Efficient Personalized Federated Learning Method with Adaptive Differential Privacy and Similarity Model Aggregation

Shiqi Mao[1], Fangfang Shan[2]*, Shuaifeng Li[3], Yanlong Lu[4], Xiaojia Wu[5]
School of Computer Science, Zhongyuan University of Technology, Zhengzhou 450007, Henan, China[1, 2, 3, 4, 5]
Henan Key Laboratory of Cyberspace Situation Awareness, Zhengzhou 450001, Henan, China[2]

***Abstract*—In recent years, personalized federated learning (PFL) has garnered significant attention due to its potential for safeguarding data privacy while addressing data heterogeneity across clients. However, existing PFL approaches remain vulnerable to privacy breaches, particularly under adversarial inference and client-side data reconstruction attacks. To address these concerns, we propose DP-FedSim, a novel PFL framework incorporating adaptive differential privacy mechanisms. First, to mitigate the limitations posed by fixed-layer personalization strategies, we evaluate parameter significance using the Fisher information matrix. By selectively retaining parameters with higher Fisher values, DP-FedSim reduces the noise impact, enabling more efficient dynamic personalization. Second, we introduce a layered adaptive gradient clipping method. By leveraging the mean and standard deviation of the gradients within each layer, this method allows DP-FedSim to automatically adjust clipping thresholds in response to real-time privacy demands and model states, enhancing the adaptability to various model structures. This ensures a more accurate balance between privacy preservation and model performance. Furthermore, we present a model similarity-based aggregation method utilizing cosine similarity. This technique dynamically adjusts each client's contribution to the global model update, prioritizing clients with models more similar to the global model. This improves the global model's performance and generalization by allowing DP-FedSim to better handle a variety of data distributions and client model attributes. Experimental results on multiple SVHN cifar-10 datasets show that DP-FedSim outperforms the state-of-the-art PFL algorithm by an average of 5% when data heterogeneity is at its strongest. The efficiency of the suggested modules is validated by ablation tests, and the visualization results shed light on the reasoning behind important hyperparameter settings.**

***Keywords*—Federated learning; differential privacy; gradient clipping; model aggregation**

I. Introduction

A distributed machine learning approach called federated learning (FL) [1] allows several separate devices to work together to train a single model without explicitly sharing local data, protecting privacy and avoiding data leaks. Only model updates are sent to a central server within the FL framework; each participant trains a model separately using their own local data. This decentralized method lowers network communication cost while also protecting data privacy. The performance of traditional federated learning models can be severely harmed by participant data that frequently exhibits non-independent and identically distributed (non-IID) features in real-world applications [2][3][4][5], which can significantly degrade the performance of conventional federated learning models [6][7][8][9]. To address this challenge, Personalized Federated Learning (PFL) algorithms [10][11][12][13] have been developed, incorporating personalization techniques to better accommodate the unique data distributions of individual participants. Despite the notable advances in PFL, privacy concerns remain unresolved. Specifically, model updates exchanged between clients and the central server remain vulnerable to inference and reconstruction attacks that can compromise the privacy of client data [14][15][16][17]. Therefore, integrating robust privacy-preserving techniques, particularly user-level differential privacy [18][19][20][21], is essential to ensure the security and reliability of PFL systems while protecting the privacy of sensitive data.

Despite significant progress, PFL still faces numerous challenges related to the implementation of differential privacy mechanisms. The current PFL techniques [10][22][23] frequently make strong assumptions about parameter partitioning, where clients share a set percentage of the model parameters while the rest are customized. This static approach lacks flexibility in parameter division and fails to fully account for the diversity in client data, which can hinder the performance of personalized models.

Within the context of differential privacy, gradient clipping is a key mechanism used to limit the size of gradients, thereby preventing sensitive information leakage and mitigating the issue of gradient explosion. Gradient clipping also reduces sensitivity, allowing for more efficient privacy budget usage without substantially degrading model performance [25]. However, traditional methods typically employ a fixed clipping threshold *c*, which can result in either over-clipping or under-clipping under different conditions, potentially impairing the performance of the model.

This paper is supported by the National Natural Science Foundation of China (No. 62302540), with author F.F.S. For more information, please visit their website at https://www.nsfc.gov.cn/ . Additionally, it is also funded by the Open Foundation of Henan Key Laboratory of Cyberspace Situation Awareness (No. HNTS2022020), where F.F.S is an author. Further details can be found at http://xt.hnkjt.gov.cn/data/pingtai/ . The research is also supported by the Natural Science Foundation ofHenan Province Youth Science Fund Project (No. 232300420422), and for more information, you can visit https://kjt.henan.gov.cn/2022/09-02/2599082.html.

In most federated learning (FL) frameworks, after a client's local training round is completed, the client sends the model updates, typically in the form of gradients, to a central server responsible for aggregation. The standard aggregation method often relies on a simple averaging of the received gradients. However, in practice, not all clients participate equally in the model training process [26]. Some clients may contribute less due to factors such as limited data, weak computational resources, or unstable network connections. This imbalance can result in the global model becoming overly dependent on clients with larger data volumes or stronger computational capabilities [27], which can hinder the model's ability to capture broader data trends, thus compromising the generalization and convergence of the global model.

However, existing studies still have obvious shortcomings in addressing the above problems. On the one hand, although some studies try to protect privacy through differential privacy techniques, they fail to adequately address the problems of inflexible parameter partitioning and overly fixed gradient trimming strategies in personalized federated learning. On the other hand, the improvement of the aggregation method also fails to effectively take into account the heterogeneity of client data and the difference in model quality, resulting in limited global model performance. Therefore, how to achieve more flexible parameter personalization, more accurate gradient tailoring, and more effective aggregation strategies under the premise of privacy protection has become a key problem to be solved in the field of personalized federated learning.

To address these challenges, we draw inspiration from the work "FedFisher: Leveraging Fisher Information for One-Shot Federated Learning" [28], which employs the Fisher information matrix to facilitate dynamic personalization of model parameters. The square of the first-order derivative of the log-likelihood function is calculated in the core mechanism to assess the contribution of parameters to the curvature of the loss function. In essence, this process captures the information content carried by each parameter. Leveraging this principle, we assess the significance of each client's model parameters through the Fisher information matrix before training begins. By retaining parameters that carry the most information, we mitigate the adverse effects of noise addition in differential privacy settings and avoid potential optimization issues stemming from inappropriate global model parameters.

Building on this, we propose an adaptive gradient pruning strategy, which introduces a hierarchical adaptive gradient clipping method. This approach automatically adjusts the clipping threshold according to the current privacy preservation requirements and the real-time state of the model. In contrast to traditional differential privacy methods that use fixed clipping bounds, this adaptive approach offers greater flexibility and can better accommodate diverse network structures and training environments. It also provides a more accurate response to privacy leakage risks. Furthermore, by allowing the use of larger learning rates, adaptive gradient clipping accelerates convergence and enhances model training performance.

Additionally, we introduce a model similarity-based aggregation method, which utilizes cosine similarity to dynamically adjust each client's contribution to the global model based on the similarity of its parameters to those of the global model. This technique is designed to better align with the data distributions and model quality of different clients, rather than simply assigning equal weight to all updates. By prioritizing updates from clients with models more similar to the global model, this method improves the overall performance and generalization of the global model, as these more similar models are likely to better capture the broader patterns and trends in the data. The following are this paper's main contributions:

- We introduce DP-FedSim, a personalized federated learning framework with adaptive differential privacy. DP-FedSim effectively integrates personalized learning with adaptive gradient tailoring, making it ideal for situations involving a high degree of client data variety and diversity.
- To address the limitations of fixed-layer approaches in traditional personalized federated learning, we leverage the Fisher information matrix to enable a dynamic approach to customization. The Fisher information matrix quantifies the importance of parameters, and under the same additive noise, parameters with higher Fisher values are more sensitive to noise, resulting in greater performance degradation. In order to lessen the effect of noise and improve model performance, we maintain parameters with higher Fisher values.
- We propose a novel adaptive gradient clipping method based on the mean and standard deviation of gradients within each layer. This method accelerates the training process and improves model performance while ensuring privacy protection. In addition, we introduce a model similarity-based aggregation strategy that effectively combines model updates from diverse clients. This method addresses the challenge of heterogeneous client data, where updates may differ significantly, by dynamically adjusting the contributions of client models based on their similarity to the global model.

This paper's remaining sections are organized as follows: We evaluate relevant work in Section II. Section III outlines the foundational concepts relevant to this study. Section IV describes the proposed methods in detail, including the implementation of the three key approaches. Section V presents a performance comparison of DP-FedSim with state-of-the-art methods, and Section VI concludes the paper.

II. Related Work

A. Personalized Federal Learning

A machine learning paradigm called Personalized Federated Learning (PFL) allows a central server to plan model training for dispersed clients without having direct access to their data. The primary objective of PFL is to address data heterogeneity by learning customized models for each client. Mainstream PFL approaches include LG-FedAvg [23], FedBABU [30], PPSGD [29], and FedBN [31]. Both FedBABU [30] and LG-FedAvg [23] adopt fixed local parameters to exploit local data performance for personalization. However, their static partitioning approaches limit the flexibility required for handling diverse data distributions. The FedPer algorithm [10] introduces personalization layers, which are appended to the base model. During training, the base model's parameters are globally aggregated, while the parameters of the personalization layers remain local and are not aggregated. Similarly, FedBN [31] incorporates one or more batch normalization (BN) layers, which remain fixed during training and do not participate in global aggregation. By keeping certain layers locally, these methods improve customization while better accommodating local data peculiarities, however they might not be as flexible in terms of model adaption.

Additionally, algorithms such as GPFL [32], pFedMe [24], and FedAMP [33] aim to learn both global and personalized feature representations. These methods strike a balance between global model consistency and local personalization, allowing models to capture common features across clients while preserving unique, client-specific information. Despite their effectiveness, these approaches are still constrained by the inflexibility of their personalization mechanisms and may see a decrease in performance as a result of using noisy global parameters directly under the differential privacy (DP) technique.

B. Differential Privacy

Differential privacy (DP) is a crucial technique for protecting data privacy, designed to minimize the risk of identifying individual records when statistical information is shared. In the context of federated learning, user-level differential privacy has been widely adopted [18][19][20][34]. This technique quantifies privacy preservation through two key parameters, (ε, δ), where smaller values of ε and δgenerally imply stronger privacy guarantees but add extra noise, which might impair model performance, to the federated learning process. In federated learning, user-level DP is usually accomplished in two steps: first, local updates are clipped and noise is added before being sent to the server. Clipping reduces the effect of local updates, further improving privacy, and the noise is adjusted based on the sensitivity of the function being assessed. Although these steps greatly improve privacy, the required noise addition and gradient clipping may cause performance issues and delayed convergence.

Recent studies have explored ways to mitigate the negative impact of noise and clipping on performance. Sparsification and uniform regularization approaches, for example, are used by LUS and BLUR [19] to mitigate the impacts of noise and accelerate model convergence. The DP-FedSAM [20] algorithm enhances robustness to noise by employing the Sharpness-Aware Minimization (SAM) optimizer, which identifies more reliable places of convergence. Furthermore, PPSGD [29] uses customisation to enhance performance without compromising privacy. In spite of the progress in these methods, research on gradient pruning and clipping within personalized federated learning remains limited. In this paper, we aim to optimize personalized federated learning algorithms through adaptive gradient tailoring, contributing to the ongoing development of personalized federated learning approaches.

C. Federated Learning Aggregation Algorithm

Model aggregation is a core component of federated learning, where the local models from clients are aggregated in each communication round to generate an updated global model. There are two main types of aggregation: parameter-based aggregation and output-based aggregation, with the distinction based on the aggregation target. One of the first and most popular federated learning algorithms is FedAvg [1], which aggregates models by average parameters from all clients, weighted by the size of each client's dataset. The FedProx algorithm [35] modifies the aggregation process by introducing a proximal term in the objective function to control the impact of local models and ensure convergence. Meanwhile, FedNova [36] improves upon FedAvg by normalizing and scaling local updates based on each client’s local iteration count, which helps achieve fairer aggregation. A data agnostic distributional fusion model, which depicts the client's heterogeneous data distribution as a global collection comprised of multiple virtual fusion components with varying parameters and weights, is used by the FedFusion algorithm [37] to characterize the global data distribution.

Although these methods contribute to the development of model aggregation in federated learning, challenges remain. For instance, the FedNova algorithm introduces additional computational complexity due to the need to track and adjust local iteration counts, which increases the computational burden on both the clients and the server, ultimately affecting model convergence speed.

III. Preliminary

A. Personalized Federal Learning

In personalized federated learning (PFL), the primary objective is to train a model that can adapt to the unique data distribution of each client while preserving a degree of global consistency across all clients. This is typically achieved by decomposing the model parameters into two distinct components: one set of globally shared parameters and another set of client-specific personalized parameters, as illustrated in Fig. 1.

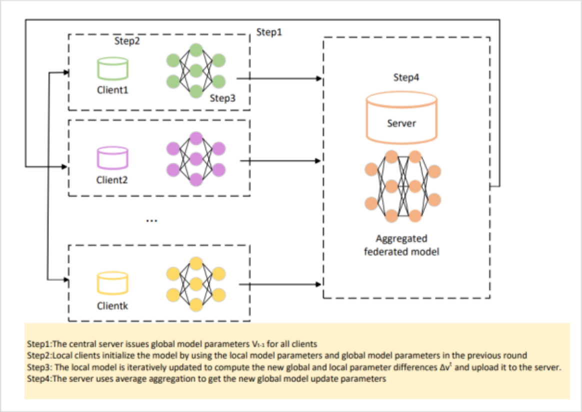

Step1
Step2
Client1
Step3
Client2
...
Clientk
Step4
Server
Aggregated federated model
Step1:The central server issues global model parameters V_{t-1} for all clients
Step2:Local clients initialize the model by using the local model parameters and global model parameters in the previous round
Step3: The local model is iteratively updated to compute the new global and local parameter differences Δv^t and upload it to the server.
Step4:The server uses average aggregation to get the new global model update parameters

Fig. 1. The key processes of traditional personalized federated learning are depicted. First, a central server publishes global model parameters for use by clients. The model is then repeatedly updated by the client using local data to calculate parameter variances, which are then uploaded to the server. Finally, the server creates global model updates by integrating the variances using an average aggregation approach.

Suppose we have k clients, each possessing a unique dataset, denoted as $D_k = \{(x_i, y_i)\}_{i=1}^{N_k}$, where N_k is the size of the dataset for client k, and i indexes the data samples. In common personalization methods, the model parameter vector w_i is typically decomposed into two components: a local part and a global part, represented as w = (u, v). The objective of personalized federated learning is to update the model parameters according to a specific optimization process, as outlined in Eq. (1).

$$\min_{\nu, u_{1:\mathbf{k}}} \left\{ f(\nu, u_{1:\mathbf{k}}) := \frac{1}{m} \sum_{\mathrm{i}}^{\mathrm{k}} f_i(\nu, u_i) \right\} \tag{1}$$

Let $u_{1:k}$ denote (u_1, . . . ,u_k), and let f_i(v, u_i) represent the average loss of parameters v and u_i over the entire dataset D_i for client i, where i = 1, . . . , N_k. The personalized differential privacy mechanism involves two iterative steps:

Local Iteration: During the local iteration, each client i receives the global model parameters v^{t-1} from the server while retaining the local parameters $u_{1\,i}^{t-}$ from the previous round. The model is initialized as $w_{1\,i}^{t-}$= (v^{t-1},$u_{1\,i}^{t-}$). The client then performs local updates, iteratively optimizing the parameters to obtain the updated model w_i^t = (v_i^t,u_i^t) . The global update △v^t is computed as the difference between v_i^t and v^{t-1}.

Global Update: In the global update phase, all clients transmit their local updates △v^t to the server. The server then aggregates these updates by averaging them across all clients, updating the global parameter as follows:

$$\mathrm{v}^t \leftarrow \mathrm{v}^{t-1} + \frac{1}{k} \sum_{i=1}^{m^t} \Delta \mathrm{v}_i^t \tag{2}$$

The new global parameter v^t is then sent back to all clients to initiate the next round of updates.

B. User-Level Differential Privacy

In personalized federated learning, differential privacy offers a robust protection mechanism that effectively addresses the issue of privacy leakage. As a comprehensive privacy protection framework, differential privacy aims to facilitate the analysis of overall dataset properties while safeguarding individual information. This is achieved at the cost of a certain degree of data accuracy, thereby ensuring stringent privacy protection for user data. The ultimate goal is to prevent adversaries from determining whether a specific individual is represented in the dataset.The concept of differential privacy [38] is defined mathematically to delineate this probability gap, as articulated in Definition 1:

Definition 1 (ε, δ). Differential Privacy A randomization mechanism M satisfies (ε, δ)-Differential Privacy (ε > 0, δ > 0) if and only if, for any adjacent input datasets S and S', and for any possible set of output values R, the following holds:

$$Pr[M(S) \in R]\text{„}\ e^{\varepsilon} \cdot Pr[M(S') \in R] + \delta \tag{3}$$

Here, δ denotes the probability of a failure in privacy protection. A randomized algorithm M satisfies (ε, δ)-DP if, for any pair of neighboring datasets D and D' differing by a single record, and for any output subset S in the range of M.

Definition 2 L2 Sensitivity Given a function M and two neighboring datasets D and D', the L2 sensitivity is defined as follows:

$$\Delta f = \max \square M(D) - M(D') \square_2 \tag{4}$$

User-level differential privacy is a specific classification within the broader framework of differential privacy. Noise must be added to the model updates that are locally calculated by each user in order to apply user-level differential privacy to customized federated learning. This approach ensures compliance with user-level differential privacy requirements. Specifically, users must incorporate noise into the gradient or model parameter updates derived from their local datasets after training.

Based on this theoretical foundation, user-level differential privacy is effectively achieved through gradient cropping and noise addition. Gradient cropping is primarily employed to regulate the model's sensitivity to individual data points. By mitigating the influence of outlier samples during a given training round, it helps protect data privacy.

However, much of the existing research focuses on fixed gradient cropping methods. If the cropping threshold is set too high, most gradients may fail to exceed this threshold, rendering the cropping process ineffective. Conversely, setting the threshold too low may excessively constrain gradient updates, hindering the model ’ s ability to glean valuable information from the data. This can diminish the training efficiency and, ultimately, the predictive performance of the model.

Therefore, this paper investigates adaptive gradient cropping within the context of personalized federated learning. A detailed exploration of this topic is presented in Section IV.

IV. Methodology

In this study, we offer a differential privacy federated learning system that combines model similarity-based aggregation, adaptive gradient cropping, and customization based on Fisher information matrices. The proposed approach consists of three key components: Fisher personalized federated learning, adaptive gradient cropping for differential privacy, and aggregation based on model similarity.

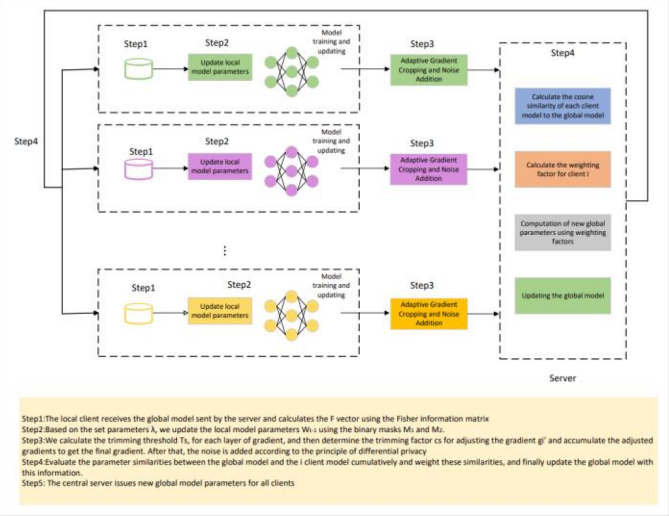
When the client gets the global model from the server, the procedure starts. Each local client then computes the Fisher information vector F_i using the Fisher information matrix. Subsequently, the client generates computational binary masks M_{1i} and $M_{2\mathrm{i}}$ based on the F_i vector and a set of parameters λ. These parameters are crucial in determining which model parameters should be deemed informative and retained throughout the personalization process.

Once the binary masks M_{1i} and $M_{2\mathrm{i}}$ are established, they are utilized to update the local model parameters $w_{1\,i}^{t-}$. The updated model parameters w_i^t are then obtained through further training using these masks.

Next, the cropping thresholds T_s are computed by leveraging the mean and standard deviation of the gradients corresponding to each layer. The gradient parameter $\| g_i \|$ is subsequently calculated, followed by the computation of the cropping factors c_s based on the defined cropping thresholds T_s. These cropping factors are employed to control the scaling of the gradient, yielding the adjusted gradient $g_i^{'}$. The complete gradient is computed by summing the cropped gradients across all layers l .

Finally, model similarity-based aggregation is achieved by calculating the cosine similarity between the global model parameters and the parameters of the i-th client model. Specifically, we first compute the similarity S_i for each client model, followed by summing and weighting all computed similarities. This cumulative similarity is then utilized to update the global model accordingly.

The detailed algorithmic procedure is illustrated in Algorithm 1 and Fig. 2.


Step1
Step2
Update local model parameters
Model training and updating
Step3
Adaptive Gradient Cropping and Noise Addition
Step4
Calculate the cosine similarity of each client model to the global model
Calculate the weighting factor for client i
Computation of new global parameters using weighting factors
Updating the global model
Server

Step1:The local client receives the global model sent by the server and calculates the F vector using the Fisher information matrix
Step2:Based on the set parameters λ, we update the local model parameters W_{t-1} using the binary masks M_1 and M_2.
Step3:We calculate the trimming threshold Ts, for each layer of gradient, and then determine the trimming factor cs for adjusting the gradient gi' and accumulate the adjusted gradients to get the final gradient. After that, the noise is added according to the principle of differential privacy
Step4:Evaluate the parameter similarities between the global model and the i client model cumulatively and weight these similarities, and finally update the global model with this information.
Step5: The central server issues new global model parameters for all clients

Fig. 2. An overview of DP-FedSim. The client first receives the global model and computes the F vector, then updates the local model parameters. Next, the gradient is adjusted by set thresholds and factors, and noise is added to maintain differential privacy. Finally, the server synthesizes the parameter similarities across clients and updates the global model.

A. Based on Fisher Personalized Federal Learning

Motivation: The motivation for personalization in federated learning is underscored by the use of the Fisher information matrix. The Fisher information matrix serves as an effective tool for quantifying the importance of model parameters; specifically, a larger Fisher value indicates a greater significance of the parameter in the model's predictive performance. This observation leads to the conclusion that parameters with elevated Fisher values contribute to a more substantial degradation of model performance when subjected to the same additive noise.

In conventional personalization approaches, after receiving the global model from the server, clients typically designate specific fixed layers within the network as personalization layers. However, this method is inherently limited, as it fails to account for the differential impact of noise on various parameters. To address this limitation, we propose the introduction of Fisher values as a metric for assessing the importance of model parameter information across each layer.

Fisher Personalization: To alleviate the current inflexibility issues associated with personalized federated learning, we implement a dynamic personalization strategy based on the Fisher information matrix. This approach enhances the model's adaptability to the non-independent and identically distributed (non-IID) data characteristic of individual clients. The procedure is as follows: each client i receives the distributed global model w^{t-1} from the server and subsequently computes the Fisher value F_i based on its local private dataset D_i. The retention of the previous round's local parameters is denoted as $w^{t-}_{1\,i}$ = (v^{t-1}, $u^{t-}_{1\,i}$). In $w^{t-}_{1\,i}$, the diagonal approximation of the true Fisher value for each parameter indexed by j is calculated as:

$$F(w_{ij}) = \left(\frac{\partial \log L(w_i, D_i)}{\partial w_{ij}} \right)^2 \tag{5}$$

This formula illustrates how sensitive the model's predictions are to changes in parameter w_j and serves as a foundational element in enhancing the personalization process.

where $L(w_i, D_i)$ denotes the log-likelihood function of w_i . Then by normalizing each parameter j in layer s layer by layer, the value of fisher's F_s is achieved as

$$\hat{F}_{s,j} = \frac{F_{s,j}}{\sum_j F_{s,j}} \tag{6}$$

After we generate the layer-by-layer Fisher values f_s by the above operation, we generate two binary masks M_1 and M_2 for dynamic selection of parameters. In the event that a parameter's Fisher value is larger than or equal to λ, it is set to 1, otherwise the value is set to 0. For each parameter j in layer s, the mask is defined as follows:

$$M_1[j] = \operatorname{sgn}(\hat{F}_{ij} - \lambda) \quad \text{and} \quad M_2[j] = 1 - M_1[j] \tag{7}$$

To choose the right parameters for customization, we next carry out an elemental multiplication between these masks and parameters. In other words that is, the parameters that had a greater Fisher value locally in the previous round are kept through the masking operation. The remaining parameters are replaced with global parameters.

$$w_i^t = M_1 \square \; w_i^{t-1} + M_2 \square \; w^{t-1} \tag{8}$$

where M_1 and $w^{t-}_{1\,i}$, M_2 and w^{t-1} serve as the Hadamard product, i.e., the point-by-point product between the elements. The global parameter supplied by the receiving server is w^{t-1}, whereas the local personalization parameter kept from the previous round is $w^{t-}_{1\,i}$. Updating with this method effectively retains the more informative parameters as personalized parameters, and the less informative parameters are updated by the corresponding informative global parameters.

B. Adaptive Gradient Tailoring Differential Privacy

Motivation: In the realm of differential privacy, traditional fixed gradient cropping methods exhibit inherent limitations, particularly when applied to diverse datasets and model architectures. These methods often lack the flexibility to adapt to varying circumstances, leading to issues of over-cropping or under-cropping. Such discrepancies can adversely affect both the training efficacy and the final performance of the model.

The primary motivation behind the adaptive gradient cropping (AGC) technique is its capacity to dynamically adjust cropping thresholds in response to the statistical characteristics of the weights at each layer. This dynamic adjustment mechanism enhances the stability of model training and offers the flexibility to accommodate different network architectures and training configurations, thereby allowing for more precise control over the risk of privacy leakage. A further significant advantage of the AGC technique is its facilitation of larger learning rates, which is critical for accelerating model convergence. This capability can substantially enhance the efficiency and performance of model training, resulting in a more expedient training process and improved model outcomes while maintaining robust privacy protection.

Algorithm 1: Heading

```
Initialize Global epochs $E_g$ local epochs $E_l$ participants number in t
he $t^{th}$ epoch $m^t$, private data of the $t^{th}$ client $D_i = (X_i, Y_i)$, global model
parameters w and client local parameters $w_i$, hyper-parameters λ,
learning rate η,
Local Update:
For $i$ = 1,2,…,$m^t$ Server
    Receive $w^{t-1}$ from Serve
    $M_{1i}$ and $M_{2i} \leftarrow (\hat{F}_i, \lambda)$
        For $e = 1, 2, ..., E_l$ do
            $w_i^{t-1} = (v^{t-1}, u_i^{t-1}) \leftarrow M_{1i} \square\ w_i^{t-1} + M_{2i} \square\ w^{t-1}$
        End
    $\Delta w_i^t = \sum_{l=1}^{L}\left(\min\left(1, \frac{T_s}{\square g_s \square + \grave{o}}\right) \cdot g_s\right) + \mathrm{N}\ (0, \sigma_{\text{noise}}^2)$
    $w_i^t \leftarrow w_i^{t-1} - \eta \cdot \nabla_w L(w_i^{t-1}; D)$
End
Server Execute:
For t = 1,2,…, $E_l$ do
    For each client model weight $w_i$ do
        $S_i \leftarrow \frac{< w_i, w_g >}{\| w_i \|\ \| w_g \|}$
        $S_{sum} \leftarrow S_{sum} + S_i$
    End

    For $i = 1, 2, ..., m^t$ do
        $w_i \leftarrow \frac{S_i}{S_{sum}}$
        $\Delta w_i^t \leftarrow \Delta w_i^t + w_i \Delta w_i$

    End
    $w^t \leftarrow w^{t-1} + \Delta w_i^t$
    For $i = 1, 2, ..., m^t$ do
        Send $w^t$ to client $i^{th}$
    End
End
```

In the context of Federated Learning (FL), the heterogeneity of data across different clients often results in substantial discrepancies in the model updates submitted by each client. Such variations may lead to excessively large or small gradient updates for some clients, consequently impacting the training stability of the global model and increasing the risk of privacy breaches. This paper presents the AGC approach, which dynamically modifies the cropping threshold according on the statistical characteristics of the gradient at each layer in order to address these issues. This method effectively addresses the complications arising from heterogeneous data, enhancing both training stability and privacy assurance.

1) Calculation of gradient trimming threshold: The fundamental principle of adaptive gradient trimming lies in performing gradient trimming on each layer's parameters of the global model while dynamically adjusting the trimming threshold to accommodate gradient variations across different data distributions. Specifically, for each layer parameter θ_i of the model, we first compute the mean μ_s and standard deviation σ_s of the corresponding gradient g_s in that layer. These statistical measures provide insight into the concentration and distribution characteristics of the gradients within that layer. The cropping threshold T_s is then calculated as follows:

$$T_s = b \times \left(1 + \frac{\sigma_s}{|\ \mu_s| + \grave{o}}\right) \tag{9}$$

where b represents a predetermined base cropping threshold, which governs the overall intensity of cropping, and ϵ is a small positive constant introduced to prevent division by zero errors. In this formulation, a larger standard deviation of the gradient for a given layer suggests a more dispersed gradient distribution, indicating the potential presence of outliers or noise. In such cases, the cropping threshold will be correspondingly elevated to mitigate excessive cropping. Conversely, when the standard deviation is small, the cropping threshold will be relatively low, thereby enforcing a stricter control over the size of the gradient.

2) Gradient trimming process: Following the determination of the cropping threshold for each layer, we proceed to trim the gradient g_s for each respective layer. Specifically, we first calculate the norm $\| g_i \|$ of the gradient, after which the cropping factor c_s is computed based on the previously established cropping threshold T_s.

$$c_s = \min\left(1, \frac{T_s}{\square g_s \square + \partial}\right) \tag{10}$$

The cropping factor cs governs the scaling of the gradient, ensuring that the cropped gradient does not exceed the predefined threshold. Ultimately, the cropped gradient $g_i^{'}$ for this layer can be expressed as follows:

$$g_s^{'} = g_s \times c_s \tag{11}$$

The complete client-side gradient is expressed as the aggregation of the cropped gradients from all layers. Assuming there are L layers, each with a corresponding cropped gradient $g_i^{'}$, the complete gradient for the client can be formulated as follows:

$$G_{\text{i}} = \sum_{l=1}^{L} g_s^{'} \tag{12}$$

where G' represents the complete client-side gradient, and $g_l^{'}$ denotes the cropped gradient for layer l.

This method preserves much of the information found in normal gradients while successfully reducing the negative impacts of anomalous gradients on the global model. As a result, it improves model training stability and reduces information loss.

Gradient update and noise addition after cropping: To further enhance privacy protection, noise is introduced to the cropped gradient $g_i^{'}$. This addition adheres to the principles of differential privacy, with the standard deviation σnoise calculated based on the base cropping threshold b and the noise multiplier noise_multiplier:

$$\sigma_{\text{noise}} = \sqrt{\frac{\text{b}^2 \times noise_{\text{m}}ultiplier^2}{N}} \tag{13}$$

where N denotes the number of clients participating in federated learning. The noise addition process involves incorporating Gaussian noise into the gradient update at each layer, represented by the following formula:

$$g_i^{\text{update}} = G_i + \text{N}\ (0, \sigma_{\text{noise}}^2) \tag{14}$$

This process ensures that the model's privacy is further reinforced while maintaining the integrity of the gradient through cropping. By appropriately calibrating the noise intensity, we can maximize the privacy of user data without compromising the accuracy of the model.

C. Aggregation Based on Model Similarity

Motivation: In the realm of federated learning, the selection of an appropriate aggregation strategy is pivotal to the ultimate performance of the model. The classical Federated Averaging (FedAvg) algorithm simply averages the model weights of all participating clients, with the averaging weighted by the volume of data each client holds. However, in personalized federated learning scenarios, the model updates from clients may exhibit substantial variability due to the inherent heterogeneity of their respective datasets. Consequently, straightforward weighted averaging may result in diminished global model performance or inadequate personalization.

To alleviate this problem, we propose a model similarity-based aggregation method, the core of which is to dynamically adjust the client's contribution weight in federated learning by measuring the consistency of update directions between the client's local model and the global model. The method adopts cosine similarity as the similarity metric: firstly, the parameter update vectors of the client model are cosine similar to the global update direction. The advantage of cosine similarity is that it focuses on the vector direction rather than the magnitude, which can effectively capture the synergy of the model updates, e.g., clients with high similarity in the update direction are consistent with the global trend, which can be given a higher aggregation weight to inhibit the bias caused by non independent identically distributed data leads to biased updates. Compared with Euclidean distance or Pearson correlation coefficient, cosine similarity is more robust to amplitude changes in high-dimensional sparse model parameter space, and the computational efficiency is more suitable for distributed scenarios. Through this mechanism, local model updates compatible with the global objective can be filtered out to improve the convergence speed, while retaining the client's personalized features, ultimately achieving a balanced optimization of global model performance and personalization.

3) Calculation of model similarity: In model similarity-based aggregation methods, it is essential to first quantify the similarity between each client model and the global model. In this paper, we employ cosine similarity as a measure of the degree of similarity between the client model and the global model. Cosine similarity is a widely utilized metric that computes and normalizes the inner product of two vectors, thereby deriving the angular similarity between them. In this approach, we treat the parameter vectors of the global model and each client model as high-dimensional vectors. We then compute the cosine similarity between these vectors layer by layer, ultimately taking the average similarity across all layers.

Specifically, let w_g denote the parameters of the global model and w_i represent the parameters of the i-th client model. The similarity between these two can be defined as

$$S(\mathbf{w}_i, \mathbf{w}_g) = \frac{< \mathbf{w}_i, \mathbf{w}_g >}{\| \mathbf{w}_i \| \| \mathbf{w}_g \|} \tag{15}$$

where <w_i, w_g> denotes the inner product of the parameter vectors w_i and w_g, while $\|w_i\|$ and $\|w_g\|$ represent their Euclidean norms, respectively. This similarity metric assesses the directional consistency of the model updates from clients relative to the global model. In our implementation, we establish a similarity metric mechanism by calculating the cosine similarity between the parameters of the client model and the global model layer by layer.

4) Aggregation methods for similarity weighting: In the traditional Federated Averaging (FedAvg) approach, the contribution of each client to the global model update is typically determined by the proportion of its data volume. Nevertheless, the degree of similarity between the client models and the global model over several training rounds is not taken into consideration by this strategy. To address this limitation, we propose an adjustment to the aggregation process by incorporating the cosine similarity between the client models and the global model as a weighting factor.

This process ensures that the model's privacy is further reinforced while maintaining the integrity of the gradient through cropping. By appropriately calibrating the noise intensity, we can maximize the privacy of user data without compromising the accuracy of the model.

During each training round, we first calculate the cosine similarity between each client model and the global model. To ensure the resulting weights are reasonable, we normalize these similarity values so that the sum of the weights of all clients equals 1. This normalization process effectively adjusts the contributions of the clients, allowing updates from clients that exhibit higher similarity to the global model to carry greater weight in the aggregation process, while minimizing the impact of clients with lower similarity on the global model updates.

Let S_i denote the similarity of the i-th client model. The corresponding weighting factor w_i is then computed using the following equation:

$$w_i = \frac{S_i}{\sum_{j=1}^{N} S_j} \tag{16}$$

where N represents the total number of clients participating in the training, and S_j is the similarity of the j-th client model. This weighting method facilitates a dynamic weighted aggregation strategy based on similarity, enhancing the efficiency and effectiveness of the model training process.

5) Polymerization update: Building upon the weights calculated from model similarity, this paper employs a weighted aggregation strategy to update the global model parameters. Each client model's updated value is weighted and superimposed according to its corresponding weights, resulting in the final updated value of the global model. The aggregation process is outlined as follows:

First, for each client model's update result, the update is multiplied by its corresponding weighting factor. Subsequently, the weighted update values of all clients are accumulated layer by layer. Specifically, let the update value of the i-th client be denoted as Δw_i. The update value of the global model, Δw_g, is then computed using the following formula:

$$\Delta w_g = \sum_{i=1}^{N} w_i \cdot \Delta w_i \tag{17}$$

Where w_i is the weight of the i-th client, and Δw_i represents its corresponding model update value. The aggregated update value Δw_g is subsequently applied to the global model to complete each round of model updates.

By utilizing this similarity-weighted aggregation strategy, the global model not only synthesizes data features from diverse clients but also dynamically adjusts the influence of each client based on model similarity. In the presence of diverse data distributions, this method improves the model's generalization performance.

V. Experiments

A. Experimental Setup

1) Dataset and models: We assessed DP-FedSim's performance against cutting-edge algorithms in a federated learning environment on a variety of image recognition tasks. Fashion-MNIST [39], SVHN [40], and CIFAR-10 [41] were among the datasets used in this assessment.A test set of 10,000 samples and a training set of 60,000 samples make up the Fashion-MNIST dataset. A 28 × 28 grayscale picture linked to a label from a total of 10 classes represents each sample. The 60,000 color, 32 × 32 pixel pictures that make up the CIFAR-10 dataset are divided into 10 different classes, with 6,000 images in each class. The digit classification-focused SVHN dataset, which consists of 26,032 test samples and 73,257 training samples, is taken from Street View photos. Every sample is a 32 x 32 color picture that shows the numbers 0 through 9.

For the model architectures, FEMNIST employs a simple convolutional neural network (CNN) comprising 2 convolutional layers and 2 fully connected layers. CIFAR-10, on the other hand, has a more intricate architecture that consists of three convolutional layers and three fully linked layers. The SVHN dataset is processed using a straightforward model featuring 2 convolutional layers, 1 pooling layer, and 2 fully connected layers.

2) Benchmarks: We evaluate DP-FedSim's performance against a number of cutting-edge federated learning techniques, including FedAvg [1], DP-FedAvg[42], DP-FedSAM [20], and DP-FedSAM-top [20]. The FedAvg algorithm serves as a baseline federated learning method that operates without noise. In contrast, DP-FedAvg guarantees client-level differential privacy (DP) by applying a Gaussian mechanism directly to the local updates. The DP-FedSAM algorithm addresses the adverse effects of differential privacy through the utilization of gradient perturbations, specifically incorporating a Sharpness Aware Minimization (SAM) optimizer to produce locally flat models that exhibit improved stability and robustness against weight perturbations. Additionally, DP-FedSAMtopk is a variant of DP-FedSAM that employs a top-k update thinning technique, further minimizing the magnitude of random noise by updating only the most significant portions of the model updates, thereby enhancing model performance while preserving privacy.

3) Implementation details: For our experiments involving the Fashion-MNIST, SVHN, and CIFAR-10 datasets, we model data heterogeneity across client datasets by partitioning local data from the original dataset using a Dirichlet sampling process. The sampling parameter α controls the degree of imbalance in data distribution among clients; larger values of α correspond to weaker data heterogeneity, while smaller values imply stronger heterogeneity. Our primary evaluation metric is global accuracy. In comparisons with other algorithms, we assess accuracy under varying degrees of non-independent and identically distributed (non-IID) data partitioning, proving that our customized federated learning approach is successful in handling non-IID data.

For the Fashion-MNIST, SVHN, and CIFAR-10 datasets, we set the learning rate to 1×10^{-3} and λ to 0.4. The parameters for differential privacy are set to $\varepsilon = 2$ and $\delta = 1\times10^{-5}$. We establish the number of global rounds at 100, local update rounds at 4, and batch size at 64, with the number of clients set to 100 and a sampling rate of 0.1.

The following sections of this paper are organized into three main parts: first, we conduct a comparative analysis of our algorithm against existing differential privacy federated learning methods. Second, we perform two ablation studies focusing on adaptive gradient cropping and model similarity-based aggregation to validate their effectiveness. Finally, we present a hyperparameter analysis to further elucidate the model's performance.

B. Performance Evaluation

1) Comparative analysis: In Table I, we evaluate the global accuracy of four baseline algorithms across three datasets: Fashion-MNIST, SVHN, and CIFAR-10. To assess the impact of data heterogeneity on the performance of these algorithms, we compare all baselines while varying α within the range of {0.05, 0.1, 0.2}. The results summarized in Table 1 indicate that our proposed algorithm demonstrates superior accuracy and generalization ability under standard noise conditions. This finding underscores the enhanced performance of personalized federated learning with differential privacy.

TABLE I. OPTIMAL TEST ACCURACY OF DP-FEDSIM AND CENTRALIZED BASELINES AT DIFFERENT NON-IID SETTINGS

Data	α	FedAvg	DP-FedAvg	DP-FedSAM	DP-FedSAM-top-k	DP-FedSim
Fminst	0.05	85.12	54.48	58.37	58.64	69.36
	0.1	93.25	62.10	65.95	66.51	71.68
	0.2	93.41	64.99	71.60	72.13	72.93
SVHN	0.05	85.47	77.12	50.21	53.01	84.27
	0.1	86.27	85.28	65.89	66.78	85.83
	0.2	88.64	86.16	72.69	74.53	86.79
Cifar10	0.05	54.62	51.69	45.28	45.78	60.63
	0.1	58.86	55.76	56.78	57.02	63.01
	0.2	64.98	61.44	58.41	59.77	64.07

For instance, in the non-independent identically distributed (non-IID) setting with α= 0.2, the accuracy achieved by our algorithm on the FEMNIST dataset is 78.93%, 86.79% on the SVHN dataset, and 64.07% on CIFAR-10. It is evident that the optimal accuracies of DP-FedSim consistently surpass those of the other baseline algorithms in most cases, highlighting the effectiveness of our approach in improving computational accuracy.

Furthermore, Table I illustrates the robustness and generalization capabilities of our algorithms under varying levels of non-IID distribution, specifically with α set at 0.05, 0.1, and 0.2. The heterogeneous distribution settings among local clients complicate the training and convergence of the global model. Notably, among the four baseline algorithms, the adverse effects of heterogeneous distribution become more pronounced as α decreases.

On the SVHN dataset, our proposed algorithm (Algorithm 1) exhibits superior convergence and generalization compared to DP-FedAvg as the non-IID level diminishes. When α = 0.05, the accuracy of Algorithm 1 exceeds that of DP-FedAvg by 7.15%, indicating a greater adaptability of Algorithm 1 in handling non-independent homogeneous distributions. Additionally, on the CIFAR-10 dataset, Algorithm 1 demonstrates differences in non-independent homogeneous distribution of 2.38% and 1.06% at varying levels of α, which are notably lower than the 4.07% and 5.68% observed for DP-FedAvg, and significantly less than the 11.24% and 2.75% exhibited by DP-FedSAMtopk. These results confirm that, despite the challenges posed by heterogeneous data, Algorithm 1 remains resilient and exhibits enhanced robustness and stability.

C. Ablation Experiment

We carried out a number of ablation tests to clarify the role that each element of our strategy had in the overall performance. Our proposed method encompasses two key components: adaptive gradient cropping (AGC) and model similarity-based aggregation (MSA). To assess their individual impacts, we explored several variant configurations, including the removal of both adaptive gradient cropping and model similarity-based aggregation, the removal of adaptive gradient cropping while retaining model similarity-based aggregation, and the removal of model similarity-based aggregation while utilizing only adaptive gradient cropping.

- In the first variant, we eliminated both adaptive gradient cropping and model similarity-based aggregation, opting for the commonly employed differential privacy and simple average weighted aggregation methods based on fixed gradient cropping. This configuration serves as the baseline model for our comparative analysis.

- In the second variant, adaptive gradient cropping was removed, and we employed the conventional differential privacy method utilizing fixed gradient cropping for training, while maintaining model similarity-based aggregation for server-side aggregation. This setup allows us to evaluate the effectiveness of the model similarity-based aggregation method.

- The third variant involved the removal of the model similarity-based aggregation component, utilizing a simple average weighted aggregation method to assess the effectiveness of the adaptive gradient cropping technique.

In this context, for the first variant, the differential privacy aspect implemented fixed gradient cropping with a cropping threshold c set to 0.4. All other parameter settings remained consistent with those outlined in Section V. The specific experimental results are presented in Table II.

TABLE II. ABLATION STUDY WITH DIFFERENT PRIVACY BUDGETS

Data	ACG	MSA	ε=2	ε=4	ε=8
Fminst			57.74	60.21	62.40
		✓	58.34	61.14	62.67
	✓		70.21	72.82	74.01
	✓	✓	72.31	73.62	74.78
SVHN			72.19	73.17	74.51
		✓	73.59	75.24	77.51
	✓		82.24	84.80	85.92
	✓	✓	84.88	86.34	87.43
Cifar10			45.77	48.31	49.46
		✓	47.44	49.09	49.74
	✓		61.12	63.84	65.73
	✓	✓	64.62	65.88	67.26

The experimental data reveal that the removal of both modules resulted in a notable decrease in model accuracy, thereby underscoring the importance of each component in the modeling framework. Adaptive Gradient Cropping. The implementation of adaptive gradient cropping significantly enhances accuracy across various privacy budgets. Adaptive gradient cropping significantly improves accuracy across all privacy budgets when used in customized federated learning (as seen in the third row of each dataset in Table 2 as opposed to the baseline model (first row of each dataset). This finding validates the effectiveness of adaptive gradient cropping in bolstering the model's performance.

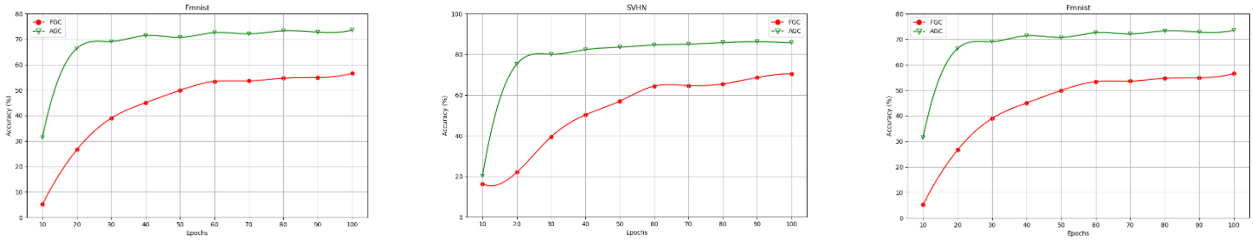


Fig. 3. Fixed gradient cropping and adaptive gradient cropping accuracy plots.

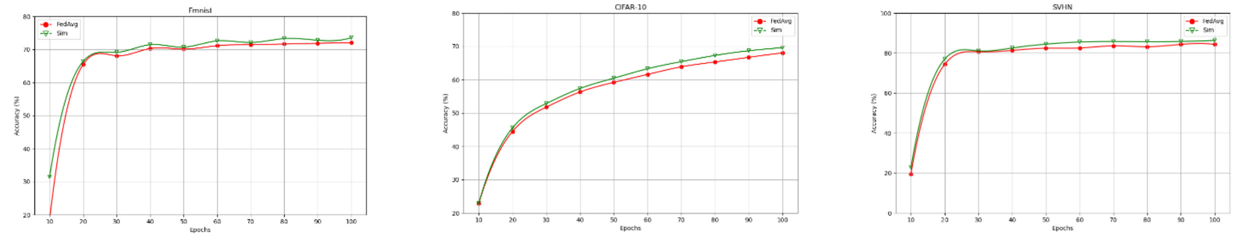


Fig. 4. Weighted average aggregation and model similarity based model accuracy plots.

Model Similarity-Based Aggregation. Similarly, when personalized federated learning relies solely on model similarity-based aggregation (as illustrated in the second row of each dataset in Table 2, there is a marked improvement in accuracy across the privacy budgets relative to the baseline model. This result further corroborates the effectiveness of model similarity-based aggregation within this framework.

Moreover, when both adaptive gradient cropping and model similarity-based aggregation are utilized in tandem (as shown in the fourth row of each dataset in Table 2), the accuracy demonstrates improvement across different privacy budgets compared to the baseline model, the model utilizing adaptive gradient cropping alone, and the model employing model similarity-based aggregation alone. The results clearly indicate that both adaptive gradient cropping and model similarity-based aggregation significantly contribute to the overall performance of the model under varying privacy budgets. The synergistic combination of these two components yields optimal results, thereby enhancing the effectiveness of our proposed algorithm.

D. Hyperparametric Analysis

In experiments concerning personalized federated learning with differential privacy, the selection of hyperparameters significantly influences both the performance and training efficiency of the model. This paper specifically examines the impact of the hyperparameter related to the number of clients on model performance, conducting tests across two distinct datasets.

As illustrated in Figures 5, when the number of clients is set to 5, 10, and 20 for both the Fashion-MNIST and CIFAR-10 datasets, the experimental results indicate a positive correlation between the number of clients and the overall accuracy of the model. The participation of a larger number of clients enables the system to leverage a broader range of local data, thereby enhancing the global model's generalization capability. Furthermore, an increased client count results in a data distribution that more closely reflects real-world scenarios, which helps mitigate the adverse effects of individual client data biases on the model's performance.

However, this increase in client numbers is accompanied by a significant rise in training time. This phenomenon is primarily attributed to the augmented participation of clients in local computations and model aggregation during each training round, leading to increased communication overhead and computational demands. Notably, with the implementation of differential privacy mechanisms, as the number of clients rises and the volume of data per client decreases, the number of communication rounds necessary to achieve a comparable level of convergence may increase, thereby exacerbating the overall training time.

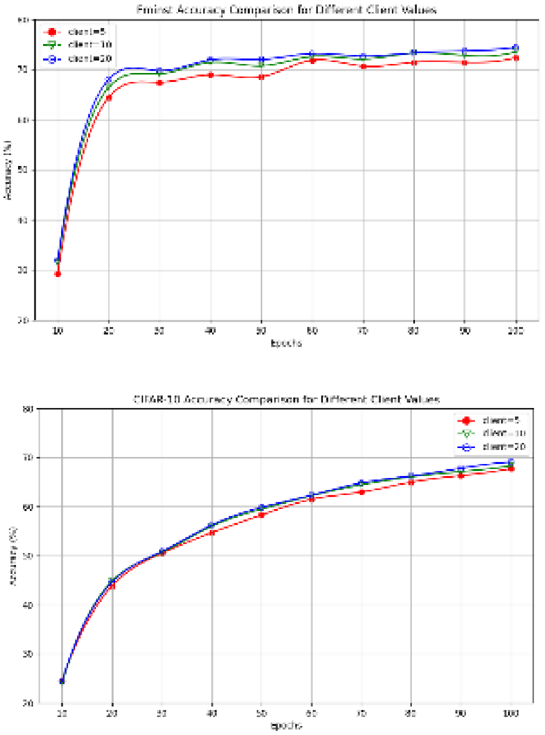


Fig. 5. Accuracy curve for different number of clients.

Consequently, selecting the optimal number of clients necessitates a careful balance between model performance and training efficiency. In scenarios where accuracy is paramount, increasing the number of clients can substantially enhance the model's generalization ability. Conversely, in time-sensitive training contexts, it is imperative to regulate the number of clients to mitigate computation and communication overhead. In practical applications, the number of clients can be adjusted according to specific requirements to achieve an optimal trade-off between performance and efficiency.

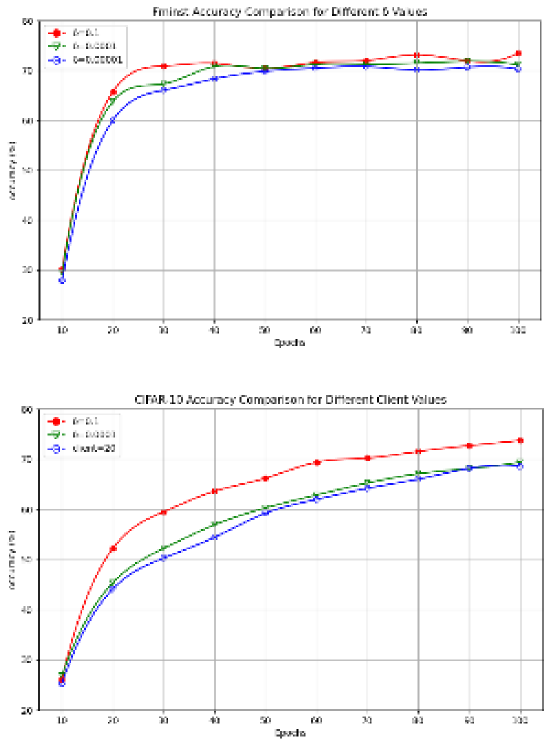


Fig. 6. Accuracy curve for different values of δ.

In Figures 6, we investigate the impact of varying δ-values (0.1, 0.0001, and 0.00001) on the performance of our model using the Fashion-MNIST and CIFAR-10 datasets. The experimental results demonstrate that model accuracy improves as the δ-value increases. This observation can be attributed to the fact that a larger δ-value signifies weaker privacy protection, resulting in reduced noise interference during the training process. Consequently, the model is able to extract useful information from the data more effectively, thereby enhancing its overall accuracy.

However, while a higher δ-value may yield performance benefits, it is crucial to acknowledge that it cannot be set excessively high within the framework of differential privacy. According to the principles of differential privacy, the δ-value represents the probability that the algorithm may violate the privacy budget. A large δ-value consequently diminishes the security of the differential privacy mechanism. If δ is set too high, the efficacy of privacy protection becomes questionable, potentially exposing sensitive data to the risk of leakage.

Therefore, in practical applications, the choice of parameter δ needs to be based on the differential privacy framework, which is a trade-off between model accuracy and privacy protection strength. Differential privacy achieves privacy protection by adding noise or data perturbation, and its privacy budget parameter ε and relaxation parameter δ together determine an upper bound on the risk of privacy leakage. Specifically, δ denotes the probability threshold that the algorithm cannot satisfy strict ε-differential privacy; a smaller value of δ enhances the privacy guarantee but may lead to a decrease in the model's utility; conversely, increasing δ may enhance the model's performance but increase the likelihood of sensitive information exposure. For example, a decrease in the noise scale will reduce the perturbation to the training data distribution, but will weaken the strictness of the privacy boundaries. Therefore, it is recommended to experimentally quantify the effects of different (ε,δ) combinations on the model metrics according to the sensitivity requirements of the application scenarios, and ultimately choose the optimal parameter configurations that can satisfy the privacy authentication criteria while maintaining the model usability.

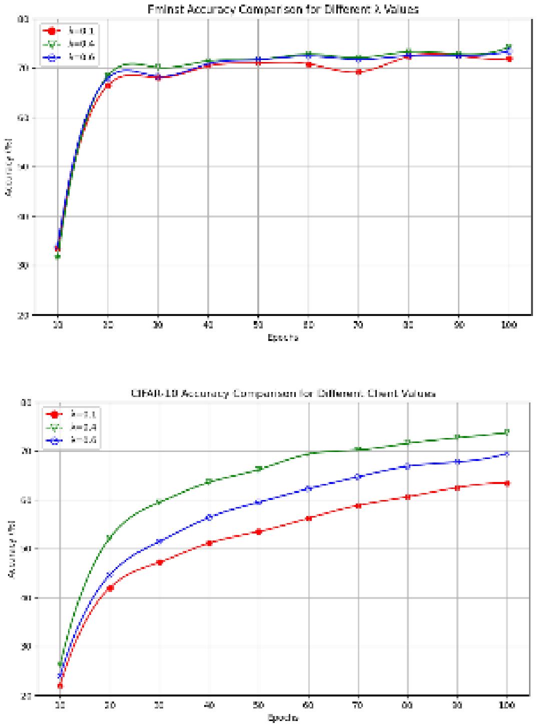


Fig. 7. Accuracy curve for different values of λ.

The value λ is a crucial hyperparameter in customized federated learning, affecting the weight balance between the global and local models in the experiment shown in Figure 7. By adjusting λ, the system can regulate the extent of fusion between the global model and the personalized model during the aggregation process. In our experiments, we set λ to values of 0.1, 0.4, and 0.6 and evaluated the model's performance across different datasets.

The results indicate that setting λ to 0.4 yields the best performance, achieving high accuracy on both the Fashion-MNIST and CIFAR-10 datasets. Specifically, when λ is 0.4, the model strikes an optimal balance between global generalization and local personalization, effectively maintaining a degree of personalization while retaining the shared knowledge encapsulated in the global model. This finding suggests that a moderate fusion of global and local models can enhance the overall performance of personalized federated learning, highlighting the importance of carefully selecting λ to achieve the desired model efficacy.

VI. Conclusion

DP-FedSim is a customized federated learning system with adaptive differential privacy that we present in this paper. This framework effectively addresses the limitations of traditional personalized federated learning, particularly its inflexibility in handling data heterogeneity, while also mitigating the adverse effects of additive noise associated with differential privacy on model performance. DP-FedSim leverages the properties of Fisher's information entropy matrix to accurately quantify the significance of model parameters, allowing for the retention of parameters with larger Fisher values. This strategy reduces the detrimental impact of noise addition on model efficacy. From the perspective of differential privacy, we introduce a hierarchical adaptive gradient cropping method that enables the system to automatically adjust the cropping threshold based on current privacy protection requirements and the real-time state of the model. During the model aggregation phase, the server evaluates the similarity between model parameters by computing metrics such as cosine similarity and dynamically modifies the contribution of each client model to the global model update. This adaptive approach enhances the framework's ability to accommodate the diverse data distributions and model qualities present among different clients. We apply the proposed algorithm to the Fashion-MNIST, SVHN, and CIFAR-10 datasets, demonstrating that our model achieves superior accuracy compared to other differential privacy algorithms, as evidenced by comparative experiments against state-of-the-art models. Furthermore, we conduct ablation experiments to analyze the contribution of each component to the overall performance of the model, while also discussing the rationale behind our hyperparameter settings through detailed hyperparameter analysis.

References

[1] McMahan, B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017, April). Communication-efficient learning of deep networks from decentralized data. In Artificial intelligence and statistics (pp. 1273-1282). PMLR.

[2] Tan, Y., Liu, Y., Long, G., Jiang, J., Lu, Q., & Zhang, C. (2023, June). Federated learning on non-iid graphs via structural knowledge sharing. In Proceedings of the AAAI conference on artificial intelligence (Vol. 37, No. 8, pp. 9953-9961).

[3] Ye, M., Fang, X., Du, B., Yuen, P. C., & Tao, D. (2023). Heterogeneous federated learning: State-of-the-art and research challenges. ACM Computing Surveys, 56(3), 1-44.

[4] Huang, W., Ye, M., Shi, Z., & Du, B. (2023). Generalizable heterogeneous federated cross-correlation and instance similarity learning. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[5] Fang, X., Ye, M., & Yang, X. (2023). Robust heterogeneous federated learning under data corruption. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5020-5030).

[6] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., ... & Zhao, S. (2021). Advances and open problems in federated learning. Foundations and trends® in machine learning, 14(1–2), 1-210.

[7] Li, T., Sahu, A. K., Talwalkar, A., & Smith, V. (2020). Federated learning: Challenges, methods, and future directions. IEEE signal processing magazine, 37(3), 50-60.

[8] Fang, X., & Ye, M. (2022). Robust federated learning with noisy and heterogeneous clients. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10072-10081).

[9] Huang, W., Ye, M., & Du, B. (2022). Learn from others and be yourself in heterogeneous federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10143-10153).

[10] Arivazhagan, M. G., Aggarwal, V., Singh, A. K., & Choudhary, S. (2019). Federated learning with personalization layers. arxiv preprint arxiv:1912.00818.

[11] Liang, P. P., Liu, T., Ziyin, L., Allen, N. B., Auerbach, R. P., Brent, D., ... & Morency, L. P. (2020). Think locally, act globally: Federated learning with local and global representations. arxiv preprint arxiv:2001.01523.

[12] Ma, X., Zhang, J., Guo, S., & Xu, W. (2022). Layer-wised model aggregation for personalized federated learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10092-10101).

[13] Tan, A. Z., Yu, H., Cui, L., & Yang, Q. (2022). Towards personalized federated learning. IEEE transactions on neural networks and learning systems, 34(12), 9587-9603.

[14] Fredrikson, M., Jha, S., & Ristenpart, T. (2015, October). Model inversion attacks that exploit confidence information and basic countermeasures. In Proceedings of the 22nd ACM SIGSAC conference on computer and communications security (pp. 1322-1333).

[15] Melis, L., Song, C., De Cristofaro, E., & Shmatikov, V. (2019, May). Exploiting unintended feature leakage in collaborative learning. In 2019 IEEE symposium on security and privacy (SP) (pp. 691-706). IEEE.

[16] Wen, Y., Geiping, J., Fowl, L., Goldblum, M., & Goldstein, T. (2022). Fishing for user data in large-batch federated learning via gradient magnification. arXiv preprint arXiv:2202.00580.

[17] Huang, Y., Gupta, S., Song, Z., Li, K., & Arora, S. (2021). Evaluating gradient inversion attacks and defenses in federated learning. Advances in neural information processing systems, 34, 7232-7241.

[18] Geyer, R. C., Klein, T., & Nabi, M. (2017). Differentially private federated learning: A client level perspective. arxiv preprint arxiv:1712.07557.

[19] Cheng, A., Wang, P., Zhang, X. S., & Cheng, J. (2022). Differentially private federated learning with local regularization and sparsification. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10122-10131).

[20] Shi, Y., Liu, Y., Wei, K., Shen, L., Wang, X., & Tao, D. (2023). Make landscape flatter in differentially private federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 24552-24562).

[21] Wei, K., Li, J., Ding, M., Ma, C., Su, H., Zhang, B., & Poor, H. V. (2021). User-level privacy-preserving federated learning: Analysis and performance optimization. IEEE Transactions on Mobile Computing, 21(9), 3388-3401.

[22] Ma, X., Zhang, J., Guo, S., & Xu, W. (2022). Layer-wised model aggregation for personalized federated learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10092-10101).

[23] Liang, P. P., Liu, T., Ziyin, L., Allen, N. B., Auerbach, R. P., Brent, D., ... & Morency, L. P. (2020). Think locally, act globally: Federated learning with local and global representations. arxiv preprint arxiv:2001.01523.

[24] T Dinh, C., Tran, N., & Nguyen, J. (2020). Personalized federated learning with moreau envelopes. Advances in neural information processing systems, 33, 21394-21405.

[25] Li, Y., Yang, S., Ren, X., Shi, L., & Zhao, C. (2023). Multi-Stage Asynchronous Federated Learning with Adaptive Differential Privacy. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[26] Qi, P., Chiaro, D., Guzzo, A., Ianni, M., Fortino, G., & Piccialli, F. (2024). Model aggregation techniques in federated learning: A comprehensive survey. Future Generation Computer Systems, 150, 272-293.

[27] Pillutla, K., Kakade, S. M., & Harchaoui, Z. (2022). Robust aggregation for federated learning. IEEE Transactions on Signal Processing, 70, 1142-1154.

[28] Jhunjhunwala, D., Wang, S., & Joshi, G. (2024, April). FedFisher: Leveraging Fisher Information for One-Shot Federated Learning. In International Conference on Artificial Intelligence and Statistics (pp. 1612-1620). PMLR.

[29] Bietti, A., Wei, C. Y., Dudik, M., Langford, J., & Wu, S. (2022, June). Personalization improves privacy-accuracy tradeoffs in federated learning. In International Conference on Machine Learning (pp. 1945-1962). PMLR.

[30] Oh, J., Kim, S., & Yun, S. Y. (2021). Fedbabu: Towards enhanced representation for federated image classification. arxiv preprint arxiv:2106.06042.

[31] Li, X., Jiang, M., Zhang, X., Kamp, M., & Dou, Q. (2021). Fedbn: Federated learning on non-iid features via local batch normalization. arxiv preprint arxiv:2102.07623.

[32] Zhang, J., Hua, Y., Wang, H., Song, T., Xue, Z., Ma, R., ... & Guan, H. (2023). Gpfl: Simultaneously learning global and personalized feature information for personalized federated learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5041-5051).

[33] Huang, Y., Chu, L., Zhou, Z., Wang, L., Liu, J., Pei, J., & Zhang, Y. (2021, May). Personalized cross-silo federated learning on non-iid data. In Proceedings of the AAAI conference on artificial intelligence (Vol. 35, No. 9, pp. 7865-7873).

[34] Li, T., Zaheer, M., Reddi, S., & Smith, V. (2022, June). Private adaptive optimization with side information. In International Conference on Machine Learning (pp. 13086-13105). PMLR.

[35] Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., & Smith, V. (2020). Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems, 2, 429-450.

[36] Wang, J., Liu, Q., Liang, H., Joshi, G., & Poor, H. V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization. Advances in neural information processing systems, 33, 7611-7623.

[37] Duan, J. H., Li, W., Zou, D., Li, R., & Lu, S. (2023). Federated learning with data-agnostic distribution fusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8074-8083).

[38] Dwork, C. (2006, July). Differential privacy. In International colloquium on automata, languages, and programming (pp. 1-12). Berlin, Heidelberg: Springer Berlin Heidelberg.

[39] Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečný, J., McMahan, H. B., & Talwalkar, A. (2018). Leaf: A benchmark for federated settings. arxiv preprint arxiv:1812.01097.

[40] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011, December). Reading digits in natural images with unsupervised feature learning. In NIPS workshop on deep learning and unsupervised feature learning (Vol. 2011, No. 2, p. 4).

[41] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[42] McMahan, H. B., Ramage, D., Talwar, K., & Zhang, L. (2017). Learning differentially private recurrent language models. arxiv preprint arxiv:1710.06963.