specifically, FedACC generates a decayed version of the current global model based on a client-side "model accuracy degrading" process defined by the server. In order to prevent the client-side local gradients generated on such an accuracy degraded version of the model from degrading the validity for the optimization of the current global model, FedACC also designs a "gradient invariance loss term" to ensure the validity of the gradients from the clients. By evaluating the clients' contributions, the server can achieve fairness to existing clients through adaptive precision control.

The main contributions of this article can be summarized as follows.

1) In this article, we propose the FedAAC framework to design a defense against the vulnerability in the fairness of model reward in the process of client participation in federated learning by dynamically controlling the model accuracy to match the contribution of existing participating clients.
2) We design the accuracy degrading algorithm for model accuracy degrading in a federated learning scenario, obtaining a decaying version of the accuracy of the global model by executing the accuracy degrading task specified by the server on the client side.
3) We introduce the problem of the unbalance between the client and the central server for the model reward. Furthermore, based on the problem, we propose and explain that assurance should be introduced to make the clients' contributions always match the model reward they received.
4) We introduce the differential privacy (DP) mechanism into the FedACC implementation, improve the client-level DP approach for the scenarios to which FedACC targets, and further analyze the characteristics and security of DP under FedACC from theoretical and experimental perspectives.
5) Our approach provides scalable frameworks that can be extended to existing federated learning approaches to obtain model reward fairness guarantees while obtaining competitive results relative to no fairness guarantees in the framework about participation from multiple clients.

The remainder of this article is organized as follows. In Section II, we discuss existing work in related areas. Section III gives some background knowledge related to our topic. Section IV lists the problems and goals we are targeting. Section V presents the implementation detail of FedACC. Furthermore, some security analyses of our method are given in Section VI. Experiments supporting our method are evaluated and analyzed in Section VII. Section VIII concludes this article.

II. Related Work

A. Federated Learning

Federated learning was first seen in a paper published by [4] as FedAvg schema. The federated learning method FedAvg's basic architecture of federated learning is provided, including a distributed training architecture with a central server for task coordination and parameter aggregation. One of its most essential features is that, in contrast to general distributed computing, the training process is restricted to the client's locality, and only the intermediate results of the training process are exchanged with the central server. However, current work points out that FedAvg has many problems in practice [16], [19], [20], [21], including different aspects, such as communication efficiency [22], non-IID [23], and privacy protection [24], in addition to fairness issues.

B. Fairness in Federated Learning

For the different parts of the basic framework of federated learning, as well as for the additional variables that arise from the practical application of federated learning, the current work proposes a variety of improvement options at the level of abstraction from theory to concrete practice. Roughly corresponding to the different stages of federated learning, existing fairness improvement schemes can be classified into fairness in client selection [25], fairness in model optimization [26], fairness in contribution evaluation [6], [27], and fairness in incentive mechanisms [6], [11], [18]. In general, the improvement schemes for a particular category have different constraints to simplify the complex environment encountered in practice.

C. Incentive Mechanism in Federated Learning

The FMore [11] uses a monetary-based metric to reward the clients, which motivates the most high-quality client while minimizing the total cost of the server. Based on the VCG mechanism, the Fair-VCG [12] tries to incentivize the clients to report their contribution honestly to let the server distribute rewards correctly by measuring the per-unit-value of data. Concerning the fairness of the rewards paid by the trained model, the HFFL framework proposed by [6] attempts to achieve different performance model control with hierarchical training of multiple models trained with different amounts of data. At the same time, the dependence on external metrics on client quality makes it not feasible in practice.

TABLE I
NOTATION DESCRIPTIONS

Notations	Descriptions
ϵ	Privacy budget for differential privacy
δ	Relaxation term for differential privacy
$\hat{M}$	Gradient clipping threshold
D	Client-side data sets
T	total rounds of federated learning
S	the participant set under the client-silo scenario
θ	Model parameters
∇	Model parameter gradient
Δ	Model parameter updates
m	Gradient first-order momentum
v	Gradient second order momentum
q	the sample rate of participant in one round
$\mathcal{L}$	the Laplace distribution or a loss item
$\mathcal{N}$	the Gaussian distribution

Furthermore, the FPPDL [18] also provides a solution in distributed decentralized scenarios based on the reputation-based gradient exchange fairness mechanism. However, FPPDL is not suitable for horizontal federated learning.

D. Differential Privacy

The theory of the DP mechanism is first introduced by [28], which protects privacy information users' data by adding noise. In subsequent work, Abadi et al. [29] further implemented DP on deep learning. In the federated learning scenario, Geyer et al. [30] gave a theoretical and practical approach to achieving client-level DP. Truex et al. [31] applied the DP to the data example level without considering the whole data distribution. Moreover, under different FL scenarios, e.g., the personalized federated learning, DP also can be adjusted to fit such situation [32]. In this article, we try to combine FedACC with the client-level DP approach proposed by [30], to enhance the security of our approach. Moreover, we will further analyze the specific characteristic under the DP mechanism to illustrate a practical principle of FedACC.

III. PRELIMINARIES

In this section, we provide a basic introduction to the principles involved, including the federated learning architecture. Some of the notations that we will use are explained in Table I.

A. Federated Learning

The federated averaging (FedAvg) [4] is the first proposed framework of federated learning. A central server and a set of participants C will exist in a typical federated learning scenario. Usually, the server will determine the total rounds T the training process will be. For each round $t \sim \{1, \ldots, T\}$, a certain percentage of participants C_t will be sampled to join the round training, in which the sample rate is represented as $q \sim (0, 1]$. Each participant has its local data set D_c, on which the participant will perform local training, e.g., the SGD. The global model θ_g^t will be distributed to each participant joining the round, and the updated model $\theta_c^t, c \sim C_t$ will be sent back to the server in each round. Furthermore, the server will average the updated models from participants to generate the global model for the next round, which can be represented as $\theta_g^t = (1/|C|) \sum_{c \in C_t} \theta_c^t$.

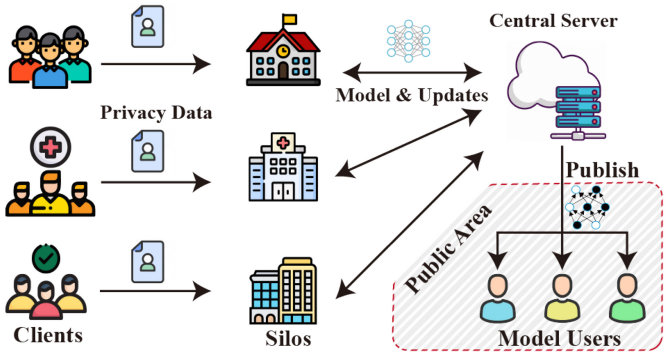


Fig. 2. Framework of system model.

B. Differential Privacy

In the theory of DP, there is an abstract query function $f(x) : x \to R$, x from a data distribution D. The query function can represent any process with its input and output, e.g., the SGD. Based on the query function, we further define the concept of sensitivity. For two data distributions D, D' that differ only in the presence of a specific data item, then D, D' is called an adjacent data set. For any two adjacent data sets, the sensitivity between them in DP can be represented as $\Delta f = \max_{D,D'} \|f(D) - f(D')\|_p$. In the definition of sensitivity, a parameter p represents the norm of distance, which will be different in different DP mechanisms, e.g., $p = 1$ (L_1 norm) for the Laplace mechanism and $p = 2$ (L_2 norm) for the Gaussian mechanism.

With the definition of the sensitivity, the core equation of DP can be represented as $P[M(D) \in S] \leq e^{\epsilon} P[M(D') \in S]$. The protection of DP is implemented by adding noise that satisfies specific conditions, which is different in different mechanisms. For the Laplacian mechanism of DP, representing the Laplacian distribution as $\mathcal{L}(\mu, \beta)$, by adding Laplace noise satisfying the distribution $\mathcal{L}(0, \Delta f/\epsilon)$ to the output of f, the noise-added function $\hat{f}$ will satisfy ϵ DP. For the Gaussian mechanism [30], representing the Gaussian distribution as $\mathcal{N}(\mu, \sigma)$, by adding Gaussian noise satisfying the distribution $\mathcal{N}(0, \hat{M}/\epsilon)$, where $\hat{M} = \max \|f(d)\|_2, d \in D$, to the output of f, the noise-added function $\hat{f}$ will satisfy ϵ DP, but with δ probability the protection fails, where $\delta \leq (4/5) \exp(-(\sigma\epsilon)^2/2)$.

IV. PROBLEM FORMULATION

In this section, we introduce the scenario and the goal of the FedACC target.

A. System Model

The framework of system model is illustrated in Fig. 2. Two main types of entities are included in our scenario: 1) the central server and 2) the clients. We describe the role of these two types of entities in the system model.

1) *Central Server:* In terms of a functional definition, the server usually plays a coordinator role, mainly defining the federated learning training tasks and controlling the training process, including the processing of intermediate results of the training process and the

interaction with the client. In many scenarios, the server usually represents a third-party requester with some business need for the performance of the final model and tries to use the client's data for training purposes.

2) *Silos:* Each silo individually has a local data set collected from the clients of this silo and trains the local model in federated learning and exchanges the trained model with the server. Although the silo has the initiative to participate and exit the federated learning, it is usually passively controlled by the server for the training task, so its prominent role can be defined as a data provider. In addition, depending on the scenario, the silo's determination of the value of model performance varies. As the overlap between client and server roles increases concerning the business domain, the value of model performance to the client increases.
   In addition, under federated learning fairness, this article divides the clients into two categories.
   a) *Contributed Silos:* Alternatively, "existing silos" refers to those clients that go through multiple iterations so that their cumulative contribution reaches the contribution threshold expected by the server. In this case, the performance of the model obtained by the silo is consistent with its cumulative contribution.
   b) *Limited Contribution Silos:* Alternatively, "new silos" refers to clients that have joined the federated learning process and have little cumulative contribution because they have only participated in a small number of training epochs. In this case, the performance of the global model obtained by them may not match their contribution to the federated learning process.
3) *Clients:* The clients do not directly join the federated learning but act as data providers. The clients contribute their data to the silo, which usually is an institution that provides some services to the clients. Each client's data is split into train and test data parts, which will be merged into silo-level train and test data sets in silo level. Although not directly performing any action during the training, clients are the target that the privacy protection of federated learning should concern, instead of the silo in some scenarios. When a silo undertakes one training step, the silo samples data at the client-level, which means data from a single client is either all adopted or none. Such client-level data sampling design is key when a privacy protection mechanism is introduced and further analyzed, making the DP can be built on the client level.
4) *Model Users:* The model user can access the model at the end of the federated learning training but is not involved in it and cannot see the various data during the training process.

Formally describe the training process of federated learning, initially with a set S of training participants consisting of N silos, each holding a local data set $D_i, i \in [N]$ ($[N] = \{1 \cdots N\}$) which is collected from their clients $c \sim C_s$. Federated learning has $T \in \mathbb{N}$ rounds of training processes. In the training round $t \in \{1 \cdots T\}$, all participants $s \sim S$ undergo a training process based on E epochs of local data D_i on the central model θ^t from the server and submit the trained model θ_i^t to the server, which aggregates it to produce a consistent update as the central model θ^{t+1} for the next round. Extending to a practical application scenario of federated learning, we assume that the complete set of participants S can change dynamically. Initially, the central server may only be able to obtain a subset of participants $\hat{S}$ as the initial participants. Thus, at different stages, a participant's role s_{new} dynamically exists to represent the one that has just joined the federated learning training and has only made a small contribution compared to other participants that have joined for some time. In the subsequent process, as the training process progresses, participants are continuously added to the current training process to improve the convergence performance of the model further. However, the server cannot determine a participant's lifetime, which means the participant can exit the training at any time. In order to simplify this dynamic join–exit process for subsequent analysis and experiments, it is assumed that, except for the initial subset of participants, the interval between the joining of any two participants in the subsequent process is greater than the length of time it takes for the model to reach convergence after the previous participant joins.

B. Design Goals

In this article, we try to design the FedACC framework from different views to make it feasible. The design goal we target list as follows.

1) *Model Performance:* Compared to existing federated learning algorithms, FedACC attempts to achieve fairness guarantees while obtaining relatively competitive performance of the final training model and limiting additional overhead.
2) *Fairness:* FedACC attempts to design a mechanism for the new joining will not receive model rewards that are inconsistent with their contribution to the federated learning system due to forced or malicious withdrawal by the participant in the middle of the training process.
3) *Security:* FedACC should have the characteristic of security inherited from the base federated learning framework. After that, FedACC should be able to cooperate with privacy protection mechanisms, e.g., the DP mechanism, and further enhance the protection strength of clients' privacy.

V. Methodology

In this section, we describe the architecture and fundamental process of FedACC and illustrate the principles and feasibility of the design. The overview of FedACC is shown in Fig. 3. Due to the complexity of realistic scenarios for federated learning, we construct the scenarios on existing data sets on which the experiments are based and describe the methods and parameter design in the construction process.

As the core of the FedACC, we will first introduce the accuracy degrading, which controls the model quality the new silo

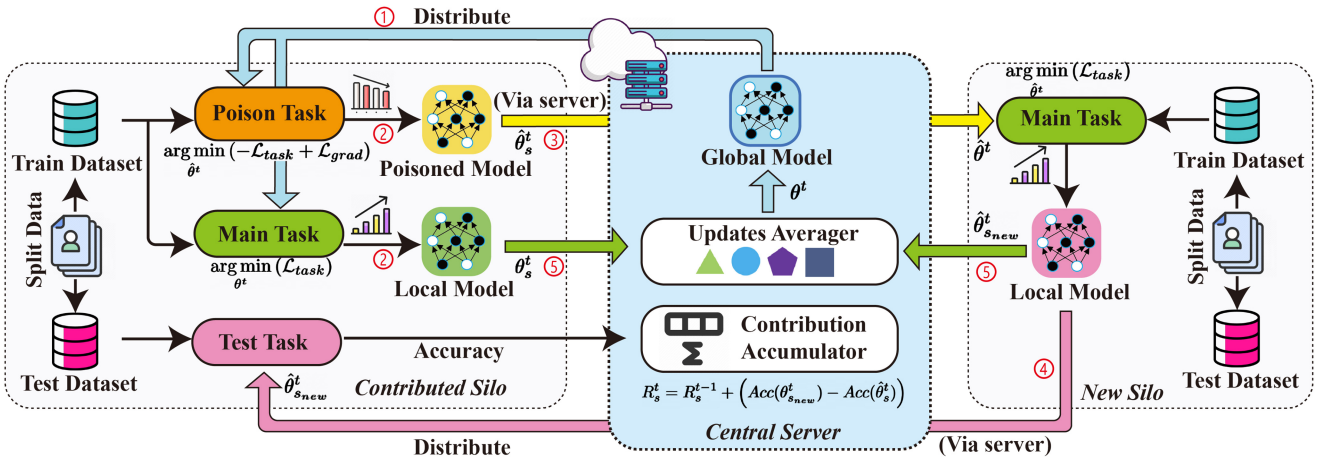


Fig. 3. Overview of FedACC. Take an epoch a new participant exists at the moment t as an example. (1) The server distributes the global model to contributed silos. (2) The contributed silos generate the updated model and the accuracy-decayed model. (3) The contributed silos submit the degraded model to the server, and the server distributes the degraded model to the new silo. (4) The new silos send the updated degraded model to the contributed silos across the server. And the contributed silos submit the accuracy gain compared ACC($\hat{\theta}^t$) to ACC($\hat{\theta}^t_{s_{\text{new}}}$). Then, the server updates the contribution of the new silo. (5) All silos submit their local models to the server, and the server averages the models to generate the global model of the next round θ^{t+1}.

can receive. Based on the degraded model the accuracy degrading gives, the convergence stage stable contribution metric is built to measure participants contribution. And at the last of this section, we combine the subparts as a whole to illustrate the FedACC framework.

A. Accuracy Degrading

FedACC is built on the FedAvg algorithm. In order to achieve fairness in model reward during dynamic join and exit of a participant, there is an algorithm called "accuracy degrading" in the FedACC framework to control the performance of the federated learning global model. For a new participant joining the federated learning training process, there is no initial contribution to it. Its contribution gradually accumulates as it continues participating in the training process iterations. In order to prevent participants from obtaining global models that do not match their cumulative contributions, the server needs to control the precision performance of the distributed models by accuracy degrading when distributing models to newly added participants. In this case, the central server distributes models differently for different participants. The server distributes a full-precision model to participants confirmed to have made contributions above a certain threshold. In contrast, it distributes a "accuracy-degraded" version of the model to new participants whose cumulative contributions are below the threshold. The pseudocode is shown in Algorithm 1. Furthermore, we will explain each part of the accuracy degrading mechanism subsequently.

1) Contribution Metric Required for Accuracy Degrading: The process of accuracy degrading is expressed formally here. For a participant $s \sim S$ in federated learning, its updated model θ^t_c generated at a round t based on the federated learning global model θ^t. Then, the server generates the global model θ^{t+1} for the next round after aggregating the local model parameters of all participants. For simplicity, to be able to measure the contribution of each participant, we assume that the federated learning model has reached a state of convergence on a current

Algorithm 1 Accuracy Degrading

Input: Participant s and client local data D_s; global model θ^t_g; the amount of accuracy decay λ; Total rounds T_p;
Output: Accuracy-degraded model $\hat{\theta}^t_s$

$\hat{\theta}^{(t,0)}_s = \theta^t_g$
for each round $t_p = 1 \cdots T_p$ **do**
  $\mathcal{L}_{grad} = MSE\Big(\mathcal{G}_{task}(\theta^t; D_s),\ \mathcal{G}_{task}(\hat{\theta}^{(t,t_p)}; D_s)\Big)$
  $\hat{\theta}^{(t,t_p+1)} = \hat{\theta}^{(t,t_p)} - \eta \nabla_{\hat{\theta}^{(t,t_p)}}(-\mathcal{L}_{task} + \mathcal{L}_{grad})$
end for
$\hat{\theta}^t_s = \hat{\theta}^{(t,T_p)}$ ▷ Accuracy-degraded model

set of participants before new participants join, i.e., it does not produce higher convergence accuracy until no data from new participants are added. Further, we assume that the interval between the joining of any two participants is longer relative to the time it takes for the previous participant to reach the global model convergence state, such that there is no overlap in the contributions of each participant. In this case, for a certain newly joined participant s, its contribution r^t_s to the federated learning system in a round t can be expressed as $r^t_s = \text{Acc}(\theta^{t+1}) - \text{Acc}(\theta^t)$. Then, suppose that the number of rounds required for a particular participant's joining to bring the federated learning system to the next convergence state is T_s. The cumulative contribution R_s of that participant in the process can be expressed as $R_s = \sum_{t=T^0_s}^{T_s} r^t_s$, where T^0_s is the point in time when participant s joins the federated learning.

2) Accuracy Decay Amount Linked to Contribution: In order to match the model's performance received by a participant with the cumulative contribution it makes, the federated learning central server needs to distribute different performance versions of the global model for newly joined participants at each moment. Formally expressed, at the moment t, the cumulative contribution of participant s is R^t_s, and the server expects to obtain a model $\hat{\theta}$ with a performance decay

of λ relative to the global model, then λ can be expressed as follows:

$$\lambda = \lambda_{\max} \cdot \left(1 - \min\left(\max\left(\frac{R_s^t}{\hat{R}}, 0\right), 1\right)\right) \tag{1}$$

where $\lambda_{\max}$ represents the amount of accuracy decay relative to the global model that should be obtained for a participant with zero contribution. $\hat{R}$ represents how much accuracy gain the server expects a participant joining in the current state should be able to bring. For the selection of $\lambda_{\max}$, we set it to the difference of the global model accuracy at the current moment t relative to the untrained model accuracy, i.e., $\text{Acc}(\theta^t) - \text{Acc}(\theta^0)$, so that a new participant just joining does not get the accuracy gain from the global model trained for a long time directly. Regarding the meaning of $\hat{R}$, since the server knows almost nothing about the participants, it may be difficult to accurately evaluate the performance gain that a participant can bring if it is fully involved in the training process when it joins, so this argument represents more of a model's expectation of the contribution gain that a participant should achieve. And for different convergence stages, the joining of the same participant may also yield different performance gains, so the setting about $\hat{R}$ needs to exploit the server's a priori knowledge about the particular task. To enable contribution evaluation and comparison between participants participating in different convergence stages at different moments, we further design a convergence stage-stabilized contribution metric in the next section.

3) Loss Item $\mathcal{L}_{grad}$ of Accuracy Degrading: In order to perform accuracy degrading on the global model without making the updated model generated by the participant based on the degraded model meaningless for the optimization of the global model, this article designs an accuracy degrading mechanism by combining the objectives of both accuracy degrading and validity of update results. For participant s, there exists a global model θ^t and the cumulative contribution R_s^t of this participant at the time point, and the server designs an optimization task of accuracy degrading to the subset of participants $S\backslash\{s_{\text{new}}\}$ except for the newly added participant s_{new} in addition to the normal training task. Suppose for the participant $s \sim S\backslash s_{\text{new}}$ whose local data set is denoted as D_s, then the loss function of the federated learning target task is denoted as $\mathcal{L}_{\text{task}}(\theta^t; D_s)$, and its gradient corresponding to the backpropagation-based algorithm is denoted as $\mathcal{G}_{\text{task}}(\text{theta}^t; D_s)$. Then, to achieve the validity of the gradient under accuracy degrading, we design a loss function $\mathcal{L}_{\text{grad}}$ based on the gradient similarity so that it ensures that the gradient of the accuracy degrading model based on the participant's local data has as high similarity as possible to the corresponding global model gradient, which is represented as follows:

$$\mathcal{L}_{\text{grad}} = \text{MSE}\left(\mathcal{G}_{\text{task}}\left(\theta^t; D_s\right),\ \mathcal{G}_{\text{task}}\left(\hat{\theta}^{(t,t_p)}; D_s\right)\right) \tag{2}$$

where $\hat{\theta}^{(t,t_p)}$ represents the intermediate result of the accuracy degrading model of the iteration t_p in the gradient descent optimization process rather than the final accuracy degraded model. By combining $\mathcal{L}_{\text{task}}$ and $\mathcal{L}_{\text{grad}}$, (3) gives the optimization terms of the degraded model during the accuracy degrading process. The $\mathcal{L}_{\text{task}}$ could be *cross_entropy*, *MSE*, or any other loss function according to specific training tasks, e.g., CV or NLP

$$\arg\min_{\hat{\theta}^t}\left(-\mathcal{L}_{\text{task}} + \mathcal{L}_{\text{grad}}\right). \tag{3}$$

In this optimization term, the optimization process tries to find a $\hat{\theta}^t$ that has a Δ_{Acc} accuracy decay for θ^t, while having as slight difference as possible about the gradient. For simplicity, we define the loss of whole accuracy degrading as $\mathcal{L}_{\text{degrading}} = -\mathcal{L}_{\text{task}} + \beta\mathcal{L}_{\text{grad}}$, where the weight β is designed for adapting to different scenarios. Due to the privacy constraints of federated learning, this accuracy loss measure usually requires a certain percentage of test data to be set aside for the participant.

The workflow inside the accuracy degrading mechanism is shown in Fig. 4.

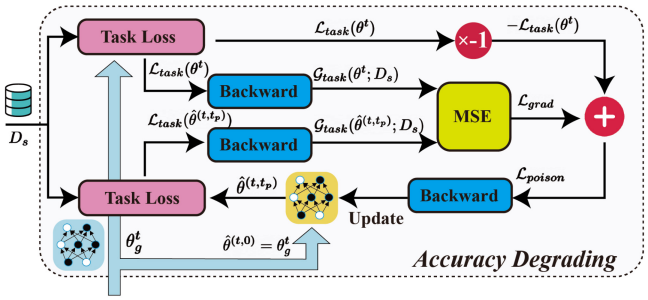


Fig. 4. Workflow inside the accuracy degrading task.

B. Convergence Stage Stable Contribution Metric

The accuracy of the model that the participants should obtain can be equivalently measured using the cumulative contributions of the participants. However, during the training process, since the convergence curve appears nonlinear, participants joining at different moments bring different expectations of accuracy gain even under the assumption of complete data homogeneity. Therefore, the concept of equal contribution corresponds to different magnitudes of accuracy gain for participants with different participation moments. In this case, to effectively measure the specific value of $\hat{R}$ at different moments, we propose the "cumulative gain of single-step accuracy recovery for degraded model" as a mechanism to measure the contribution of the model, incorporating the specific mechanisms of the federated learning and degraded models. In the accuracy degrading mechanism, we propose the $\lambda_{\max}$ parametric to measure the accuracy of the degraded model that a participant should obtain with zero contribution. With this parametric, the initial degraded model obtained by a new participant will have the same accuracy for participants participating at different moments, even though the global model has different accuracy. In this case, for participants participating at different moments, we can more easily use the cumulative accuracy improvement of the new participant for the degraded model to measure the participant's contribution. The workflow inside the convergence stage stable contribution metric is shown in Fig. 5.

1) Definition of Contribution Metric: Formally, for a newly joined participant s_{new}, the degraded model it obtains at

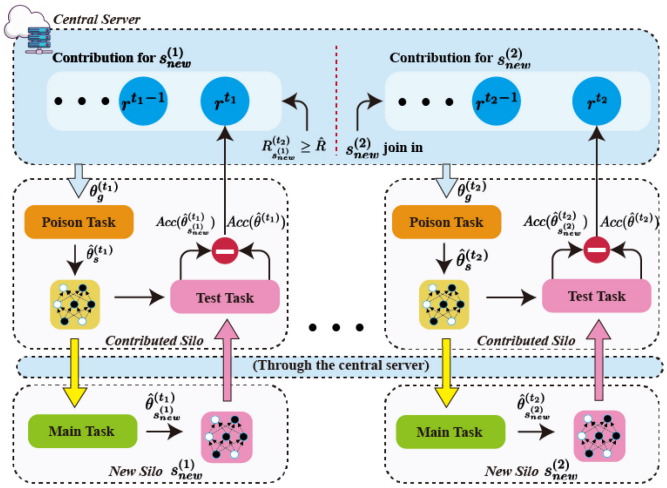


Fig. 5. Workflow of the convergence stage stable contribution metric.

moment t is $\hat{\theta}^{(t)}$ and its precision is $Acc(\hat{\theta}^{(t)})$. Then, the contribution $r_{s_{\text{new}}}^t$ made by this participant at that moment can be expressed as

$$r_{s_{\text{new}}}^t = \text{Acc}\left(\hat{\theta}_{s_{\text{new}}}^{(t)}\right) - \text{Acc}\left(\hat{\theta}^{(t)}\right) \tag{4}$$

where $\text{Acc}(\hat{\theta}_{s_{\text{new}}}^{(t)})$ represents the updated degraded model generated by s_{new} after training based on local data. To mitigate the impact of accuracy fluctuations on the contribution evaluation during training, we propose to crop $r_{s_{\text{new}}}$ to a value greater than zero. Further, the cumulative contribution $R_{c_{\text{new}}}^T$ made by this participant at the moment T from the moment T_0 at which this participant joins the federated learning to some subsequent moment T can be expressed as

$$R_{s_{\text{new}}}^{(T)} = \sum_{t=T_0}^{T} r_{s_{\text{new}}}^t. \tag{5}$$

2) Interrelationship of λ_{max} and $\hat{R}$ Under the Contribution Metric: The setting of the cumulative contribution threshold $\hat{R}$ also determines the proportion of the performance of the global model that the participant per unit contribution can obtain. Assuming a degraded model $\hat{\theta}^{(t)})$ with accuracy $\text{Acc}(\hat{\theta}^{(t)})$ at moment t, for the optimization process of participant s_{new}, the performance gain resulting from its contribution to the degraded model is denoted as $\tau(\hat{\theta}^{(t)}; D_{c_{\text{new}}})$, and $0 \leq \tau(\hat{\theta}^{(t)}; D_{c_{\text{new}}}) \leq \text{Acc}(\hat{\theta}^{(t)})$. With (1), it is possible to give the single-step contribution values accumulated from T_0, at the moment T, satisfying the following relation:

$$\begin{aligned} r^{(T)} &= \tau\left(\lambda_{\max} \cdot \left(1 - \frac{\sum_{t=T_0}^{T-1} r^{(t)}}{\hat{R}}\right)\right) \\ &\leq \lambda_{\max} \cdot \left(1 - \frac{\lambda_{\max}}{\hat{R}}\right). \end{aligned} \tag{6}$$

Summing over (6) yields the relationship between the ratio of the cumulative contribution to the contribution threshold $R_{s_{\text{new}}}^{(T)}/\hat{R}$ and the cumulative contribution threshold $\hat{R}$ at moment T

$$\frac{R_{s_{\text{new}}}^{(T)}}{\hat{R}} \leq 1 - \left(1 - \frac{\lambda_{\max}}{\hat{R}}\right)^{T-T_0+1}. \tag{7}$$

Algorithm 2 FedAAC

Input: Participants set S; Total rounds T; Initial model θ^0
Output: Trained global model θ_g^T

$\theta_g^0 = \theta_0$ ▷ initialize the global model
$R_s^0 = 0$ ▷ Contribution for new participant
for each round $t = 1 \cdots T$ **do**
  $S_t^{degrading} = sample\_silos(S \backslash \{s_{new}\})$
  $\lambda = \lambda_{\max} \cdot \left(1 - \min\left(\max\left(\frac{R_s^t}{\hat{R}}, 0\right), 1\right)\right)$
  for each silo $s \in S_t^{degrading}$ **do**
    $\hat{\theta}_s^t = accuracy\_degrading(\theta_g^t, \lambda; D_s)$
  end for
  $\hat{\theta}^t = \sum \frac{1}{|S_t^{degrading}|} \hat{\theta}_s^t$
  $R_s^t = R_s^{t-1} + \left(Acc(\theta_{s_{new}}^t) - Acc(\hat{\theta}_s^t)\right)$
  for each participant $s \in S \backslash \{s_{new}\}$ **do**
    $\theta_s^t = local\_train(\theta_g^t, D_s)$
  end for ▷ train in silo locally
  $\theta_{s_{new}}^t = local\_train(\hat{\theta}^t, D_{s_{new}})$
  $\theta_g^t = \sum \frac{1}{|S|} \theta_s^t$
end for

The meaning of (7) is that at a particular moment t, with a global model accuracy $\text{Acc}(\theta^{(t)})$ and zero-contribution accuracy decay $\lambda_{\max}$, as the contribution threshold $\hat{R}$ increases, the server and the newly added participants will try to employ more interaction rounds to achieve model convergence. Suppose a participant tries to maximize the accuracy of the degraded model for the next round by maximizing its contribution. In that case, the server can also limit this by setting a larger $\hat{R}$. Thus, this mechanism provides an upper bound for which the fairness risk is manageable, and lowering this upper bound value for the server implies less fairness risk from the early exit of malicious participants.

In general, $\text{Acc}(\hat{\theta}_{s_{\text{new}}}^{(t)})$ is proportional to the quality of the local data of the participants and the overhead spent by the participants to participate in the training. For a lazy participant, its inability to optimize the degraded model effectively makes the server still provide the lazy participant with a degraded model that has a high accuracy decay after multiple iterations.

C. Federated Adaptive Accuracy Controlling

To match the model performance with the cumulative contribution of the participants on each round of accuracy in federated learning, we propose the FedACC framework. Built based on the accuracy degrading mechanism, this article improves on FedAvg to achieve control of the model's accuracy. Its pseudocode is shown in Algorithm 2. In the flow of FedACC, in addition to the normal federated learning training tasks *local_training*, additional server processes *accuracy_degrading* and silos' interactions are added to achieve the accuracy control of the model and the calculation of silos' contributions.

It is also worth noting that the new participant s_{new} in the FedACC algorithm does not refer to a specific silo but more to a participant under the cumulative contribution threshold $\hat{R}$.

As a participant accumulates contributions by gradually participating in training iterations, that participant will also be removed from the role of s_{new}.

1) Silos Sampling for Accuracy Degrading: To make FedACC have as small as possible additional overhead relative to the federated learning process without fairness design while obtaining similar model performance, we additionally design the sampling process for the silos of the accuracy degrading task so that a controlled proportion of silos receive model degrading optimization task in each round. With a lower sampling rate, the additional per-participant overhead of the entire training process is also reduced. Depending on the degree of IID of the client's data distribution, this sampling rate can be scenario-specific adjusted in different scenarios.

2) Analysis of the Complexity of the FedACC: The complexity of FedACC could be analyzed theoretically. First, we assume that the runtime among each step in *local_training* and *accuracy_degrading* is constant, and the complexity of each of them is $\mathcal{O}(|D_s^{\text{train}}|)$. And the complexity of Acc procedure is $\mathcal{O}(|D_s^{\text{test}}|)$. As Algorithm 1 showing, there will be T_p steps in each *accuracy_degrading* on each participant's local. Thus, the complexity of each *accuracy_degrading* whole is $\mathcal{O}(T_p|D_s^{\text{train}}|)$. Let us assume that for a round t, the participant set is S_t, and the silos sampling rate is q. Due to the distributed computing framework, the complexity of accuracy degrading is $\mathcal{O}(T_p|D_s^{\text{train}}|)$ for lines 6–8, $\mathcal{O}(q|S_t||\theta|)$ for line 9 and $\mathcal{O}(|D_s^{\text{test}}|)$ for line 10 in Algorithm 1 each round. And for the main task that target for outputting the final model θ_g^T, corresponding to lines 11–15 in Algorithm 1, its complexity is $\mathcal{O}(|D_s^{\text{train}}|)$ for lines 11–14 and $\mathcal{O}(|S_t||\theta|)$ for line 15 if we assume that there is only one local train step. Thus, the total complexity of FedACC in each round is $\mathcal{O}((T_p+1)|D_s^{\text{train}}| + |D_s^{\text{test}}| + (q+1)|S_t||\theta|)$.

Furthermore, let us consider the assumption that an extra participant joining occurs only after the convergence of θ_g after the previous joining. The complexity spreading out equally to each round of all T rounds is $\mathcal{O}((T_p+1)|D_s^{\text{train}}| + |D_s^{\text{test}}| + (q+1)|S_T||\theta|)$ which is proportional to the maximum participants $|S_T|$ with the assumption that the convergence rounds after each participant joining in are the same.

VI. Security Analysis

Our FedACC framework tries to design a mechanism to ensure model reward fairness. Besides considering the fairness of federated learning, combining it with an external privacy-protect mechanism, e.g., DP, is becoming a fundamental operation in practice. In this section, we combine FedACC with DP further to enhance security and fairness assurance.

Although an additional privacy protection mechanism is essential to federated learning, it is still an optional part of FedACC implementation, which means FedACC can still achieve its fairness goal without a privacy protection mechanism. Because of it, we introduce the DP privacy protection mechanism as a separate part, which will also enhance the extensibility of FedACC when combined with other protection mechanisms.

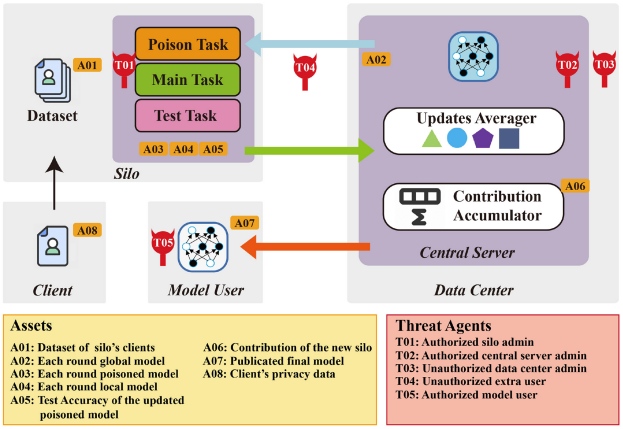


Fig. 6. Framework of threat model.

A. Threat Model

Based on the behavior and ability to perform attacks of each entity involved in the whole federated learning process, we try to divide them into several categories. The threat model framework is shown in Fig. 6.

From the views of threat agents, we could analyze the weakness of the FedACC framework.

1) *The Authorized Central Server Admin (T02) and the Unauthorized Data Center Admin (T03):* The authorized central server admin is the one who has full access to the central server and can obtain any data that is sent to and distributed from the server. The unauthorized data center admin can control all servers in the data center, including the one performing federated learning. Thus, in this view, we could consider them the same threat entity. In federated learning, the central server will design and distribute the task the server wishes to perform honestly under the semi-honest assumption. In the training process, the central server admin can obtain the updated models from silos and distribute the averaged global model to each silo as the federated learning defines. Thus, the admin could get any information that exists directly in the model parameters, e.g., the statistical information, and should be inferred further by performing the methods, including membership attack [33] and reconstruction attack [34].
2) *The Authorized Silo Admin (T01):* The authorized silo admin is the one who has full access to the silo and can obtain any data that is sent from and distributed to the silo. Besides, the silo admin has access to its clients' private data. From the view of privacy, the silo admin can access the global model that is averaged on the local model submitted by each silo. To obtain the privacy information from the averaged model, the silo admin can perform the membership attack [33] and reconstruction attack [34]. However, due to the averaging operation performed on the server, the attack strength from the silo will be weakened compared to the one on the server.
3) *The Unauthorized Extra User (T04):* The unauthorized extra user usually does not have access to the training

process. However, suppose the data is exchanged in the training process without being encrypted. In that case, the extra user may obtain all the exchanged information, e.g., the global model and silo's update model, which means all attacks the central server admin could perform are available. Nevertheless, using an encryption method like SSL can eliminate the risk.

4) *The Authorized Model User (T05):* The model user usually exists after the entire training process is over. Usually, the only information the model user can access is the final model. Due to the invisibility of gradients and the iterative averaging process, the reconstruction attack is hard to perform on model users. However, the model users can perform the membership attack if they have full access to the published model.

We could give the threatening ranking of threat agents for FedACC: $T02 \approx T03 > T05 > T04 > T01$. Because $T05$ is at the end of the training, the average overlay can better hide the training process's information. For $T04$, usually simply applying traffic encryption can be defended. Although $T01$ has direct access to the user's data, the defense against $T01$ is usually impossible at the federated learning level. Thus, in the following sections, we try to apply the DP to defend against the threat agents $T02$ and $T03$.

B. Standalone Module With Differential Privacy Assessment

The client-level DP [30] is the DP model we are mainly based on, in which the protection of DP is applied at the client level. Suppose we try to introduce DP to FedACC and analyze the security characteristic of FedACC under DP. In that case, one important thing is that we should figure out how each part of FedACC consumes the DP budget [28], [35]. As Algorithm 2 shows, each process that will exchange parameters with participants will consume the DP budget. For convenience, we could divide the FedACC into three parts from the perspective of DP.

1) *Average for Global Model:* The *local_train* and θ_s average process. This process is that the training model solves problems outside the federated learning training process, such as image classification. This part is the same as the FedAvg one. To apply DP to this process, as the approach implemented in [30], we could add Gaussian noise to the θ_g^t after θ_g^t built by average every θ_s^t.
2) *Average for Degraded Model:* The *accuracy_degrading* and $\hat{\theta}_c$ average process. This process degrades the global model to generate an accuracy decay version to match the contribution of the new participant from the view of the model reward. This process could be seen as a FedAvg process nested within the FedAvg process of the global model. Thus, it could apply DP by the same approach as the average for the global model, and the noise will be added to the $\hat{\theta}^t$.
3) *Accuracy of Updated Degraded Model:* The process calculates the accuracy of the updated degraded model $\theta_{s_{\text{new}}}$ for evaluating the contribution made by the new participant. With the characteristics and limits of federated learning, we assume there is no auxiliary data set for model performance testing in the central server. Thus, to test the accuracy of the updated degraded model $\theta_{s_{\text{new}}}$ for contribution measure, we should design it as a general federated process as the two processes above.

C. Gaussian Mechanism on Client-Silo Level

As the approach of the client-level Gaussian mechanism proposed by [30], cutting the update and adding Gaussian noise satisfies the following equation:

$$\Delta_{\theta^t} = \frac{1}{|C|}\left(\sum_{c\sim C}\frac{\Delta_{\theta_c^t}}{\max\left(1, \frac{\|\Delta_{\theta_c^t}\|_2}{\tau}\right)} + \mathcal{N}\left(0, \sigma^2\tau^2\right)\right) \tag{8}$$

in which τ presents the sensitivity of the $\Delta_{\theta_c^t}$, and σ is the base scale of the Gaussian noise.

For the Gaussian mechanism, selecting a scenario-suitable parameter, e.g., the update cutting threshold S and noise scale σ, could measure the DP broken probability δ with fixed DP budget ϵ. Based on the Gaussian mechanism, we can use moments accountant proposed by [28] to track the DP broken probability δ, accumulated during the training process. Then for the *Average for global model* and *Average for degraded model*, we could get two mechanisms $\mathcal{M}_g$ and $\mathcal{M}_p$ which represent the *Average for global model* and *Average for degraded model* after being applied Gaussian mechanism, respectively. And $\mathcal{M}_g$ and $\mathcal{M}_p$ will satisfy (ϵ, δ_g) and (ϵ, δ_p), in which δ_g and δ_p are add up by moments accountant in each step.

However, to defend against the semi-honest central server and fit the system model we target, this article will apply the client-level DP proposed by [30] to the client-silo level. In original client-level DP, no additional privacy protection method, e.g., DP mechanism, protects privacy information in the data clients send to the server from any potential attack token by the server but defends against the model users. To expand the protected area covered by the DP mechanism, we redirect the "clients" role to the clients in each silo and the "server" role to each silo. The reason why we could apply such a hierarchical transformation is, according to the theory, if a process $\mathcal{M}$ with input D satisfies ϵ DP. Any process $\mathcal{M}'$ with input $\mathcal{M}(D)$ does not produce a process that is below the ϵ protection strength through the combination. Another reason related to our system model is that since the silos often are the owner of the users' data, there is no need and no meaning to add any protection mechanism against silos. Based on this scenario-related assumption, we can transfer the equation under the client–server architecture to the client-silo-server architecture, which can be explained as follows:

$$\begin{aligned}\Delta_{\theta_s^t} &= \Delta_{\theta_s^t}\left(\theta_g; c \sim C_s\right) + \frac{1}{|C_s|}\mathcal{N}\left(0, \sigma^2\tau^2\right)\\ \Delta_{\theta^t} &= \frac{1}{|S|}\sum_{s\sim S}\Delta_{\theta_s^t}\end{aligned} \tag{9}$$

in which C_s represents the clients set in silo s, $\Delta_{\theta_s^t}$ represents the update of silo s, and S represents the silo collection.

Since the silo has full access to devices' data at the device-silo level, the average process of (8) can be simplified to the gradient descent process. As (9) designs, the calculation of $\Delta_{\theta_s^t}$ is performed at the silo level, so the calculation process is invisible to the server. And according to the client-level DP mechanism, $\Delta_{\theta_s^t}$ itself will satisfy (ϵ, δ) DP. Thus, after the moment that $\Delta_{\theta_s^t}$ is calculated, subsequent calculation at the server level is just performed on privacy-protected data.

D. Differential Budget Evaluation of Module Combinations

For the process $\mathcal{M}_g$, $\mathcal{M}_p$, and $\mathcal{M}_a$, we can implement them according to the calculation form of $\Delta_{\theta_s^t}$ as (9). Thus, for the whole FedACC training process, we can abstract it as a process $\mathcal{M} = \mathcal{M}_g + \mathcal{M}_p + \mathcal{M}_a$. Before analyzing the characteristic DP through the process $\mathcal{M}$, it is important to analyze how the DP broken probability δ is accumulated for a specific client in one silo. Depending on whether a silo is a contributed silo or a limited contribution silo, the calculation of δ for the silo's clients is different, as explained below.

Besides, we introduce the DP mechanism's sequential combination theorem and parallel combination theorem.

Lemma 1 (Sequential Combination Theorem [36]): Suppose a mechanism M consists of a sequence of submechanisms $\{M_1, \ldots M_n\}$ serially combined, and each subprocess satisfies (ϵ_i, δ_i) DP by itself. For this case, it is noted that the sequence $M_i(X)$ should satisfy the DP of $(\sum_i^n \epsilon_i, \sum_i^n \delta_i)$, where X is the output of the previous item.

Lemma 2 (Parallel Combination Theorem [36]): Suppose a mechanism M consists of a sequence of submechanisms $\{M_1, \ldots M_n\}$ combined in parallel, and each subprocess satisfies ϵ DP by itself. Suppose D_i is any data set that satisfies the input requirements of mechanism M_i. For this case, the sequence $M_i(X \cap D_i)$ should satisfy ϵ DP, where X is the output of the previous item.

For contributed silo, there exists $\mathcal{M}_g$, $\mathcal{M}_p$, and $\mathcal{M}_a$. Since $\mathcal{M}_g$ and $\mathcal{M}_p$ are performed on the same training data set for each client, according to (1), the combination process $\mathcal{M}_{g\&p}$ of $\mathcal{M}_g$ and $\mathcal{M}_p$ will satisfy $(\epsilon_g + \epsilon_p, \delta_g + \delta_p)$ DP. For $\mathcal{M}_a$, since it is performed on the test data set separated from the train data set, according to (2), if combines $\mathcal{M}_a$ with $\mathcal{M}_{g\&p}$, the whole process $\mathcal{M}$ will just satisfy $(\max(\epsilon_g + \epsilon_p, \epsilon_a), \max(\delta_g + \delta_p, \delta_a))$ DP. But the fact is that since $\mathcal{M}_a$ and $\mathcal{M}_p$ are operated simultaneously, so the $\max(\epsilon_g + \epsilon_p, \epsilon_a) = \epsilon_g + \epsilon_p$ and $\max(\delta_g + \delta_p, \delta_a)) = \delta_g + \delta_p$ will always be satisfied, so $\mathcal{M}$ will satisfy the $(\epsilon_g + \epsilon_p, \delta_g + \delta_p)$ DP.

VII. Experiments

In this section, we verify the effectiveness of the algorithm designed in this article by performing it in simulation environments. To demonstrate the effectiveness, we compare the performance of FedACC and existing algorithms. Further, we validate some of the ideas revealed in the experiments.

A. Experimental Setup

1) Federated Data Sets: In our experiments, we use existing data sets to construct the local data of clients that are available in federal learning. To keep the experiment simple without losing generality, we choose the Fashion-MNIST [37], Rice [38], and CIFAR10 [39] data sets as the data sources in our experiments here. The properties of the data sets are shown in Table II. By dividing the data set into different subsets at a fixed size, we can simulate a scenario where several silos have their silo-local data collected from their clients.

TABLE II
Federated Data Sets in Experiment

Dataset	Samples	Classes	Images size
Fashion-MNIST	Train 60,000 / Test 10000	10	28×28 Gray
Rice	Train 3840 / Test 960	5	250×250 RGB
CIFAR10	Train 50000 / Test 10000	10	32×32 RGB

2) Implementation: To simplify the complexity of the practical scenario, we assume that the federated learning training process initially exists for a finite proportion of the participant subset $S_0 \in S$, in which we set to $|S_0| = 1$ in the absence of a specific specification of the size of S_0. Also, for simplicity, we assume that the local data set D_s, $s \sim S$ has an equal size on different silos, and by default, we set the size of the local data set to $|D_s| = 200$. By default, the SGD optimizer with $lr = 7 \times 10^{-3}$ will be used in the experiments, and its other parameters are kept at their default settings. For reducing the DP budget consumption, the sample rate q of sampling clients in $\mathcal{M}_g$ and $\mathcal{M}_p \& \mathcal{M}_a$ is $q = 0.05$ and $q = 0.1$, respectively. To simplify the complexity, we set the cumulative contribution threshold $\hat{R} = 2$, and we terminate the accuracy degrading mechanism early when $R_{c_{\text{new}}} = 50\%\hat{R}$ to enhance the stability of contribution evaluation in practical scenarios. In addition, to simplify the difficulty of the training process while maintaining generality, we use the vgg11_bn pretraining model in the PyTorch [40] framework to encode the data set.

B. Contribution Metric Compared to Existing Methods

In the FedACC framework, the contribution metric component plays an important role in instructing how much of the decay the accuracy degrading method should perform. In this article, we design a convergence stage stable contribution metric based on the accuracy degrading method and try to make the contribution metric suitable for the problem scenario and the system model. In this experiment, we try to compare the contribution metric of FedACC with existing methods. The state-of-the-art contribution-metric methods we select to compare with are the leave-one-out (LOO) [41] and data sharply (DS) [27], which are both metrics of the contribution made by one participant on its data. We apply the three methods to the DP-protected federated learning with the accuracy degrading mechanism, which is necessary to ensure fairness. For the implementation of this experiment, an initial of participants $|S_0| = 1$ is selected, and an extra participant is added, at which moment the contribution of the new participant is measured. Furthermore, to show the methods' behaviors, we construct two types of subexperiments, i.e., the helpful and the unhelpful, respectively, which represent whether a new

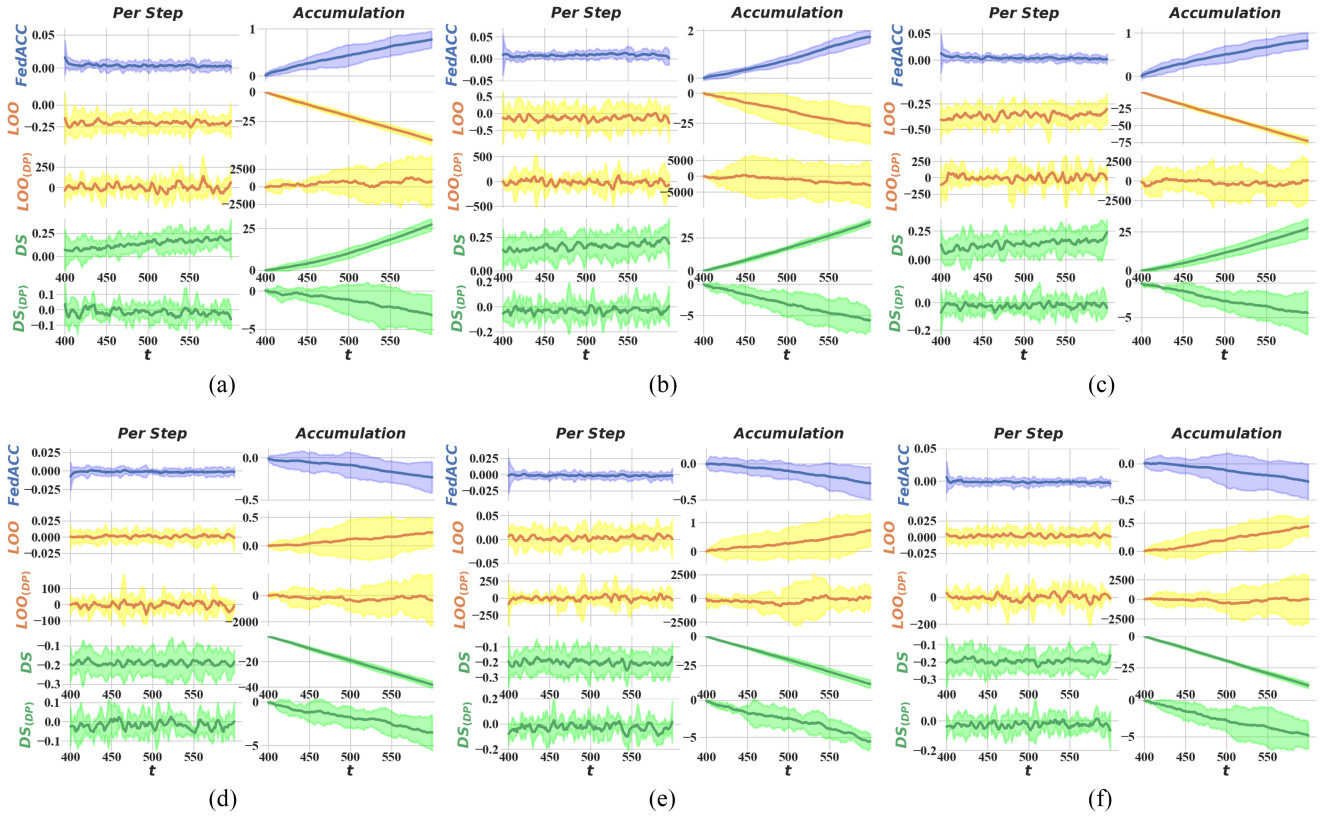


Fig. 7. Accumulation contribution provided by FedACC, LLO, and DS after adding the extra participant join when $T_0 = 400$. $\text{LOO}_{(\text{DP})}$ and $\text{DS}_{(\text{DP})}$ represent additional DP protection of the client's data exchanging process, which is necessary according to the LOO and DS algorithms when combined with federated learning. (a) Helpful; Fashion-MNIST. (b) Helpful; Rice. (c) Helpful; CIFAR10. (d) Unhelpful; Fashion-MNIST. (e) Unhelpful; Rice. (f) Unhelpful; CIFAR10.

participant has a local data set that is helpful for the task of federated learning. For the data set of the unhelpful participant, we replaced its local data with randomly generated ones.

The contribution calculated by each method is shown in Fig. 7. The results are shown with the per-step and accumulation forms. It is worth noting that the contribution values given by LOO and DS are not directly about the expected improvement in model accuracy, but more about evaluating the relative validity between the data. Therefore, it is meaningless to compare the values between these metrics. However, we can still determine whether the client data contribution is at the average level of other clients by comparing the relative values between client data points under one metric. And this level of contribution can be determined by reaching a consensus in advance on an initially trusted limited number of participants. The results show that the contribution-measure method of FedACC correlates more to the helpfulness of the participant. Moreover, in some scenarios, the LOO methods cannot distinguish whether a client is a contributor by the positive or negative contribution value.

In essence, since the data evaluation methods of LOO and DS rely on the comparison between the model and the data, if the data are small and the model is biased, then the newly added data will be in a nonfitted state to the model, which leads to the situation that the data are stochastic or even wrong to the model. What is worse, due to the limitation of the architectures of LOO and DS, these two methods rely on the model, clients' training data for the test, and reliable test data simultaneously to calculate the new participant's data value. An additional DP protection is needed to protect participants' privacy when exchanging these data. And the noise further degrades the data validity provided to LOO and DS. Thus, for the existing methods that can ensure the fairness of model reward in each round, i.e., the HFFL, it may be challenging to obtain a framework with similar functionality to FedACC by directly combining the LOO or the DS methods.

C. Adaptive Accuracy Control With Participant Mid-Join

In this experiment, we try to simulate the case where additional participants join when the model reaches convergence after a relatively long period of federated learning training. In this case, the contribution of the additional participants to the federated learning training increases gradually as their participation in the training iterations progresses. In order to ensure the fairness of the model reward, the server adaptively generates an accuracy-degraded version of the global model matching the contributions of the newly added participant according to the FedACC framework and sends it to the new participant for local training. To illustrate the specific behavior of the FedACC framework during this process, we try to visualize the model convergence curve for this additional participant joining process and the relationship between the accuracy degrading and the participants' cumulative contributions.

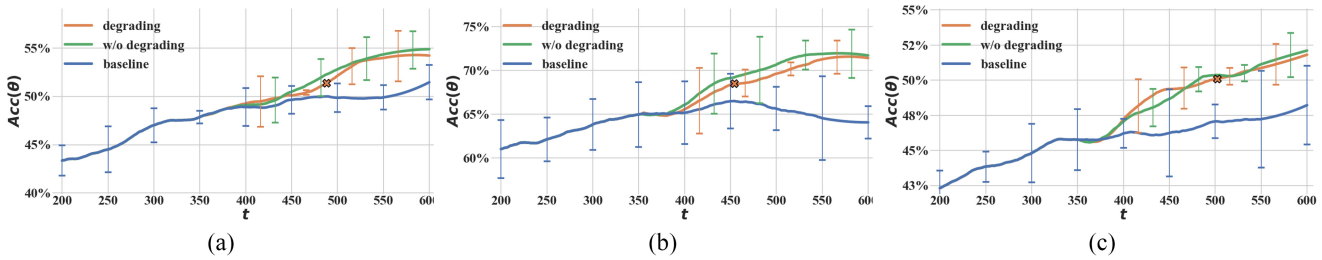


Fig. 8. Convergence curves for additional participant joins at $T_0 = 400$. The cross markers represent the moment when $\lambda = 0$ in the accuracy degrading scenario. (a) Fashion-MNIST. (b) Rice. (c) CIFAR10.

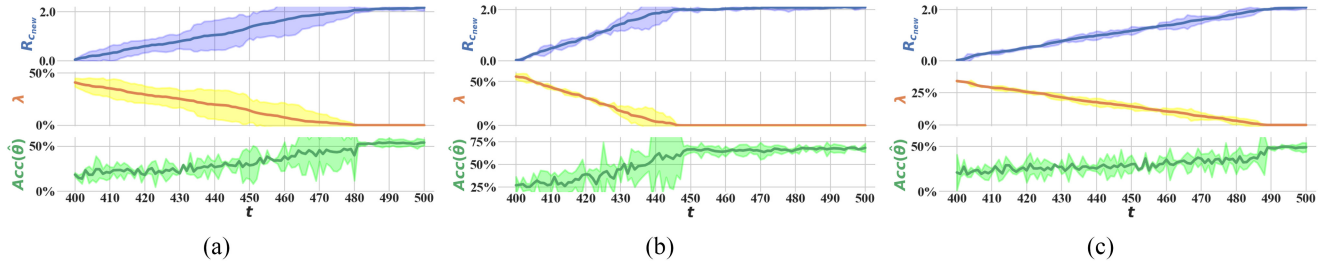


Fig. 9. Relationship between λ and $Acc(\hat{\theta})$ after adding the extra participant join when $T_0 = 400$. (a) Fashion-MNIST. (b) Rice. (c) CIFAR10.

The experimental results are shown in Fig. 8. It can be seen that for the baseline curve, with an initial participant size of $|S_0| = 1$, the federated learning training process has reached full convergence at $t = 100$, after which further learning causes even a slight degradation of the model performance due to the effect of overfitting. At the $t = 200$ moment, for the degrading and w/o degrading curves, adding additional participants further increases the global model's performance.

We experimentally verified the relationship between the server's expected accuracy decay magnitude λ and the factual accuracy of the degraded model received by the participant after adding the participant in the degrading scenario in Fig. 8. The experimental results are shown in Fig. 9. It can be seen that in the case of additional participants just joined, corresponding to $R_{c_{\text{new}}} = 0$ with approximately $\lambda > 40\%$ accuracy decay, the experimental scenario is verified by the global test data set also proves that the actual accuracy value of the model after degrading with this accuracy does roughly match the desired accuracy decay amount. As the contributions from subsequent processes accumulate, the server gradually reduces the accuracy degrading amount of the global model distributed to s_{new}. It can be seen that λ drops to zero as the cumulative contribution of s_{new} reaches the threshold, at which point, s_{new} can obtain a global model with complete accuracy. At this point, s_{new} can be removed from the role of a "new participant" and added to the normal set of participants S. After this point in time, the detachment of the participant does not lead to an imbalance in the fairness of federal learning because the model payoffs of the participant match their cumulative contributions.

D. Performance Comparison of Multiparticipant Serial Join

To show the convergence of the FedACC method, we consider the model learning process on multiparticipant sequential joins over the adaptive accuracy control mechanism of single-participant joins. In the multiparticipant join scenario, we assume that only at most one participant joins at the moment and that no other participant joins until the model training reaches convergence after that participant joins. In the initial case, we set $|S_0| = 1$. In the subsequent process, we set the interval $\Delta t = 400$ between two participant joins to ensure that the convergence state can be reached. To illustrate the adaptability of our method for different scenarios with different participants, we compare the final accuracy with the existing FedAvg method in experimental scenarios with different numbers of subsequent participants.

To compare the state-of-the-art methods under the same problem background, we consider combining the HFFL with an existing contribution metric, e.g., the DS and the LOO. Moreover, the HFFL method will retrain the model when a new participant joins. To ensure the model reward fairness, each contributed participant will add noise to its data to decay the value for training. With the new participant contributing to the federated learning, the scale of noise other participants add to their data will decrease. To make the value of noise-added data linearly correlated to the contribution measured by the DS, the power of the noise scale should be linearly inversely proportional to the accumulation contribution of the new participant. In this experiment, with the Gaussian noise, the initial noise scale is $\sigma = 2^4$. The contribution threshold can be measured by one of the initial participants by joining for certain rounds.

The results are presented in Table III and Fig. 10. It shows that for different participant set sizes $|S|$, our approach can approach the FedAvg algorithm implemented by the no-fairness mechanism at an average level. Moreover, compared with the HFFL+DS method, FedACC can gain about 10%–20% accuracy with the number of silos increasing. The nonlinear contribution measurements for different amounts of silos may lead to poor performance of the HFFL.

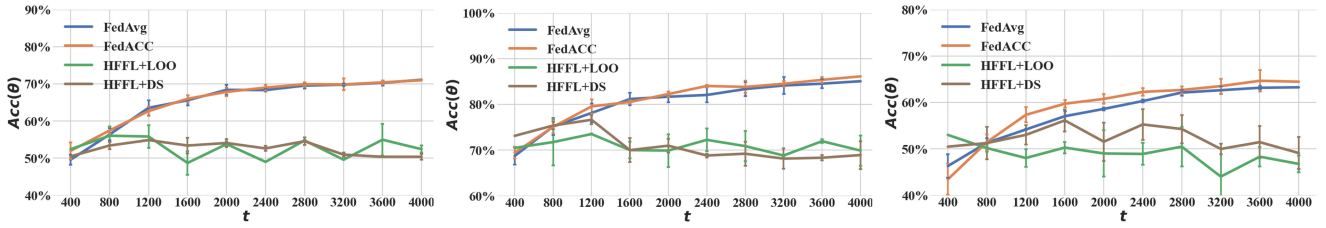


Fig. 10. Convergence curves for multiparticipant serial join. An extra participant joined for each $t = 400 \times i$, where $i \in \{1, \ldots, 10\}$. Each curve represents the sequence of the global accuracy before the next participant joins. (a) Fashion-MNIST. (b) Rice. (c) CIFAR10.

TABLE III
MODEL ACCURACY(%) FOR DIFFERENT PARTICIPANTS AMOUNT. (A) WITH FASHION-MNIST DATA SET. (B) WITH RICE DATA SET. (C) WITH CIFAR10 DATA SET

(a)

#Silos	1	2	5	10
FedAvg	49.2±2.1	55.7±1.8	68.2±1.7	70.8±0.9
HFFL+LOO	52.4±3.7	56.1±2.5	53.8±0.7	52.4±1.0
HFFL+DS	50.5±0.1	53.4±1.0	54.1±1.1	50.4±0.8
FedACC	51.8±1.8	57.3±1.5	68.1±1.5	71.2±0.8

(b)

#Silos	1	2	5	10
FedAvg	67.6±1.8	75.1±2.4	81.6±0.7	85.3±1.6
HFFL+LOO	70.5±2.5	71.8±5.1	69.9±3.6	69.9±3.3
HFFL+DS	73.1±1.8	75.4±0.8	71.0±1.5	68.9±3.0
FedACC	69.7±2.1	74.4±0.8	81.9±0.3	86.0±0.6

(c)

#Silos	1	2	5	10
FedAvg	46.2±2.3	51.3±1.0	58.1±0.1	63.2±0.4
HFFL+LOO	53.0±1.0	50.2±0.8	49.0±5.0	46.8±1.8
HFFL+DS	50.5±4.0	51.3±3.5	51.6±4.1	49.1±3.5
FedACC	43.1±3.8	51.0±1.8	60.5±1.0	64.2±2.0

TABLE IV
DEGRADATION REMOVAL PERFORMED LOCALLY BY THE PARTICIPANTS ON DIFFERENT TRAINING PHASES. (A) WITH FASHION-MNIST DATA SET. (B) WITH RICE DATA SET. (C) WITH CIFAR10 DATA SET

(a)

#Silos	1	2	5	10
Global Acc	48.2±0.9	52.9±1.4	63.7±2.5	69.7±1.6
Degraded Acc	12.8±1.3	17.7±1.8	42.7±1.1	47.1±4.2
Recoverd Acc	45.8±1.4	48.2±2.7	49.0±1.4	58.6±1.0

(b)

#Silos	1	2	5	10
Global Acc	60.8±0.9	70.0±1.1	81.6±0.7	84.6±0.5
Degraded Acc	21.5±0.5	10.0±1.6	26.7±4.5	23.0±2.8
Recoverd Acc	61.5±2.1	56.1±1.9	58.8±0.4	65.4±1.9

(c)

#Silos	1	2	5	10
Global Acc	40.3±0.8	49.8±0.7	58.5±0.6	60.4±1.4
Degraded Acc	15.1±3.2	28.3±3.5	46.4±3.5	43.6±1.3
Recoverd Acc	43.7±0.8	47.6±0.9	51.4±1.4	50.8±2.4

E. Resistance to Local Degradation-Remove on the Silo Side

To illustrate the robustness of the model degrading approach proposed in this article, we consider that participants may try to remove the added accuracy degradation by fine-tuning after receiving an accuracy-degraded model and try to recover the global model with complete accuracy. In this case, it is assumed that there is no complicity between participants and that an individual participant s has only its own local data D_s. For the fine-tuning task, it can be represented as an attempt by the participant to optimize the loss term $\mathcal{L}_{\text{task}}(\hat{\theta}; D_s)$ with the expectation that the converged result of the optimization will recover the complete model accuracy as much as possible. To this end, we experimentally validate the results for different training phases in which a new participant is added for degrading removal on the initial degraded model it receives.

The experimental results are shown in Table IV. For the degraded model $\hat{\theta}$ generated based on the global model θ, the participant can recover some accuracy by fine-tuning based on its local data. Then under the approach of this article, we expect that the participant cannot get additional accuracy gain from the degraded model. As can be seen, in the experimental results, the accuracy $\text{Acc}(\hat{\theta}_{s_{\text{new}}})$ recovered by the new participant does not exceed the global model accuracy $\text{Acc}(\theta)$ at the current moment in most cases. In addition to the effect of the accuracy degrading method in this article, the presence of overfitting effects in the federated learning scenario, where participants usually have a limited number of samples, also leads to the fact that it may be challenging to remove accuracy degrading for a single participant. This experimental result illustrates that under this article's fairness framework design, it is possible to ensure that new participants cannot break the fairness guarantee with local degradation removal.

F. Analysis of $\mathcal{L}_{degrading}$ Weight Terms β

In the model degrading design of FedACC, we provide a weight term β for $\mathcal{L}_{\text{degrading}}$ to ensure the adaptability of FedACC for different scenarios. To illustrate the impact of this weight term on FedACC and to show the necessity of $\mathcal{L}_{\text{degrading}}$ loss term design for model degrading as well as FedACC mechanism, we validated the effect of different β weight term value settings on the convergence of FedACC on different data sets. The experimental parameter settings are the same as those of the corresponding experimental setting in Fig. 8. For different values of β, we simply and without losing generality take $\beta \sim \{0, 0.1, 1, 10\}$ for the experiments, respectively. In particular, for the case $\beta = 0$, the equivalent (3)

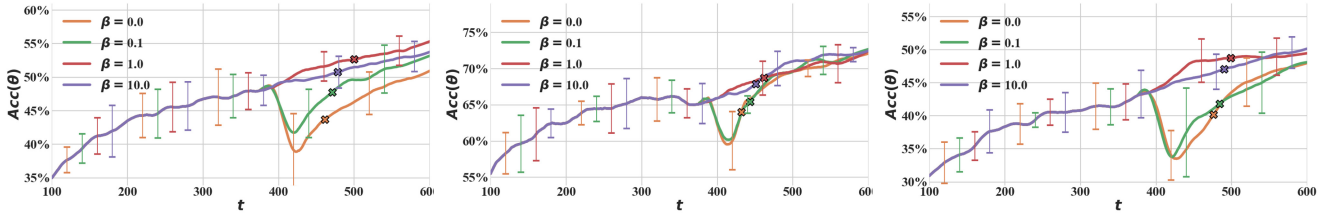


Fig. 11. Effect of $\mathcal{L}_{\text{grad}}$ weight term β on FedACC convergence. The cross markers represent the moments where the participants reach the contribution threshold. (a) Fashion-MNIST. (b) Rice. (c) CIFAR10.

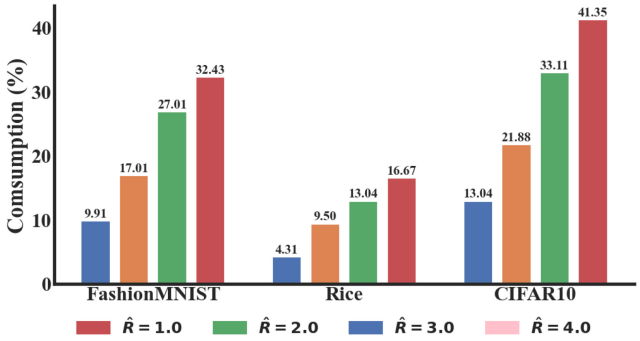


Fig. 12. Proportion of DP budget consumed by $\mathcal{M}_p\&\mathcal{M}_a$.

becomes $\arg\min_{\hat{\theta}^t}(-\mathcal{L}_{\text{task}})$ which is the ablation experiment on $\mathcal{L}_{\text{degrading}}$.

The experimental results are shown in Fig. 11. It can be seen that the case of $\beta = 0$ shows poor convergence in most scenarios. The reason for this is that removing the $-\mathcal{L}_{\text{degrading}}$ loss term and optimizing only the $-\mathcal{L}_{\text{task}}$ loss term will result in the applicability of the gradient generated by the participant on $\hat{\theta}$ to the global model θ is not guaranteed. Comparatively, from the experimental results, the taking of $\beta = 1$ is better than the results produced by $\beta = 0.1$ in most scenarios. For the setting of $\beta = 0$ and $\beta = 0.1$, it could be seen that the global model accuracy $Acc\theta$ at the moments that the model degrading stopped are both under the one just the new participant joined. For the setting of $\beta = 10$, the performance gain is slight compared to $\beta = 1$ and may lead to incomplete accuracy decay. Therefore, we generally recommend setting $\beta = 1$ or conducting further experimental validation in specific scenarios.

G. Analysis of DP Budget Consumption

In the security section, the characteristics of DP are analyzed, especially, how the DP broken probability δ is accumulated through the FedACC training process. In this experiment, how the δ from $\mathcal{M}_g$, $\mathcal{M}_p$, and $\mathcal{M}_a$, respectively, are accumulated under different FedACC parameters setting is shown. In practice, the parameter that would be changed under different scenarios is the contribution accumulation threshold $\hat{R}$, which controls how imbalances the fairness risk of a participant exiting the federated learning without making enough contribution. Since the $\mathcal{M}_p$ and $\mathcal{M}_a$ are always performed simultaneously, δ from $\mathcal{M}_p$ and $\mathcal{M}_a$ always the same. Thus, it will be merged into one data in the result. In the experiment, the FedACC is performed under $\hat{R} \sim \{1.0, 2.0, 3.0, 4.0\}$, respectively, which reflects different sensitivity about the model-reward fairness under different scenarios. For the experiment implemented, it is conducted that a single participant joined the federated learning, which is trained for $T = 400$ epochs with initial participants set $|S_0| = 1$.

The result is shown in Fig. 12. It could be seen that with $\hat{R}$ set from 1.0 to 4.0, the proportion of DP budget consumed by $\mathcal{M}_p\&\mathcal{M}_a$ increases with it, which is nearly linear. It can be explained that as the value of $\hat{R}$ increases, the new participant will take more epochs to make more contributions to the global model, reflecting on the degraded model's performance improvement. Thus, there will be more information, i.e., the new participant's update, to be exchanged in more rounds, which results in a higher δ accumulation. By the way, it can be seen that the proportion of DP budget consumed by $\mathcal{M}_p\&\mathcal{M}_a$ is different across different data sets. It could be explained that compared to a relatively easy data set, federated learning will take less useful information in each round under a harder data set. Thus, although there is the same $\hat{R}$ setting, the rounds the new participant takes to reach the contribution threshold will also be different.

H. FedACC Additional Overhead Under Sample Rates

FedACC includes an adjustable silos sampling process to sample silos undertaking the accuracy degrading task. In some scenarios, e.g., when there are relatively many initial clients or when a single client has relatively large amounts of data, reducing the sample rate could split the whole accuracy degrading task to smaller pieces to each contributed silos. And we could analyze the relationship between sampling rate and participants' overhead, e.g., communication overhead. To implement such a scenario, we assume that the initial silos are relatively large, i.e., $|S_0| = 100$, to show the influence of the sample rate. And a new joining silo should take relatively large number of rounds, i.e., $T = 200$, to make the global model convergence. For simplicity, we use a 3 layer $(1000 \times 100 \rightarrow 100 \times 100 \rightarrow 100 \times 10)$ FC model.

The average communication overhead per round is shown in Table V. We assume the FedAvg part samples all silos each round, thus, has a constant communication. For the accuracy decay part, although one sampling communication overhead is constant for a silo, but the share expenses will decrease with a small sample rate, due to the decreased sampling times through the whole T.

TABLE V
FEDACC COMMUNICATION OVERHEAD TO FEDAVG

Sample rate	0.01	0.1	0.5	1
FedAvg	0.84MB			
FedACC	0.857MB	0.933MB	1.27MB	1.70MB
Overhead(%)	101%	110%	150%	200%

VIII. CONCLUSION

Due to the dynamic and flexible nature of federated learning, it may be difficult for existing federated learning solutions to ensure that participants do not unintentionally or maliciously exit early during the training process and acquire global models that do not match their contributions due to participant joining and exit uncertainties. In this article, we propose the FedACC approach to ensure that newly joined participants do not prematurely acquire a federated global model that does not match their contributions by implementing adaptive model accuracy control based on their cumulative contributions and that participant updates based on degraded models are effective in improving the global model. We experimentally measure the different properties of the FedACC approach and verify that the FedACC approach can achieve a competitive model performance relative to the existing widely used FedAvg federated learning approach while achieving model reward fairness.

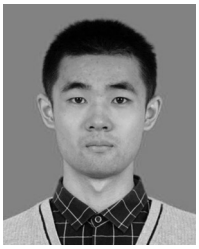
Due to the limitation of the length of this article and the experimental environment, and the complexity of federated learning itself, we believe that there are still some issues that need to be addressed in this article.

1) *Impact of Participant-Side Non-IID on FedACC:* When applying federated learning in practice, the problem of data heterogeneity between participants may be encountered depending on the specific properties of the scenario. In non-IID scenarios, how the FedACC approach should be improved to ensure that model degrading is still effective in keeping the fairness of model rewards still needs further validation.
2) *Defense Against Active Malicious Participants:* In this article, we analyze FedACC properties for a semi-honest participants scenario. In the scenario of this article, the participant will follow the federated learning task protocol during training, including what parameters to train with and honestly accepting and passing back the model. In the malicious participant scenario, the participant may not follow the protocol exactly and try to disrupt the training process actively. Further extensions to FedACC are needed for the active malicious scenario of contributed participants and limited contributed participants.
3) *Combined With Data Privacy Protection Mechanisms:* In complex threat environments, the expanded attack surface may make existing federated learning designs insufficient to protect participants' privacy. Other defense against different attack surfaces can be achieved by combining DP or homomorphic encryption. In this case, it is still worthwhile to investigate how to implement FedACC while being compatible with existing data privacy protection mechanisms.

REFERENCES

[1] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, pp. 350–361, May 2017.
[2] T. J. Saleem and M. A. Chishti, "Deep learning for the Internet of Things: Potential benefits and use-cases," *Digit. Commun. Netw.*, vol. 7, no. 4, pp. 526–542, 2021.
[3] T. R. Gadekallu, Q.-V. Pham, T. Huynh-The, S. Bhattacharya, P. K. R. Maddikunta, and M. Liyanage, "Federated learning for big data: A survey on opportunities, applications, and future directions," 2021, *arXiv:2110.04160*.
[4] H. 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. Stat.*, 2017, pp. 1273–1282.
[5] T. Zhang, L. Gao, C. He, M. Zhang, B. Krishnamachari, and A. S. Avestimehr, "Federated learning for the Internet of Things: Applications, challenges, and opportunities," *IEEE Internet Things Mag.*, vol. 5, no. 1, pp. 24–29, Mar. 2022.
[6] J. Zhang, C. Li, A. Robles-Kelly, and M. Kankanhalli, "Hierarchically fair federated learning," 2020, *arXiv:2004.10386*.
[7] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proc. IEEE VTS Asia–Pacific Wireless Commun. Symp. (APWCS)*, 2019, pp. 1–5.
[8] D. Ye, R. Yu, M. Pan, and Z. Han, "Federated learning in vehicular edge computing: A selective model aggregation approach," *IEEE Access*, vol. 8, pp. 23920–23935, 2020.
[9] L. Lyu, X. Xu, Q. Wang, and H. Yu, "Collaborative fairness in federated learning," in *Federated Learning: Privacy and Incentive*. Cham, Switzerland: Springer, 2020, pp. 189–204.
[10] Z. Song, H. Sun, H. H. Yang, X. Wang, Y. Zhang, and T. Q. S. Quek, "Reputation-based federated learning for secure wireless networks," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1212–1226, Jan. 2022.
[11] R. Zeng, S. Zhang, J. Wang, and X. Chu, "FMore: An incentive scheme of multi-dimensional auction for federated learning in MEC," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2020, pp. 278–288.
[12] M. Cong, H. Yu, X. Weng, J. Qu, Y. Liu, and S. M. Yiu, "A VCG-based fair incentive mechanism for federated learning," 2020, *arXiv:2008.06680*.
[13] J. Zhang, Y. Wu, and R. Pan, "Incentive mechanism for horizontal federated learning based on reputation and reverse auction," in *Proc. Web Conf.*, 2021, pp. 947–956.
[14] R. H. L. Sim, Y. Zhang, M. C. Chan, and B. K. H. Low, "Collaborative machine learning with incentive-aware model rewards," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 8927–8936.
[15] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," in *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13. San Rafael, CA, USA: Morgan & Claypool, 2019.
[16] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, 2021.
[17] Y. Shi, H. Yu, and C. Leung, "Towards fairness-aware federated learning," 2021, *arXiv:2111.01872*.
[18] L. Lyu et al., "Towards fair and privacy-preserving federated deep models," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 11, pp. 2524–2541, Nov. 2020.
[19] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.
[20] K. Zhang, X. Song, C. Zhang, and S. Yu, "Challenges and future directions of secure federated learning: A survey," *Front. Comput. Sci.*, vol. 16, no. 5, pp. 1–8, 2022.
[21] J. Ding, E. Tramel, A. K. Sahu, S. Wu, S. Avestimehr, and T. Zhang, "Federated learning challenges and opportunities: An outlook," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2022, pp. 8752–8756.
[22] O. Shahid, S. Pouriyeh, R. M. Parizi, Q. Z. Sheng, G. Srivastava, and L. Zhao, "Communication efficiency in federated learning: Achievements and challenges," 2021, *arXiv:2107.10996*.
[23] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-IID data silos: An experimental study," 2021, *arXiv:2102.02079*.

[24] K. Wei et al., "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.
[25] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–7.
[26] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*.
[27] A. Ghorbani and J. Zou, "Data Shapley: Equitable valuation of data for machine learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2242–2251.
[28] C. Dwork, "Differential privacy," in *Proc. Int. Colloq. Automata Lang. Program.*, 2006, pp. 1–12.
[29] M. Abadi et al., "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.
[30] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.
[31] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "LDP-fed: Federated learning with local differential privacy," in *Proc. 3rd ACM Int. Workshop Edge Syst. Anal. Netw.*, 2020, pp. 61–66.
[32] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9530–9539, Oct. 2020.
[33] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Security Privacy (SP)*, 2017, pp. 3–18.
[34] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, vol. 32. Red Hook, NY, USA: Curran, 2019.
[35] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1376–1385.
[36] F. D. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2009, pp. 19–30.
[37] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.
[38] I. Cinar and M. Koklu, "Classification of rice varieties using artificial intelligence methods," *Int. J. Intell. Syst. Appl. Eng.*, vol. 7, no. 3, pp. 188–194, 2019.
[39] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Rep. TR-2009, 2009.
[40] A. Paszke et al., "Automatic differentiation in PyTorch," in *Proc. 31st Conf. Neural Inf. Process. Syst.*, 2017, pp. 1–4.
[41] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1885–1894.



Xianyao You received the B.Sc. degree from the College of Computer and Data Science, Fuzhou University, Fuzhou, China, in 2021, where he is currently pursuing the master's degree.

His current research interests include privacy and security in federated learning.

Ximeng Liu (Senior Member, IEEE) received the B.Sc. degree in electronic engineering and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2010 and 2015, respectively.

He is currently a Full Professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. He was a Research Fellow with the School of Information System, Singapore Management University, Singapore. He has published more than 250 papers on the topics of cloud security and big data security, including papers in IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and IEEE INTERNET OF THINGS JOURNAL. His research interests include cloud security, applied cryptography, and big data security.

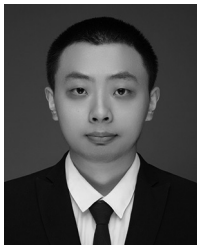
Prof. Liu received the following awards: "Minjiang Scholars" Distinguished Professor, "Qishan Scholars" in Fuzhou University, and ACM SIGSAC China Rising Star Award in 2018.

Xuanwei Lin received the B.E. degree from Minnan Normal University, Zhangzhou, China, in 2019. He is currently pursuing the M.S. degree with the College of Computer and Data Science, Fuzhou University, Fuzhou, China.

His current research interests include machine learning security and spiking neural networks.

Jianping Cai received the master's degree from Fuzhou University, Fuzhou, China, in 2016, where he is currently pursuing the Ph.D. degree with the College of Computer and Data Science.

His research interests include federated learning, Internet of Things technology, machine learning, differential privacy, and optimization theory.



Shaoquan Chen received the B.Sc. degree in mathematics and computer science from Fuzhou University, Fuzhou, China, in 2020, where he is currently pursuing the master's degree.

His current research interests include privacy and machine learning.