Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.079808

ARTICLE

FedGLP-ADP: Federated Learning with Gradient-Based Layer-Wise Personalization and Adaptive Differential Privacy

Di Xiao*, Wenting Jiang and Min Li

College of Computer Science, Chongqing University, Chongqing, China

*Corresponding Author: Di Xiao. Email: dixiao@cqu.edu.cn

Received: 28 January 2026; Accepted: 09 March 2026; Published: 09 April 2026

ABSTRACT: The rapid advancement of the Internet of Things (IoT) has transformed edge devices from simple data collectors into intelligent units capable of local processing and collaborative learning. However, the vast amounts of sensitive data generated by these devices face severe constraints from "data silos" and risks of privacy breaches. Federated learning (FL), as a distributed collaborative paradigm that avoids sharing raw data, holds great promise in the IoT domain. Nevertheless, it remains vulnerable to gradient leakage threats. While traditional differential privacy (DP) techniques mitigate privacy risks, they often come at the cost of significantly reduced model performance—a limitation particularly pronounced in resource-constrained IoT environments characterised by non-independent and identically distributed (non-IID) data distribution. To bridge the gap between privacy preservation and high performance on heterogeneous data, this paper proposes a novel personalized federated learning (PFL) method, FedGLP-ADP. This method leverages historical gradient information to provide a more detailed partitioning of parameters, aiming to prevent personalized knowledge from being affected by noise as much as possible, thereby reducing model degradation. Building on this, we propose an adaptive DP mechanism that optimizes both the clipping and noising steps to minimize the impact of noise on global knowledge. Experimental results show that FedGLP-ADP exhibits superior performance compared to other representative methods under different privacy levels and non-IID degrees.

KEYWORDS: Personalized federated learning; non-IID data; differential privacy; parameter decoupling

1 Introduction

With the rapid advancement of information technology, the Internet of Things (IoT) has become an essential component of digital infrastructure, connecting billions of distributed devices. This evolution marks a significant transformation: edge devices are no longer mere data collectors but intelligent units equipped with local processing and collaborative capabilities. These devices continuously generate vast amounts of data at the network edge, powering various applications in industries, cities, healthcare, maritime sectors, and beyond. However, under traditional centralized machine learning frameworks, data aggregation faces significant practical challenges. The raw data collected by distributed IoT nodes often involves users' sensitive information, making the sharing of such data increasingly difficult due to the dual constraints of strict privacy regulations and intense commercial competition. This has led to the "data silo" problem, which greatly hinders the collaborative development of distributed intelligence. To break these silos, Federated Learning (FL) [1] has emerged. As a novel distributed paradigm, FL allows IoT participants to retain their data locally and only share model parameters or gradients with a central server. However, research has shown that

the model parameters or gradients shared by clients may still leak information about their private training data [2,3].

In recent years, researchers have proposed various methods to enhance privacy protection in FL, including homomorphic encryption (HE) [4], secure multiparty computation (SMC) [5], and differential privacy (DP) [6]. Compared with HE, which has high computational complexity in encryption and decryption, and SMC, which requires complex protocols and frequent communication, DP only needs to perturb the original data by adding a small amount of noise to provide substantial privacy protection, making it more suitable for resource-constrained scenarios such as the IoT. FL based on DP obscures individual client contributions through random noise perturbation, but this often comes at the cost of model performance. In addition, in heterogeneous IoT environments, data across different nodes exhibits non-independent and identically distributed (non-IID) characteristics. In such cases, the significantly divergent gradient values among clients are more sensitive to noise, leading to a sharp decline in both model accuracy and convergence speed.

To address these challenges, this paper proposes a novel personalized federated learning (PFL) method, FedGLP-ADP. We propose a gradient-based layer-wise personalization (GLP) strategy. GLP regards parameters that undergo significant changes before and after local training as personalized ones, keeps them locally without adding noise, thereby effectively handling non-IID data and alleviating convergence difficulties caused by noise. Meanwhile, we adopt an adaptive clipping-and-noising DP (ACNDP) mechanism. ACNDP dynamically allocates layer-wise clipping thresholds based on the variation trend of shared gradients and adaptively adjusts the noise scale according to the degree of personalization, accelerating model convergence while preserving privacy. We summarize our contributions as follows:

- We propose a parameter-granularity personalized strategy, which achieves layer-wise personalized partitioning based on the parameter variation during local training. It quantifies the contribution of clients to the global model and implements weighted aggregation, thereby enhancing robustness to heterogeneous data and noise.
- We design an adaptive DP mechanism that optimizes both the clipping and noising steps, improving model performance under privacy protection.
- We conduct extensive experiments on three datasets, and the results validate the effectiveness of our proposed method.

2 Related Work

2.1 Differential Privacy

FL based on DP obscures individual client contributions through random noise perturbation, but this often comes at the cost of model performance. To minimize the impact of DP on model performance, researchers have proposed various methods. AE-DPFL [7] uses a voting-based mechanism to avoid dimension dependency issues, thereby ensuring secure aggregation and significantly reducing communication costs. Fed-SMP [8] sparsifies model gradients before adding Gaussian noise, mitigating the impact of privacy protection on model accuracy. Beyond local update sparsification (LUS), BLUR+LUS [9] further constrains the norm of local updates via bounded local update regularization (BLUR) to reduce the amount of noise added, thus improving model convergence. FedDPA [10] applies different levels of constraint to different parameters, minimizing the negative effects of clipping. Although effective, these methods apply a uniform clipping threshold, overlooking significant variation in parameter/update magnitudes across layers in FL. This often leads to improper clipping and noising, harming convergence and performance. Moreover, as parameter updates shrink during training, the relative impact of added noise increases [11], making convergence difficult in later stages, especially under a small privacy budget.

2.2 Personalized Federated Learning

In real-world non-IID scenarios, the local objectives of clients deviate from the global objective, leading to an increasing gradient discrepancy. This makes the model more sensitive to noise and ultimately makes it challenging for the global model to achieve usable accuracy. PFL aims to address the issue of data heterogeneity by enabling clients to adapt to their local data distributions [12]. Current PFL methods mainly adopt parameter decoupling to divide the model into two components: a feature extractor (the backbone) and a classifier (the head). FedPer [13], FedBABU [14], FedRep [15], FedPAC [16], FedAS [17], PPFed [18], and FedMPS [19] treat the head as the personalized part and the backbone as the shared part, so that different clients facing heterogeneous data share the same feature extraction process while applying their own classification strategies. In contrast, methods like LG-FedAvg [20], FedClassAvg [21], and FedSSA [22] regard the backbone as the personalized part and the head as the shared part, they share a consistent classification strategy but employ client-specific feature extractors locally. Nevertheless, recent studies [23] have shown that even within the same layer, parameters can differ significantly in their importance for prediction. The above methods rely on preselecting personalized partitions, which lack flexibility in adapting to diverse data characteristics and thus limit the potential of parameter decoupling.

To address these issues, we propose a novel PFL method with adaptive DP. Our method dynamically selects personalized parameters based on the variation of parameter change before and after local training on the client side. This strategy not only addresses the inflexibility of fixed parameter partitioning, but also avoids potential degradation in generalization performance caused by personalizing too many parameters at once by progressively expanding the personalized portion. To prevent personalized parameters from being affected by noise, we only upload the gradients of the shared portion and allocate different clipping thresholds to each layer of the model based on the gradient variation trend. Furthermore, as the training progresses, the number of gradients in the shared portion decreases, and the scale of the added noise is reduced, thereby facilitating model convergence.

3 Preliminaries

3.1 Federated Learning and Personalized Federated Learning

In the training process of the standard FL framework, there exist multiple clients and one central server. When the number of clients is N, the set $C = \{C_1, C_2, \ldots, C_N\}$ denotes all clients, and the set $D = \{D_1, D_2, \ldots, D_N\}$ represents the data distributions of all clients. The total number of communication rounds between the server and clients is T. At the t-th communication round ($t \leq T$), clients perform local training and then send their gradients to the central server S. Then, the server S calculates a global model with the received gradients instead of training samples of each client. The goal of FL is to minimize the losses across all clients in C:

$$\min_{(w_1, w_2, \ldots, w_N)} F(w) := \frac{1}{N} \sum_{i=1}^{N} f_i(w_i) \tag{1}$$

where $w_i \in \mathbb{R}^d$ encodes the parameters of the local model of client C_i, and $f_i(w_i) := \mathbb{E}_{(x,y)\sim D_i}[f_i(w_i; x, y)]$ represents the expected loss over the data distribution D_i of client C_i.

In standard FL, all clients share a global model with parameter w, and $w = w_1 = \cdots = w_N$. Therefore, the learning objective of standard FL can be described as

$$\min_{w \in \mathbb{R}^d} F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{(x,y)\sim D_i}[f_i(w; x, y)] \tag{2}$$

In practical applications, the data distributions of clients in FL may differ, a phenomenon known as data heterogeneity. In such circumstances, merely minimizing the average local loss with no personalization will result in poor performance. In contrast, in PFL, $w \neq w_1 \neq \cdots \neq w_N$, and model parameters are partitioned into a personalized part and a shared part, i.e., $w_i = (u_i, v_i)$. Therefore, the objective is to optimize all the parameters from w_i to w_N. Combining Eqs. (1) and (2), we obtain the following objective function:

$$\min_{(w_1, w_2, \ldots, w_N)} F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{(x,y)\sim D_i}[f_i(w_i; x, y)] \tag{3}$$

PFL can achieve this goal through a two-step iteration:

- **Local Update:** During the local update phase, each client C_i combines the latest received global parameters u^t with the retained local parameters v_i^t from the previous round to obtain the initial model $w_i^t = (u^t, v_i^t)$ for the current round. Then, the client trains on its local data to generate new model parameters $w_i^{t+1} = (u_i^{t+1}, v_i^{t+1})$. Subsequently, the local update Δu_i^{t+1} is calculated as the difference between u_i^{t+1} and u^t.
- **Global Aggregation:** In the aggregation phase, all clients send their local updates Δu_i^{t+1} to the server. The server averages these updates to update the parameters as $u^{t+1} = u^t + \frac{1}{|\mathcal{P}^t|} \sum_{i=1}^{|\mathcal{P}^t|} \Delta u_i^{t+1}$, where $|\mathcal{P}^t|$ is the number of clients participating in the training at round t. These updated parameters are then distributed back to all clients for the next round of local updates.

3.2 User-Level Differential Privacy

DP, as a lightweight privacy-preserving approach, can achieve strong privacy protection by simply adding a small amount of random noise to the data. It is formally defined as follows:

Definition 1 Differential Privacy: A randomized algorithm M satisfies (ϵ, δ)-DP if for any two adjacent datasets D and D' for any subset of outputs $S \subseteq Range(M)$, the following holds:

$$\Pr[M(D) \in S] \leq e^{\epsilon} \cdot \Pr[M(D') \in S] + \delta \tag{4}$$

where ϵ is the privacy budget and δ denotes a small failure probability. These two parameters together regulate the overall level of privacy protection: the smaller ϵ and δ are, the higher the level of privacy protection.

This definition implies that adjacent datasets cannot be distinguished merely by observing the output of M, thereby protecting individual records in the dataset from being identified. The most commonly used method to achieve (ϵ, δ)-DP is the Gaussian mechanism, whose core idea is to add random noise sampled from a Gaussian distribution $N(0, \sigma^2)$ to the output of function f, where σ is the noise intensity. The noise intensity depends not only on the privacy parameters ε and δ, but also on the sensitivity Δf of function f.

Definition 2 L2 Sensitivity: Let f be a function, the L2-sensitivity of f is defined as

$$\Delta_2 f = \max_{D, D' \text{ adjacent}} \|f(D) - f(D')\|_2 \tag{5}$$

In this paper, we adopt user-level DP, which is a stronger privacy requirement suitable for IoT environments. We define the adjacency of datasets at the user granularity.

Definition 3 User-Level Adjacency: Let $D = \{D_1, D_2, \ldots, D_N\}$ be a collection of datasets of all clients. Two collections D and D' are adjacent if D' can be obtained by adding or removing the entire dataset D_i of a single client C_i from D.

DP-FedAvg [24] achieved user-level DP in FL for the first time by applying the Gaussian mechanism, protecting the participation of individual clients from being identified. To guarantee user-level DP, DP-FedAvg constrains the norm of each client's local model update Δw within a threshold C (clipping), and then adds Gaussian noise to the clipped updates (noising) before uploading them to the server. Formally, this process can be expressed as

$$\overline{\Delta w} = \Delta w \cdot \min\left(1, \frac{C}{\|\Delta w\|_2}\right) \quad \widetilde{\Delta w} = \overline{\Delta w} + \mathcal{N}(0, \sigma^2 C^2 / |\mathcal{P}^t|) \tag{6}$$

where C denotes the clipping bound, which controls the maximum contribution of a single client to the global update, and σ is the noise multiplier computed by privacy accountant and composition mechanism.

4 Method

The proposed FL scheme is illustrated in Fig. 1. In general, each client receives the global model and combines its local personalized parameters with the global parameters using the binary mask of the current round to obtain the initialized model. Then, the client trains the model on its local data to generate updated gradients, and segregates the shared gradients. Before uploading the shared gradients to the central server, each client perturbs the shared gradients according to the clipping threshold weights and the binary mask. Subsequently, the binary mask for the next round is computed using the perturbed shared gradients, and the clipping threshold weights are updated based on the gradient variation trend. Finally, the central server receives the perturbed shared gradients and masks from all clients, and aggregates the gradients via weighted aggregation based on their masks, thereby obtaining the updated global model.

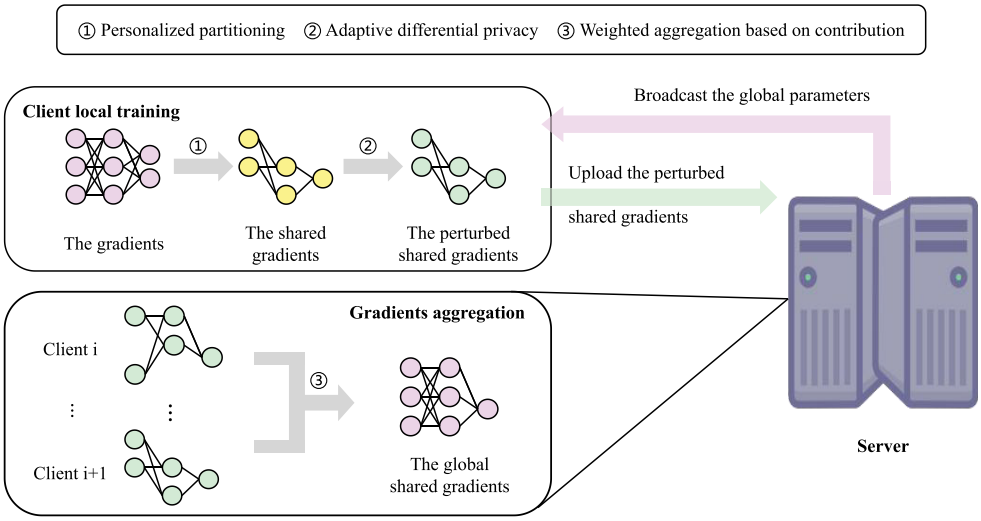


Figure 1: Overall architecture of the FedGLP-ADP algorithm.

4.1 Gradient-Based Layer-Wise Personalization

In previous works on PFL, specific layers of the model are usually preselected as the personalized part to adapt to the local data distribution. However, such an inflexible partitioning limits the potential of parameter decoupling and fails to account for the varying importance of parameters within the same layer. Our intuition is that if a subset of parameters in the client model exhibits greater changes during local training compared to others, it indicates that these parameters are more likely to capture personalized features fitting the unique local data distribution, and thus should be designated as personalized parameters. Conversely, the

remaining parameters, which capture more generic and generalizable features, should be treated as shared parameters for global aggregation. Thus we propose a GLP strategy, achieving personalized partitioning via simple traversal with nearly negligible computation cost. Specifically, after local training, parameters with smallest variations are global knowledge contributors (their gradients uploaded as shared ones to the server for aggregation), while those with largest variations are retained locally to learn personalized knowledge. However, to prevent the mask from revealing the most active features in the local dataset, we postpone the computation of the mask until after the perturbation of the shared gradients. At this stage, the generation of the mask can be interpreted as a utility-based randomized selection process: under the guarantee of DP, parameters with larger gradients are assigned a higher probability of being selected. In the following, we provide a detailed description of this module.

In communication round t, after receiving the global model parameters w_g^t from the central server, client i first uses the binary mask m_i^t calculated in the previous round to obtain the initial model parameters for the current round:

$$u_i^t = w_g^t \odot \neg m_i^t, \quad v_i^t = w_i^t \odot m_i^t \tag{7}$$

where $\odot$ denotes the Hadamard product (element-wise multiplication), and $\neg$ represents the binary negation of the mask. The binary mask m_i^t is defined as

$$m_i^t = \begin{cases} 1, & \text{if personalized} \\ 0, & \text{if shared} \end{cases} \tag{8}$$

In other words, the mask m_i^t takes the value 1 at positions corresponding to personalized parameters and 0 at positions corresponding to shared parameters, and $\neg m_i^t$ reverses these values. Note that at the beginning of FL, the masks of all clients are initialized to 0.

Subsequently, the process enters the local update phase, during which, with reference to [10], the two sets of parameters are updated using two different loss functions, respectively.

$$\mathcal{L}_1 = \text{CrossEntropy}(y, \hat{y}) + \frac{\lambda_1}{2} \| v_i - v_i^t \|_2 \tag{9}$$

$$\mathcal{L}_2 = \text{CrossEntropy}(y, \hat{y}) + \frac{\lambda_2}{2} \left\| \| u_i - u_i^t \|_2 - C \right\|_2 \tag{10}$$

where y is the ground-truth label, $\hat{y}$ is the predicted label, λ_1 and λ_2 are two hyperparameters, and C is the global clipping threshold. $\mathcal{L}_1$ retains personalized knowledge effectively and accelerates convergence, while $\mathcal{L}_2$ ensures shared parameter updates stay near the clipping boundary, mitigating clipping's negative impact.

After the local update, the perturbed gradients are denoted as $\widetilde{\Delta u}_i^{t+1}$. Then, according to the personalization rate $p\% \in [0,1]$, a subset of the shared parameters is gradually converted into personalized ones for the next round. Specifically, the mask at the network positions corresponding to the top $p\%$ largest values in $|\widetilde{\Delta u}_i^{t+1}|$ is changed from 0 to 1, resulting in a new binary mask m_i^{t+1}, which will serve as the input for the local update function in the next round. Through the above operations, $p\%$ of parameters are personalized per communication round until the final round, when the proportion hits the personalization threshold $\beta \in [0,1]$. It should be noted that we adopt a per-round expansion strategy for the personalization scope, rather than redefining personalisation partition based on β at each round, in order to avoid insufficient global knowledge. This mechanism balances global knowledge accumulation in the early rounds with personalized knowledge accumulation in the later rounds, effectively reconciling the model's generalization ability and personalization ability.

Moreover, we argue the personalization threshold β should be dynamically adjusted to noise intensity σ:

$$\beta = \beta_0 \cdot \exp\left(a \cdot (\sigma - \sigma_0)\right) \tag{11}$$

where β_0 is the initial personalization threshold, a is a hyperparameter, and σ_0 is a moderate noise intensity. When the noise intensity is moderate ($\sigma = \sigma_0$), β takes the value of β_0. The exponential form in Eq. (11) is adopted due to the nonlinear relationship between noise intensity and model utility. As the noise intensity σ increases, the performance of the global model typically degrades at an accelerating rate. The growth property of the natural exponential function makes β increase more sensitively under high-noise conditions, thereby retaining more critical parameters locally to mitigate the impact of noise.

Client i uploads only the perturbed shared gradients $\widetilde{\Delta u}_i^{t+1}$. Given heterogeneous client masks, not all clients contribute to the same global parameter, making simple arithmetic averaging unsuitable. It is necessary to count the number of clients contributing to each global gradient to perform global aggregation.

$$\Delta w_g^{t+1} = \frac{\sum_i (1/|\mathcal{P}^t| \cdot \widetilde{\Delta u}_i^{t+1})}{\sum_i (1/|\mathcal{P}^t| \cdot \neg m_i^t)} \tag{12}$$

4.2 Adaptive Clipping-and-Noising Differential Privacy

In client models, the update norms across different layers may vary significantly, making a uniform clipping threshold no longer an appropriate solution. A uniform threshold leads to insufficient clipping for layers with small update norms, introducing disproportionately large noise relative to their scale, while layers with large update norms suffer from excessive clipping, resulting in the loss of important information. Ultimately, this imbalance prevents the system from converging stably and degrades overall model performance. Furthermore, as parameter changes become smaller during training, the impact of added noise grows [11]. To overcome these challenges, we design an ACNDP mechanism, which assigns an appropriate clipping threshold and noise scale to each layer's gradient, thereby improving the utility of the model.

In communication round t, client i obtains the shared gradients Δu_i^{t+1} after completing local update. First, the clipping operation is performed:

$$\overline{\Delta u}_{i,l}^{t+1} = \Delta u_{i,l}^{t+1} \cdot \min\left(1, \frac{C_{i,l}}{\|\Delta u_{i,l}^{t+1}\|_2}\right) \tag{13}$$

where the clipping threshold $C_{i,l}$ is the sensitivity of this layer. Each layer's clipping threshold square $C_{i,l}^2$ is obtained by multiplying the clipping threshold weight tw_i^t by the global clipping threshold square C^2:

$$C_{i,l}^2 = C^2 \cdot tw_{i,l}^t \tag{14}$$

When computing the noise to be added to the shared gradients for layer l, it is necessary to avoid the personalized part, the zero-value positions in $\overline{\Delta u}_i^{t+1}$:

$$\text{Noise}_{i,l}^{t+1} = \mathcal{N}\left(0, L\sigma^2 C_{i,l}^2 / |\mathcal{P}^t|\right) \odot \neg m_{i,l}^t \tag{15}$$

$$\widetilde{\Delta u}_{i,l}^{t+1} = \overline{\Delta u}_{i,l}^{t+1} + \text{Noise}_{i,l}^{t+1}, \tag{16}$$

where σ is the noise multiplier, $\mathcal{N}\left(0, L\sigma^2 C_{i,l}^2 / |\mathcal{P}^t|\right)$ denotes a Gaussian noise. The above noise performs the Hadamard product with the mask $\neg m_{i,l}^t$ of layer l, resulting in the noise $\text{Noise}_{i,l}^{t+1}$ with non-zero values only in the shared part, which gradually reduces the noise and facilitates model convergence.

The next round's clipping threshold weights tw_i^{t+1} are computed based on the trend of gradient norm changes. To ensure that tw_i^{t+1} can be adjusted at an appropriate scale, we perform computations in the log-odds space and carry out renormalization, as detailed in Algorithm 1. It should be noted that we adopt the proportion of parameter counts in each layer of the initial model as the initial clipping threshold weights. For example, given the initial clipping threshold weight for layer l as tw_l^0, we calculate its log-odds ratio as $h_l^0 = log(\frac{tw_l^0}{1-tw_l^0})$. The variable b in Algorithm 1 serves solely as a numerical indicator of the direction of gradient norm changes in a specific layer. It carries no raw gradient information, thus the entire process of allocating adaptive clipping thresholds does not introduce additional privacy leakage risk for clients.

Algorithm 1: Adaptive clipping threshold weights update

Require: Input: Log-odds h^t, the step size for log-odds update γ, the total number of model layers L, the shared gradient from the previous round Δu^t and current round Δu^{t+1},

Ensure: Output: h^{t+1}, tw^{t+1}

for $l = 0, 1, \ldots, L-1$ **do**
  if $\|\Delta u_l^{t+1}\|_2 > \|\Delta u_l^t\|_2$ **then**
    $b = 1$
  else
    $b = -1$
  end if
  $h_l^{t+1} = h_l^t + \gamma b$
  $tw_l^{t+1} = \frac{e^{h_l^{t+1}}}{1+e^{h_l^{t+1}}}$
end for
for $l = 0, 1, \ldots, L-1$ **do**
  $tw_l^{t+1} = \frac{tw_l^{t+1}}{\sum_j tw_j^{t+1}}$
end for

The pseudocode of FedGLP-ADP is outlined in Algorithm 2.

Algorithm 2: FedGLP-ADP

Require: Global epochs T, participants number $|\mathcal{P}^t|$, batched data $D_i = (b_{i,1}, b_{i,2}, \ldots, b_{i,K})$ of client i, global model parameters w_g and client local parameters w_i, personalized threshold β derived from Eq. (11),
learning rate η, personalization rate p%, and global clipping bound C.

Local Update:
for $i = 0, 1, \ldots, |\mathcal{P}^t| - 1$ **do**
  Receive w_g^t from Server
  $w_i^t = (u_i^t, v_i^t) \leftarrow (w_g^t \odot \neg m_i^t, w_i^t \odot m_i^t)$
  for $k = 0, 1, \ldots, K-1$ **do**
    $v_{i,k+1}^t \leftarrow v_{i,k}^t - \eta \nabla_v \mathcal{L}_1(u_{i,k}^t, v_{i,k}^t, b_{i,k})$ y $\mathcal{L}_1$ from Eq. (9)
    $u_{i,k+1}^t \leftarrow u_{i,k}^t - \eta \nabla_u \mathcal{L}_2(u_{i,k}^t, v_{i,k+1}^t, b_{i,k})$ by $\mathcal{L}_2$ from Eq. (10)
  end for

(Continued)

Algorithm 2 (continued)

$\Delta u_i^{t+1} = u_i^{t+1} - u_i^t$
update h_i^{t+1}, tw_i^{t+1} using Algorithm 1
for $l = 0, 1, \ldots, L-1$ **do**
  Clip and add noise using Eqs. (13) and (16)
  if sparsity of $\neg m_{i,l}^t < \beta$ **then**
    $m_{i,l}^{t+1} = m_{i,l}^t$ + mask for largest p% values in $|\widetilde{\Delta u}_{i,l}^{t+1}|$
  else
    $m_{i,l}^{t+1} = m_{i,l}^t$
  end if
end for
end for
Server Execute:
for $t = 0, 1, \ldots, T$ **do**
  $w_g^{t+1} = w_g^t + \frac{\sum_i (1/|\mathcal{P}^t| \cdot \widetilde{\Delta u}_i^{t+1})}{\sum_i (1/|\mathcal{P}^t| \cdot \neg m_i^t)}$
  for $i = 0, 1, \ldots, |\mathcal{P}^t| - 1$ **do**
    Send w_g^{t+1} to participant i
  end for
end for

4.3 Privacy Analysis

In this subsection, we will prove that the proposed method satisfies the (ϵ, δ)-DP condition. First, we introduce some lemmas [25] that will support our proof.

Lemma 1 (Gaussian Mechanism of Rényi Differential Privacy (RDP)): *Let $\mathcal{M}(D) = f(D) + \mathcal{N}(0, \sigma^2 I)$ be the Gaussian mechanism, where f has ℓ_2-sensitivity Δf. Then, the mechanism $\mathcal{M}$ satisfies $(\alpha, \frac{\alpha \Delta f^2}{2\sigma^2})$-RDP.*

Lemma 2 (Sequential Composition of RDP): *Let $\{\mathcal{M}_i\}_{i=1}^T$ be a sequence of randomized mechanisms applied to the same dataset. If each mechanism $\mathcal{M}_i$ satisfies (α, ϵ_i)-RDP, then their composition $\mathcal{M}(D) = (\mathcal{M}_1(D), \ldots, \mathcal{M}_T(D))$ satisfies $(\alpha, \sum_{i=1}^T \epsilon_i)$-RDP.*

Lemma 3 (Parallel Composition of RDP): *Let a dataset D be partitioned into disjoint subsets $D = \bigsqcup_{i=1}^T D_i$. Suppose that each mechanism $\mathcal{M}_i$ is applied only to subset D_i and satisfies (α, ϵ)-RDP. Then the combined mechanism $\mathcal{M}(D) = (\mathcal{M}_1(D_1), \ldots, \mathcal{M}_T(D_T))$ also satisfies (α, ϵ)-RDP.*

Lemma 4 (From RDP to DP): *If a mechanism $\mathcal{M}$ satisfies $(\alpha, \epsilon_\alpha)$-RDP, then for any $0 < \delta < 1$, $\mathcal{M}$ also satisfies $(\epsilon_\alpha + \frac{\log(1/\delta)}{\alpha - 1}, \delta)$-DP.*

Lemma 5 (Post-Processing): *If a algorithm $\mathcal{M}$ satisfies (ϵ, δ)-DP, then for any data-independent function f, the composition $f(\mathcal{M}(D))$ also satisfies (ϵ, δ)-DP.*

Theorem 1 (Privacy Guarantee of FedGLP-ADP): *The proposed algorithm satisfies $(\min_\alpha (\frac{T\alpha}{2\sigma^2} + \frac{\log(1/\delta)}{\alpha - 1}), \delta)$-DP, where T is the total number of communication rounds and σ is the noise multiplier of the Gaussian mechanism.*

Proof of Theorem 1: In FedGLP-ADP, the mask m_i^t is computed based on the noisy gradients $\widetilde{\Delta u}_{i,l}^t$. If $\widetilde{\Delta u}_{i,l}^t$ is already guaranteed to be (ϵ, δ)-DP, the mask m_i^t, as a function of $\widetilde{\Delta u}_{i,l}^t$, does not incur any additional privacy loss according to Lemma 5. Furthermore, the progressive expansion of the mask (from 0 to personalization

threshold β) is controlled by a globally fixed hyperparameter $p\%$, which is independent of local data complexity. This ensures that the rate of personalization does not reveal information about the client's dataset. In summary, the mask does not introduce additional privacy loss, and all that remains is to prove that the perturbation to gradients satisfies DP.

At round t, client i has L layers of parameters. For each layer l, let the clipping threshold be $C_{i,l}$, and the noise added to that layer be $\sigma_{i,l}^2 = \frac{L\sigma^2 C_{i,l}^2}{n}$, where n is the number of participating clients. According to Lemma 1, the resulting RDP loss for this layer is

$$\epsilon_{i,l} = \frac{\alpha n}{2L\sigma^2} \tag{17}$$

According to Lemma 2, the RDP loss for client i is $\epsilon_i = \frac{\alpha n}{2\sigma^2}$, which is equivalent to adding noise $\sigma_i^2 = \frac{C\sigma^2}{n}$, where

$$\Delta f^2 = \sum_l C_{i,l}^2 = C^2 \sum_l tw_{i,l} = C^2 \tag{18}$$

Due to the additivity of Gaussian noise, the total noise is $\sigma_g^2 = C\sigma^2$, so the privacy loss per round is $\epsilon = \frac{\alpha}{2\sigma^2}$. Over T rounds, the privacy loss is

$$\epsilon_{\text{total}} = \frac{T\alpha}{2\sigma^2} \tag{19}$$

Therefore, according to Lemma 4, our proposed method overall satisfies $\left(\min_\alpha \left(\frac{T\alpha}{2\sigma^2} + \frac{\log(1/\delta)}{\alpha - 1}\right), \delta\right)$-DP. □

We employ the RDP algorithm provided by Opacus as the privacy accountant to compute the optimal noise multiplier σ, ensuring that the cumulative privacy cost does not exceed the target (ϵ, δ)-DP requirement.

5 Experiments

In this section, we compare FedGLP-ADP with different methods of FL with DP. We use various datasets and learning settings to demonstrate the superiority of FedGLP-ADP.

5.1 Experimental Setup

Datasets and Models. To evaluate the performance of FedGLP-ADP in diverse IoT visual perception scenarios, we conduct experiments on three representative datasets: Fashion-MNIST [26], SVHN [27], and CIFAR-10 [28]. These datasets simulate various data types captured by distributed edge sensors:

- **Fashion-MNIST:** Fashion-MNIST represents grayscale sensor data in smart retail or logistics inventory systems, containing 10 categories of clothing items with a total of 70,000 images.
- **SVHN:** SVHN mimics smart city surveillance tasks such as automated street address recognition, consisting of color images of house numbers (0−9) collected from real-world street views.
- **CIFAR-10:** CIFAR-10 serves as a proxy for general-purpose object recognition at the network edge, covering 10 common categories with 6000 images per category.

To align with the limited computational resources and memory capacity of typical IoT devices, we adopt lightweight Convolutional Neural Network (CNN) architectures: Fashion-MNIST and SVHN both

use networks with two convolution layers followed by two fully connected layers, while CIFAR-10 uses a CNN with three convolution layers and three fully connected layers, as well as ResNet-18.

Comparison Methods. Our goal is to improve the performance of DP-FedAvg [24], so we select it as the baseline. In addition, we compare our method with three state-of-the-art methods of FL with DP. There is a brief description of these methods as follows:

- **DP-FedAvg [24]:** It is the first method to implement user-level DP in FL by applying the Gaussian mechanism, and provides a baseline level of model performance under privacy protection.
- **BLUR+LUS [9]:** This algorithm improves model quality without sacrificing privacy by combining bounded local update regularization and local update sparsification.
- **FedDPA [10]:** FedDPA achieves flexible personalization via dynamic fisher personalization and adaptive constraints, mitigating slow convergence caused by clipping operations and improving both performance and clipping resilience.
- **FedADDP [29]:** FedADDP proposes a PFL algorithm with adaptive dimensional DP, which reduces the accuracy loss of personalized models caused by DP.

Evaluation Metrics. We use average accuracy as the performance metric for the models. Specifically, each client trains a local model on its own dataset, and the accuracy of each client's model is computed individually. The average of these accuracies serves as the final evaluation metric. To thoroughly validate the effectiveness of our method, we designed corresponding evaluation schemes for different datasets. For Fashion-MNIST and SVHN, we measure average accuracy under varying privacy budgets to demonstrate our method's advantages in noise resilience. For CIFAR-10, we evaluate average accuracy under different degrees of non-IID data distributions, highlighting the strengths of our personalized algorithm in handling heterogeneous data.

Implementation Details. For a fair comparison, all methods adopt the same network architecture and hyperparameter settings. We set the number of clients to 10 and the sampling rate to 1. For all datasets, the Adam optimizer is used with a learning rate of 1e−3, local training rounds set to 1, and batch size set to 16. The DP slack parameter δ is set to the reciprocal of the number of participating clients, and the clipping bound is set to 0.5. Unless otherwise specified, for Fashion-MNIST and SVHN, the number of global rounds is set to 20; while for CIFAR-10, it is set to 40, with a noise multiplier of 0.3. In addition, we simulate non-IID data distributions using a Dirichlet distribution $Dir(\alpha)$ [30,31], where smaller α values indicate higher data heterogeneity. All codes are implemented in Python using the PyTorch framework and executed on an NVIDIA 3090 GPU.

5.2 Results and Discussion

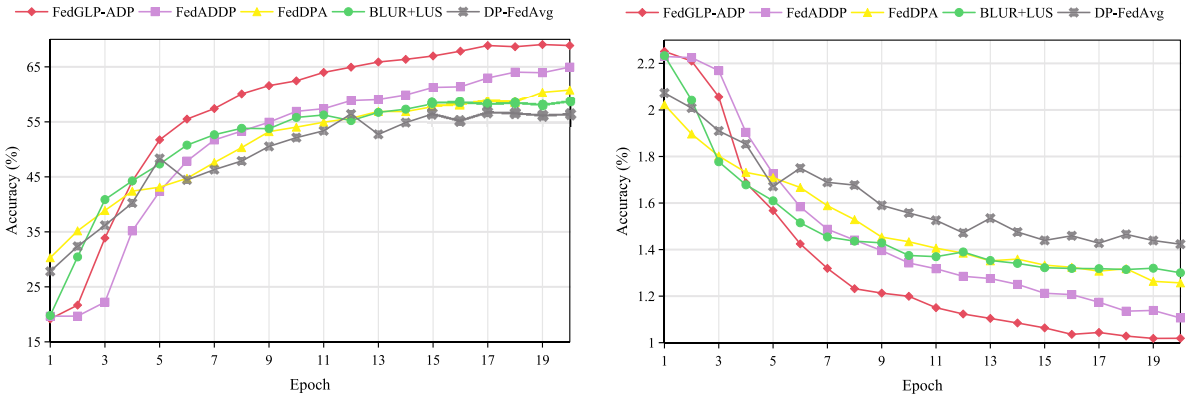
For FedGLP-ADP, the initial personalization threshold β_0 is set to 0.3, the adjustment factor a is set to 0.2, and the initial noise intensity σ_0 is the noise intensity when $\epsilon = 6$. The initial log-odds ratio h_l^0 depends solely on the number of parameters in each layer of the model.

Results under Different Privacy Budgets. We evaluate the performance of each method on the Fashion-MNIST and SVHN datasets under various privacy budgets, and the results are summarized in Table 1. The experimental results show that FedGLP-ADP achieves higher accuracy than other methods under all privacy budgets. Particularly in high-noise environments, the accuracy of FedGLP-ADP still outperforms the best method by 1.88% and 4.87% respectively, demonstrating its robustness against noise. To visually illustrate the convergence behavior of our method, we plotted the client-wise average accuracy and loss curves on the SVHN dataset (with a privacy budget of $\epsilon = 16$). As shown in Fig. 2, FedGLP-ADP steadily

increases in accuracy and decreases in loss as training progresses, exhibiting faster convergence compared to other methods.

Table 1: Accuracy (%) under different privacy budgets.

Dataset	Methods	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$	$\epsilon = 16$
Fashion-MNIST	DP-FedAvg	77.91	77.85	79.16	79.31	79.24
	BLUR+LUS	78.05	78.85	79.56	79.73	79.74
	FedDPA	78.55	79.00	79.87	80.57	80.58
	FedADDP	78.93	79.90	80.70	80.83	81.13
	FedGLP-ADP	80.33	80.79	81.26	81.47	82.09
SVHN	DP-FedAvg	51.48	53.33	54.48	55.24	56.41
	BLUR+LUS	54.92	55.59	56.86	57.84	58.58
	FedDPA	56.89	57.94	58.70	60.07	60.77
	FedADDP	60.87	61.41	64.02	63.96	64.96
	FedGLP-ADP	62.60	65.91	66.37	67.98	69.05



(a) Accuracy curve

(b) Loss curve

Figure 2: Comparison of model accuracy and loss variation.

Results under Different Non-IID Degrees. We evaluate the performance of each method under different non-IID degrees, with results in Table 2. The results indicate that FedGLP-ADP achieves the highest accuracy under all non-IID degrees, demonstrating that it enhances the robustness agaist non-IID data.

To further investigate the robustness of FedGLP-ADP under complex architectures, we conduct supplementary experiments using the ResNet-18 model on the CIFAR-10 dataset ($\alpha = 1$). Fig. 3 illustrates the accuracy convergence curves of all methods. The results indicate that on the deeper ResNet-18 model, the performance gap among different methods becomes more pronounced. DP-FedAvg, BLUR+LUS, and FedDPA, which employ static and uniform clipping thresholds, fail to accommodate the substantial gradient variation across layers in deep architectures, leading to severe accuracy stagnation. In contrast, while FedADDP exhibits a certain degree of adaptability (34.94%), FedGLP-ADP achieves a superior accuracy of 43.54%. This strongly validates the superiority of our method in handling large-scale parameter spaces and complex model structures.

Table 2: Accuracy (%) under different non-IID degrees.

Dataset	Methods	$\alpha = 0.1$	$\alpha = 1$	$\alpha = 10$	$\alpha = 100$	IID
CIFAR-10	DP-FedAvg	18.55	31.17	39.88	41.47	42.57
	BLUR+LUS	19.08	32.75	41.98	43.15	43.79
	FedDPA	19.52	31.89	42.55	42.80	44.05
	FedADDP	21.06	34.94	45.46	48.51	48.77
	FedGLP-ADP	22.52	36.11	46.25	49.38	49.87

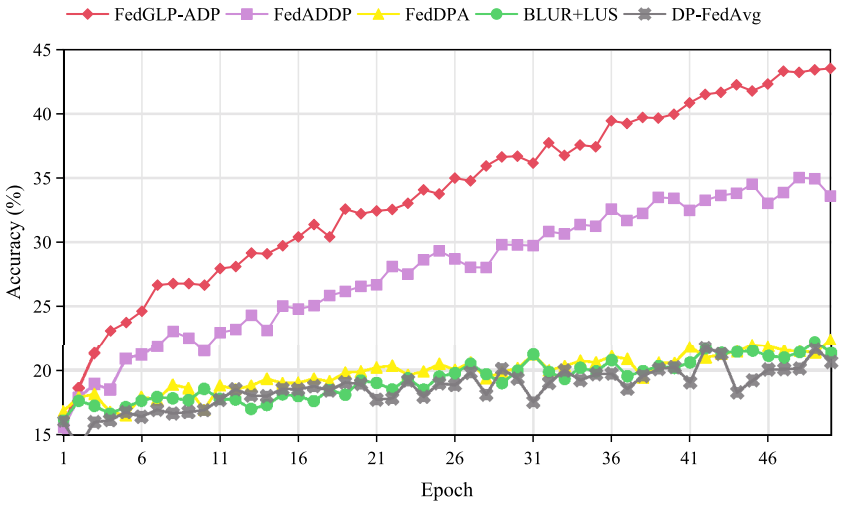


Figure 3: Performance comparison on CIFAR-10 dataset ($\alpha = 1$) using ResNet-18 architecture.

Study on the Personalization Threshold. We compare the model accuracy when the personalization threshold β is fixed vs. dynamically adjusted. Table 3 reports the personalization thresholds and corresponding model accuracies under different privacy budgets. It can be observed that the dynamically adjusted threshold consistently outperforms the fixed threshold, demonstrating that this design better balances privacy protection and model utility.

Table 3: Performance comparison under different modes.

ϵ	2		4		16	
Mode	**Fixed**	**Dynamic**	**Fixed**	**Dynamic**	**Fixed**	**Dynamic**
β	0.30	0.45	0.30	0.34	0.30	0.25
Accuracy (%)	59.95	62.60	65.62	65.91	68.48	69.05

Study on the Log-Odds Update Step Size. We investigate the impact of the log-odds update step size γ on performance to reveal its sensitivity. Table 4 presents the model accuracy for $\gamma = \{0, 0.2, 0.4, 0.8, 1.6\}$. When $\gamma = 0$, the clipping weights remain fixed at their initial values, failing to accommodate the dynamic gradient variation across layers, which leads to a drop in accuracy. In contrast, for all non-zero value of γ (ranging from 0.2 to 1.6), the accuracy remains relatively stable, demonstrating excellent robustness.

This result indicates that FedGLP-ADP has low dependency on the step size hyperparameter, achieving competitive and stable performance without extensive tuning in practical deployments—significantly reducing deployment costs in resource-constrained IoT scenarios.

Table 4: Performance comparison under different γ.

γ	0	0.2	0.4	0.8	1.6
Accuracy (%)	65.66	69.10	69.36	69.60	68.95

Ablation Experiments. To investigate the contribution of each component in FedGLP-ADP to the overall performance, we conduct a series of ablation experiments on the SVHN dataset ($\alpha = 1$). The results are shown in Table 5.

Table 5: Results of ablation experiments (%).

Components		$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 6$	$\epsilon = 8$	$\epsilon = 16$
GLP	ACNDP					
\	\	39.97	43.92	45.91	47.03	48.38
√	\	44.29	48.02	49.38	51.67	52.62
√	√	46.89	51.65	52.15	54.34	54.63

Gradient-Based Layer-Wise Personalization (GLP). Compared with DP-FedAvg (first row in the table), DP-FedAvg+GLP (second row) achieves significant improvements in model performance under all privacy budgets, demonstrating the effectiveness of GLP in enhancing model accuracy under privacy protection.

Adaptive Clipping and Noising Differential Privacy (ACNDP). ACNDP is inherently and closely coupled with the GLP strategy. Specifically, the adaptive logic of ACNDP—both the allocation of clipping thresholds based on shared gradients and the noise injection scaled by the current degree of personalization—is specifically designed in response to the evolving personalization boundary established by GLP. Therefore, we did not report results for DP-FedAvg+ACNDP alone. Instead, ACNDP was applied on top of GLP, forming the complete FedGLP-ADP method (third row in the table). The results show that the addition of ACNDP further improves model accuracy, confirming that the synergy between ACNDP and GLP can significantly enhance the overall performance of the method.

Analysis of Communication Overhead. Unlike standard FL approaches (such as DP-FedAvg), which require uploading the full model parameters in each round, FedGLP-ADP reduces uplink communication overhead through its progressive personalization strategy. Let M be the total number of model parameters. In round t, the number of personalized parameters is approximately $32 \cdot M \cdot t \cdot (\beta/T)$. Since only shared gradients and mask are uploaded to the server, the communication load for client i in round t, denoted as S_t, is:

$$S_t = 32 \cdot M \times \left(1 - \frac{t \cdot \beta}{T}\right) + M \tag{20}$$

As the training progresses, the communication overhead decreases linearly. By the final round T, the per-round communication volume is reduced to $(32 \cdot M \cdot (1 - \beta) + M)$.

The total communication cost over T rounds is $\sum_t^T S_t = 32 \cdot M \cdot T \cdot (1 - \frac{\beta}{2}) + M \cdot T = M \cdot T \cdot (33 - 16 \cdot \beta)$. Compared with the total cost of standard FL $32 \cdot M \cdot T$, FedGLP-ADP achieves a bandwidth saving of $\frac{16 \cdot \beta - 1}{32}$. For example, when the personalization threshold $\beta = 0.3$, the total data transmission is reduced by 11.875%. Therefore, the transmission of masks not only does not increase communication overhead but actually reduces it to some extent, making it more suitable for IoT sensors with limited transmission bandwidth.

Study on Computational Overhead. To evaluate the computational overhead of FedGLP-ADP, we measure the average wall-clock time required per local training round for different methods and model architectures on the CIFAR-10 dataset. The result in Table 6 shows a significant efficiency gap between FedGLP-ADP and other state-of-the-art methods. While other advanced methods incur substantial computational overhead (ranging from 4.22× to 10.55×), FedGLP-ADP still demonstrates an excellent lightweight nature, with relative computational overheads of only 1.74× (CNN) and 1.92× (ResNet-18). Moreover, as the number of model parameters increases substantially, the growth in relative computational overhead of FedGLP-ADP is much smaller compared to other methods. This is because the sparsification and personalization strategies of other methods involve relatively complex computations (e.g., Fisher information), while FedGLP-ADP avoids such computations, making it a feasible solution for resource-constrained IoT devices.

Table 6: Comparison of average execution time per round (s).

Architecture	DP-FedAvg	BLUR+LUS	FedDPA	FedADDP	FedGLP-ADP
CNN	3.36 (1.00×)	13.57 (4.04×)	19.52 (5.81×)	21.10 (6.28×)	5.84 (1.74×)
ResNet-18	8.91 (1.00×)	46.53 (5.22×)	65.66 (7.37×)	93.98 (10.55×)	17.07 (1.92×)

6 Conclusion

This paper proposes a novel FL method, FedGLP-ADP, which consists of two core components: gradient-based layer-wise personalization (GLP) and adaptive clipping-and-noising DP (ACNDP). GLP leverages the variation of model parameters before and after local training to perform personalized partitioning of the model, while ACNDP implements adaptive allocation of clipping thresholds and determines the magnitude of added noise based on the personalization degree achieved by GLP. This design overcomes the limitations of coarse-grained parameter decoupling and the convergence challenges faced by traditional DP. Experimental results demonstrate that FedGLP-ADP consistently outperforms existing methods under various privacy budgets and different degrees of non-IID data distributions.

Acknowledgement: None.

Funding Statement: The work was supported by the Chongqing Research Program of Basic Research and Frontier Technology (Chongqing Talent) (Grant No. cstc2024ycjh-bgzxm0048).

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Di Xiao; methodology, Wenting Jiang and Min Li; software, Wenting Jiang and Min Li; validation, Di Xiao and Wenting Jiang; writing—original draft preparation, Wenting Jiang; writing—review and editing, Di Xiao; visualization, Wenting Jiang and Min Li; supervision, Wenting Jiang; project administration, Di Xiao. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this paper are respectively in references [26–28].

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. McMahan B, Moore E, Ramage D, Hampson S, Arcas BA. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS); 2017 Apr 20–22; Fort Lauderdale, FL, USA. Westminster, UK: PMLR; 2017. Vol. 54, p. 1273–82.
2. Fredrikson M, Jha S, Ristenpart T. Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security; 2015 Oct 12–16; Denver, CO, USA. New York, NY, USA: ACM; 2015. p. 1322–33. doi:10.1145/2810103.2813677.
3. Geiping J, Bauermeister H, Dröge H, Moeller M. Inverting gradients-how easy is it to break privacy in federated learning? In: Advances in Neural Information Processing Systems; 2020 Dec 6–12; Virtual Event. Vol. 33, p. 16937–47.
4. Xie Q, Jiang S, Jiang L, Huang Y, Zhao Z, Khan S, et al. Efficiency optimization techniques in privacy-preserving federated learning with homomorphic encryption: a brief survey. IEEE Internet Things J. 2024;11(14):24569–80. doi:10.1109/JIOT.2024.3382875.
5. Mohassel P, Zhang Y. Secureml: a system for scalable privacy-preserving machine learning. In: Proceedings of the 38th IEEE Symposium on Security and Privacy (SP); 2017 May 22–26; San Jose, CA, USA. p. 19–38. doi:10.1109/SP.2017.12.
6. Dwork C, Roth A. The algorithmic foundations of differential privacy. Found Trends® Theor Comput Sci. 2014;9(3–4):211–407. doi:10.1561/0400000042.
7. Zhu Y, Yu X, Tsai YH, Pittaluga F, Faraki M, Wang YX, et al. Voting-based approaches for differentially private federated learning. arXiv:2010.04851. 2020.
8. Hu R, Guo Y, Gong Y. Federated learning with sparsified model perturbation: improving accuracy under client-level differential privacy. IEEE Trans Mobile Comput. 2023;23(8):8242–55. doi:10.1109/TMC.2023.3343288.
9. Cheng A, Wang P, Zhang XS, Cheng J. Differentially private federated learning with local regularization and sparsification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2022 Jun 21–24; New Orleans, LA, USA. p. 10122–31. doi:10.1109/CVPR52688.2022.00988.
10. Yang X, Huang W, Ye M. Dynamic personalized federated learning with adaptive differential privacy. In: Advances in Neural Information Processing Systems; 2023 Dec 10–16; New Orleans, LA, USA. Vol. 36, p. 72181–92. doi:10.52202/075280-3160.
11. Jiao S, Cai L, Wang X, Cheng K, Gao X. A differential privacy federated learning scheme based on adaptive gaussian noise. Comp Model Eng Sci. 2024;140(2):1679–94. doi:10.32604/cmes.2023.030512.
12. Tan AZ, Yu H, Cui L, Yang Q. Towards personalized federated learning. IEEE Trans Neural Netw Learn Syst. 2022;34(12):9587–603. doi:10.1109/TNNLS.2022.3160699.
13. Arivazhagan MG, Aggarwal V, Singh AK, Choudhary S. Federated learning with personalization layers. arXiv:1912.00818. 2019.
14. Oh J, Kim S, Yun SY. Fedbabu: towards enhanced representation for federated image classification. In: Proceedings of the 9th International Conference on Learning Representations (ICLR); 2021 May 3–7; Virtual Event.
15. Collins L, Hassani H, Mokhtari A, Shakkottai S. Exploiting shared representations for personalized federated learning. In: Proceedings of the 38th International Conference on Machine Learning (ICML); 2021 Jul 18–24; Virtual Event. Westminster, UK: PMLR. Vol. 139, p. 2089–99.
16. Xu J, Tong X, Huang SL. Personalized federated learning with feature alignment and classifier collaboration. In: Proceedings of the 11th International Conference on Learning Representations (ICLR); 2023 May 1–5; Kigali, Rwanda.
17. Yang X, Huang W, Ye M. Fedas: bridging inconsistency in personalized federated learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2024 Jun 17–21; Seattle, WA, USA. p. 11986–95.
18. Zhang G, Liu B, Zhu T, Ding M, Zhou W. PPFed: a privacy-preserving and personalized federated learning framework. IEEE Internet Things J. 2024;11(11):19380–93. doi:10.1109/JIOT.2024.3360153.

19. Wang D, Gao Y, Pang S, Zhang C, Zhang X, Li M. FedMPS: a robust differential privacy federated learning based on local model partition and sparsification for heterogeneous IIoT data. IEEE Internet Things J. 2025;12(10):13757–68. doi:10.1109/JIOT.2025.3536035.
20. Liang PP, Liu T, Liu Z, Allen NB, Auerbach RP, Brent D, et al. Think locally, act globally: federated learning with local and global representations. arXiv:2001.01523. 2020.
21. Jang J, Ha H, Jung D, Yoon S. Fedclassavg: local representation learning for personalized federated learning on heterogeneous neural networks. In: Proceedings of the 51st International Conference on Parallel Processing (ICPP); 2022 Aug 29–Sep 1; Bordeaux, France. p. 1–10. doi:10.1145/3545008.3545073.
22. Yi L, Yu H, Shi Z, Wang G, Liu X, Cui L, et al. FedSSA: semantic similarity-based aggregation for efficient model-heterogeneous personalized federated learning. arXiv:2312.09006. 2023.
23. Renda A, Frankle J, Carbin M. Comparing rewinding and fine-tuning in neural network pruning. In: Proceedings of the 8th International Conference on Learning Representations (ICLR); 2020 Apr 26–30; Addis Ababa, Ethiopia (Virtual Event).
24. McMahan HB, Ramage D, Talwar K, Zhang L. Learning differentially private recurrent language models. In: Proceedings of the 6th International Conference on Learning Representations (ICLR); 2018 Apr 30–May 3; Vancouver, BC, Canada.
25. Mironov I. Rényi differential privacy. In: 2017 IEEE 30th Computer Security Foundations Symposium (CSF); 2017 Aug 21–25; Santa Barbara, CA, USA. p. 263–75. doi:10.1109/CSF.2017.11.
26. Xiao H, Rasul K, Vollgraf R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747. 2017.
27. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng AY, et al. Reading digits in natural images with unsupervised feature learning. In: NIPS 2011 Workshop on Deep Learning and Unsupervised Feature Learning; 2011 Dec 12–17; Granada, Spain.
28. Krizhevsky A. Learning multiple layers of features from tiny images [master's thesis]. Toronto, ON, Canada: University of Toronto; 2009.
29. Guo Y, Zhang T, Mu X, Li H, Dong X, Li Q. FedADDP: privacy-preserving personalized federated learning with adaptive dimensional differential privacy. In: Proceedings of the 24th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP); 2024 Oct 25–27; Guilin, China. p. 22–40. doi:10.1007/978-981-96-1548-3_3.
30. Hsu TMH, Qi H, Brown M. Measuring the effects of non-identical data distribution for federated visual classification. arXiv:1909.06335. 2019.
31. Zhu Z, Hong J, Zhou J. Data-free knowledge distillation for heterogeneous federated learning. In: Proceedings of the 38th International Conference on Machine Learning (ICML); 2021 Jul 18–24; Virtual Event. Westminster, UK: PMLR. Vol. 139, p. 12878–89.