ADP-FL: Federated Learning Based on Adaptive Differential Privacy

Abstract. Federated Learning (FL) is a distributed machine learning framework where each participant achieves model training by exchanging model parameters. However, when the server is dishonest, it may lead to the leakage of the participants' private data. In recent years, many research works have combined federated learning with differential privacy. To ensure user-level differential privacy in federated learning algorithms, the model updates transmitted by the clients need to be clipped before adding noise. However, a fixed clipping threshold may lead to loss of information or the addition of larger noise, especially in scenarios with data heterogeneity where the updates vary greatly among clients, which can impact the performance of the aggregated model. Therefore, in this paper, we propose a Federated Learning Based on Adaptive Differential Privacy (ADP-FL) that dynamically adjusts the clipping threshold based on changes in the norm of the update, aiming to align it as closely as possible with the trend of local updates. To address the problem of privacy protection imbalance among clients caused by a uniform clipping threshold, we introduce a privacy budget allocation strategy that allocates privacy budgets based on the influence of the clipping threshold on the norm of update, to better balance model performance and privacy protection. Finally, we evaluate the performance of the ADP-FL algorithm on the CIFAR-10, CIFAR-100, and MNIST datasets. The experimental results show that our proposed algorithm not only provides effective privacy protection but also exhibits better performance.

Keywords: User-level Differential Privacy, Federated Learning, Adaptive Clipping.

1 Introduction

Federated learning [1] is a distributed machine learning approach in which each client's data is stored and trained locally, and only updated parameters are transmitted to the server for aggregation, thus jointly training a global model. Although federated learning keeps data local to avoid data sharing across domains, model parameters, and gradient information shared between clients can be potentially at risk of privacy leakage [2]. Malicious attackers can utilize various attacks, such as inference attacks [3], and model inversion attacks [4], to obtain the privacy information of clients [5-7]. To achieve privacy protection, various techniques such as homomorphic encryption [8], secure multi-party computation [9], and differential privacy [10-11] are usually used for privacy protection. Both homomorphic encryption and secure multi-party computation are cryptography-based methods to encrypt the communication content, as the encryption process requires additional arithmetic power, and has the limitations of high interaction cost and high computational overhead. Differential privacy, on the other hand, does not need to encrypt the communication content. Hence, the computational cost is low and

has strict mathematical proofs and reliable background knowledge of information theory, so it has become a research hotspot in the field of data security.

However, privacy issues may become more complex in a data heterogeneous scenario. Different users have significantly different data distributions, resulting in varying privacy requirements. Additionally, as the number of iterations in model training increases, the norm of model parameter updates gradually decreases. If a fixed clipping threshold is used, it may not accurately capture the changes in the norm, leading to excessive clipping of information from certain clients, while others remain almost unaffected, or even more noise may be added due to too large a clipping threshold.

To solve the above problems, this paper proposes a new framework called ADP-FL. This framework enhances privacy protection by applying user-level differential privacy to the model and proposes an adaptive clipping strategy. By dynamically adjusting the clipping threshold in each round, it better protects privacy while ensuring model performance. Furthermore, based on the clipping threshold and the norm of the parameter update, this paper proposes a privacy budget allocation strategy to solve the imbalance problem caused by a uniform clipping threshold among clients, ensuring that each client's privacy is fully protected while optimizing model performance. The main contributions of this paper are as follows:

- We propose a federated learning based on adaptive differential privacy (ADP-FL), in which we consider the impact of data heterogeneity on model updates and adjust the clipping threshold according to the trend of the historical L2 norm of each client to make it as consistent as possible with the local update trend. By dynamically adjusting the clipping threshold in each round, we aim to achieve the optimal balance among different clients.
- We propose a privacy budget allocation strategy based on the above adaptive clipping, which allocates the privacy budget based on the relationship between the clipping threshold and the L2 norm. The privacy budget allocation can better satisfy the privacy requirements of different clients and improve the model performance and privacy protection.
- We evaluated the performance of the ADP-FL algorithm on the CIFAR-10, CIFAR-100, and MNIST datasets. The experimental results demonstrate that our proposed algorithm achieves higher accuracy while providing strong privacy protection.

The rest of this paper is organized as follows. Section 2 reviews related work; Section 3 briefly introduces the required preparatory knowledge; Section 4 introduces the framework and algorithms proposed in this paper, and gives a formal privacy analysis; We demonstrate the evaluation results in Section 5; Finally, we conclude this paper in Section 6.

2 Related Works

Differential privacy ensures privacy in the process of parameter transfer for federated learning. Currently, research on differential privacy-based federated learning mainly focuses on user-level differential privacy and sample-level differential privacy.

In user-level differential privacy, McMahan et al. [12] were the first to introduce user-level differential privacy into federated learning. They protected user privacy by clipping and adding noise to the updates. Zhang et al. [13] conducted an in-depth study on the theoretical and empirical aspects of clipping operations in federated learning algorithms. The research showed that even with significant data heterogeneity, the clipped FedAvg algorithm still exhibits remarkable performance. McMahan et al. [1] first defined and studied adaptive clipping techniques applied to differential privacy models. They adaptively adjusted the clipping threshold based on the gradient values in the training model, aiming to protect privacy while preserving useful information and minimizing the impact on model accuracy. Geyer et al. [14] proposed a user-level differential privacy federated learning framework called CS-DPFL. In this framework, they measured the gradient updates using the L2 norm and applied median clipping on the L2 norm. Andrew et al. [15] proposed the DP-FedAvgAC method, which considers the impact of the clipping threshold on model availability. They introduced a method to dynamically compute the threshold by sacrificing a portion of the privacy budget. This adaptive threshold improved model availability but increased the algorithm's complexity.

In sample-level differential privacy, Noble et al. [16] introduced machine learning based on sample-level differential privacy. They obtained sensitivity through per-sample gradient clipping and added Gaussian noise to local stochastic gradients before uploading them to the server for aggregation. Xu et al. [17] proposed the ADADP algorithm, which adopted an adaptive strategy similar to RMSPROP. They adaptively clipped and added noise to each gradient variable based on the historical gradient's mean squared variance. Wang et al. [18] proposed the DP-FedMeta algorithm, which used a sliding window to store the historical gradients from the previous five rounds. They selected the clipping threshold for the next round based on percentile values. This algorithm was applied in the context of federated meta-learning to protect privacy.

3 Preliminaries

3.1 Federated Learning

Federated learning can be defined as follows: suppose there are N users who want to participate in the training process of machine learning, and the datasets of these users are denoted as $\{D_1, D_2, \dots, D_N\}$. Each user is trained using their local dataset by minimizing the local loss function:

$$w_i^* = arg\,min\,L(D_i, w_i) \tag{1}$$

where $L(D_i, w_i)$ is the loss function of user i on the dataset D_i.

The user sends the updated model to the server to update the global model parameters through averaging or weighted averaging methods. The formula is as follows:

$$W_G = \sum_{i=1}^{N} p_i w_i \tag{2}$$

where w_i is the model parameter uploaded by the i-th participating user, W_G is the model parameter after aggregation, N is the number of participating users, and p_i represents the proportion of data from the i-th participating user in relation to the total amount of data.

3.2 Differential Privacy

In federated learning, differential privacy can be divided into sample-level differential privacy and user-level differential privacy. The definition of these two types of differential privacy is dependent on the definition of neighboring datasets.

Definition 1 (Sample-level differential privacy) [11]: Let $M: D \rightarrow \mathrm{R}$ be a random mechanism, where D and D' are neighboring datasets that differ by at most one record. If for any output $S_M \in R$ generated by the randomized mechanism M on D and D', the following inequality holds, then the mechanism M satisfies $(\varepsilon, \delta) - DP$.

$$\Pr[\mathrm{M}(D) \in S_M] \leq e^{\epsilon} \times \Pr[\mathrm{M}(D') \in S_M] + \delta \tag{3}$$

where the parameter ε denotes the privacy budget, which reflects the degree of privacy protection of the algorithm, the smaller ε is the higher the degree of privacy protection. δ is a relaxation factor indicating the probability of privacy disclosure.

Definition 2 (User-level differential privacy) [12]: Let D and D' be neighboring datasets and differ by only one user's all data. For a randomized algorithm M, and any output $S \in Range(M)$, the randomized algorithm M satisfies (ε, δ)-user-level differential privacy if and only if Equation (4) holds:

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

Definition 3 $((\boldsymbol{\alpha}, \boldsymbol{\varepsilon}) - \boldsymbol{RDP}$ [19]): For any two neighboring datasets D and D', if there exists a randomized mechanism $M: \mathrm{D} \rightarrow R$ that satisfies the following Equation (5), then the randomized mechanism M is said to satisfy $(\alpha, \varepsilon) - RDP$.

$$D_{\alpha}(M(D)||M(D')) = \frac{1}{\alpha - 1} \log\left(E_{\theta \sim M(D')} \left[\left(\frac{M(D)(\theta)}{M(D')(\theta)} \right)^{\alpha} \right] \right) \leq \varepsilon \tag{5}$$

where $D_{\alpha}(M(D)||M(D')) \leq \varepsilon$ indicates that the Rényi divergence between the neighboring datasets is restricted to the privacy parameter ε, when $\alpha \rightarrow \infty$, $(\alpha, \varepsilon) - RDP$ reduces to $(\varepsilon, 0) - DP$.

Proposition 1 (Gaussian Mechanism of RDP [19]): Let $S = max_{D,D'} \|M(D) - M(D')\|_2$ be the L_2 sensitivity of function M. Then, for the Gaussian mechanism $f(D) = M(D) + N(0, \sigma^2 I)$, it satisfies $(\alpha, \alpha S^2 \,/\, 2\sigma^2) - RDP$.

Proposition 2 (Conversion from RDP to $(\varepsilon, \delta) - DP$ [19]): If M is a $(\alpha, \varepsilon) - RDP$ mechanism, then for $\forall \delta \in (0,1)$, mechanism M also satisfies $(\varepsilon^*, \delta) - DP$, where ε^* is defined as follows.

$$\varepsilon^* = \varepsilon + \frac{\log(1/\delta)}{\alpha - 1} \tag{6}$$

Proposition 3 (Sequential composition theorem of RDP [19]): Let $M_1: D \to R_1$ satisfy $(\alpha, \varepsilon_1) - RDP$, $M_2: D \to R_2$ satisfy $(\alpha, \varepsilon_2) - RDP$. Then the sequential composition mechanism $M_{1,2}$ of M_1 and M_2 satisfies $(\alpha, \varepsilon_1 + \varepsilon_2) - RDP$.

Proposition 4 (Parallel Composition Theorem of RDP [20]): Assuming mechanism M consists of n random mechanisms $M_1, \ldots, M_n$, where each mechanism M_i satisfies $(\alpha, \varepsilon_i) - RDP$. Let there be n mutually independent datasets $D_1, \ldots, D_n$. The composite mechanism $M(M_1(D_1), \ldots, M_n(D_n))$ formed by these mechanisms satisfies $\left(\alpha, \max_i \varepsilon_i\right) - RDP$.

Proposition 5 (Postprocessing [21]): Suppose there is a mechanism M satisfies $(\varepsilon, \delta) - DP$ and another randomized mechanism A. The mechanism M after the operation of mechanism A still satisfies $(\varepsilon, \delta) - DP$.

4 Federated Learning Based on Adaptive Differential Privacy

In federated learning, we avoid the leakage of data privacy by only uploading model parameters rather than raw data. However, recent research has found that attackers can still infer partial private data of users by analyzing the uploaded parameters. To solve this problem, a common approach is to employ differential privacy techniques to protect data privacy. Within differential privacy, the technique of clipping plays a crucial role. It mainly truncates or scales the parameters before adding noise to limit the range of parameters, thus reducing the risk of privacy leakage. It is a common practice to add noise by clipping the local model parameters, yet existing clipping techniques still have certain limitations. Most clipping techniques use a fixed clipping threshold, not taking into account the dynamic changes of parameters during the training process. Moreover, determining an appropriate fixed clipping threshold is a significant challenge, as the setting of the clipping threshold directly impacts the model's performance. A threshold set too low may lead to excessive clipping of local updates, resulting in the loss of a substantial amount of update information; while a threshold set too high, although preserving more update information, may introduce more noise, which could affect the accuracy of the model.

To address these issues, this paper proposes federated learning based on adaptive differential privacy (ADP-FL). In this framework, the clipping threshold should be set in line with the trend of local updates, and the clipping threshold is dynamically adjusted in each round by observing the trend of the historical L2 norm of each client, aiming to achieve the optimal balance among clients. However, in federated learning scenarios with high data heterogeneity, there may be significant differences in the data distributions among clients, leading to large gaps in their respective model updates. Since the clipping threshold adjusted in each round needs to consider the updated trends of all clients, it can only reflect the overall demand of these clients to a certain extent. Therefore, we resolve the imbalance caused by the uniform clipping threshold among clients by allocating a privacy budget based on the clipping threshold in each round and

the updated amount of clients. This ensures that while each client receives sufficient privacy protection, model performance is also optimized.

Table 1. Main symbols.

Symbols	Description
D_i	Local datasets
G	Global model
Ens	Ensemble model
L	Number of personalization layer
W_b^t	The base layer of the model
W_L^t	The personalization layer of the model
s	Standard deviation
ε	Privacy budget
Δ_i	Update
S_w	Sliding window
C	Clipping threshold

In this section, we first present the ADP-FL framework as shown in Fig. 1. The main symbols involved in the paper are provided in Table 1. We then introduce the relevant algorithms involved in the training process. Finally, we analyze the differential privacy guarantee provided by ADP-FL and provide corresponding proofs.

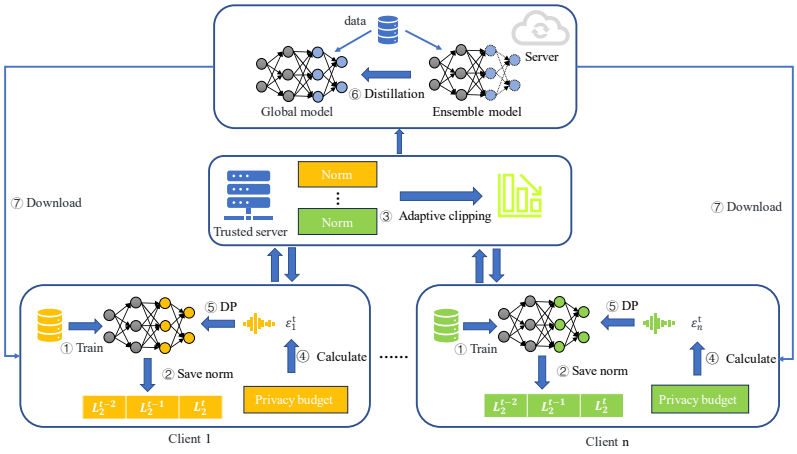


Fig. 1. ADP-FL framework

4.1 Algorithm Design

In the ADP-FL framework, we constructed a federated learning framework consisting of two servers and N clients, where the middle server is trusted, and the upper server is

untrusted. These clients are independent of each other, and each client has heterogeneous data, which has significant differences in distribution. In this framework, the ADP-FL algorithm is jointly completed through collaborative training between the server and multiple clients.

In the ADP-FL framework, the first step is to send the initialized global model W^0 to the clients. After receiving the global model, each client trains the model on their respective heterogeneous data (step 1). Once the local model finishes updating, we calculate the update Δ_i^t of the model parameters, as shown in Equation (7):

$$\Delta_i^t = W_i^t - W^{t-1} \tag{7}$$

Each client creates a sliding window S_w and adds the L2 norm of the update to S_w for subsequent calculations (step 2). Next, the update is processed for privacy protection. This begins with the step of clipping, where the selection of the clipping threshold is crucial. A too small threshold would restrict the range of the update, while a too large threshold would result in the addition of more noise. Therefore, we calculate the norm information of clients' historical updates and upload it to a trusted server. Based on the information from all clients, the clipping threshold is dynamically adjusted. Subsequently, the clipping threshold is sent to each client (step 3). Afterward, the clients calculate the privacy budget based on the clipping threshold and the norm information of the update (step 4). Finally, the update is added with noise (step 5), as shown in the following:

$$\widetilde{\Delta_i^t} = \frac{\Delta_i^t}{\max\left(1, \frac{\left\|\Delta_i^t\right\|_2}{C^t}\right)} + N\left(0, \sigma^2 {C^t}^2 I\right) \tag{8}$$

where $N(0, \sigma^2 {C^t}^2 I)$ is a random Gaussian noise vector, σ is the noise scale, $||\ ||_2$ is the L2 norm.

Noise is added to the update and then uploaded to the server, which receives the uploaded parameters and updates the ensemble model Ens. Then with the help of auxiliary data, knowledge distillation is performed between the global model and the integrated model (step 6) to enhance the global model, and finally, the global model is sent to the client.

The specific process is shown in Algorithm 1. We first initialize the parameters of the global model and send it to the clients (lines 1-2). After the client receives the global model, the model is trained using the local dataset (lines 8-12). After the local model finishes updating, we calculate the update (line 13) and then compute the clipping threshold based on the norm of the update (lines 14-15), the detailed steps will be presented in Algorithm 2. After the clipping threshold is calculated, we use Algorithm 3 to calculate the privacy budget based on the clipping threshold and norms of each client (line 16). Then, the updates are clipped and noised (lines 17-18) and uploaded to the server. The server receives the uploaded parameters updates the ensemble model Ens (lines 22-23), and then utilizes the knowledge distillation technique to update the global model (lines 24-25). Finally, the global model is sent back to the client (line 26) and the federated training process continues.

Algorithm 1. Federated Learning Based on Adaptive Differential Privacy (ADP-FL)

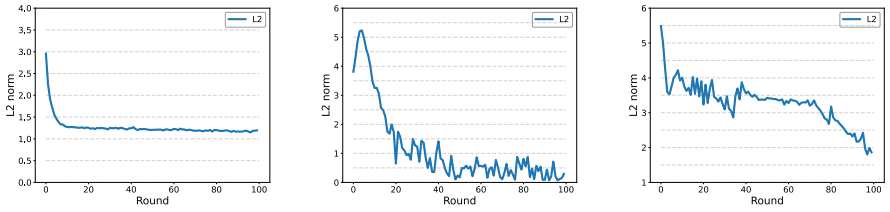
Input: local datasets D_i, number of clients N, rounds of communication T, number of local epochs E, learning rate η, clipping threshold C^0, personalization layer L, global model G

1: Initialize all model with random weights
2: Send W^0, C^0 to each client
3: **for** t =1,2,…,T **do**
4: // **Client Update**
5: **for** i =1,2,…N **do**
6: Receive W^t, C^t to each client
7: $W_i^t \leftarrow (W_b^t, W_L^{t-1})$
8: **for** e =1, 2,…,E **do**
9: **for** batch b ∈ B **do**
10: $W_i^t \leftarrow W_i^t - \eta \nabla F(W_i^t, b)$
11: **end for**
12: **end for**
13: $\Delta_i^t \leftarrow W_i^t$ - W^{t-1}
14: s ←**Client** (Δ_i^t, C^{t-1})
15: C^t ←**Trusted server** (s)
16: ε_i^t ←**Algorithm 3** ($\left\|\Delta_i^t\right\|_2$, C^t, ε^t)
17: $\Delta_i^t \leftarrow \Delta_i^t / \max\left(1, \frac{\|\Delta_i^t\|_2}{C^t}\right)$
18: $\widetilde{\Delta_i^t} \leftarrow \Delta_i^t + N(0,\ \sigma^2 {C^t}^2 I)$
19: **end for**
20: Send $\widetilde{\Delta_i^t}$, to server
21: // **Server Update**
22: Receive $\widetilde{\Delta_i^t}$ from client
23: $Ens^t \leftarrow$ update $(W^{t-1}, \frac{1}{N}\sum_{i=1}^{N} \widetilde{\Delta_i^t})$
24: $L_d \leftarrow L(G(x, W^{t-1}), Ens(x, Ens^t))$
25: $W^t \leftarrow W^{t-1} - \eta \nabla L_d$
26: Send W_b^t to each client
27: end for
Return: Model parameters W_i for each client

4.2 Adaptive Clipping

In machine learning, clipping techniques can be broadly classified into two categories: value-based clipping and norm-based clipping. Value-based clipping performs threshold clipping on the values in a single update, while norm-based clipping focuses on the overall norm of the update, and clipping is performed if the overall norm exceeds the clipping threshold. Compared to value-based clipping, the norm-based approach effectively preserves more local update information by scaling the overall update.

Setting a clipping threshold that is too large protects local updates better and the clipping operation produces less update bias. However, a larger threshold leads to the addition of excessive noise, reducing the utility of the final aggregated global model. Setting too small a clipping threshold, while adding less noise, can cause excessive clipping of the users' local updates. This leads to a loss of local update information and impacts the usability of the final global model.



(a) MNIST (CNN) (b) CIFAR-10 (ResNet-18) (c) CIFAR-10 (ResNet-34)

Fig. 2. L2 norm

From the observation of Fig. 2, we can see the following: Fig. 2(a) illustrates the change in parameter update norms of a CNN model on the MNIST dataset. It is evident that as the number of training rounds increases, the norm of the update diminishes gradually and eventually stabilizes, displaying a smoothed trend. In contrast, Fig. 2(b) and 2(c) depict the norm variation trends of ResNet-18 and ResNet-34 models respectively on the CIFAR-10 dataset. Unlike the smooth curve of the CNN model, these two models exhibit significant fluctuations despite an overall decreasing trend. This fluctuation reflects the impact of data heterogeneity, that is, the variations in the norm of updates between different clients. The displayed curve represents the changes in the average norm of updates across clients. Therefore, when selecting a clipping threshold, our strategy is to find a threshold that best aligns with the local update trends of all clients to the maximum extent possible.

First, each client creates a sliding window S_w locally to store the norm values of each round. When the sliding window reaches its maximum capacity, we calculate the average and standard deviation of its norms. Assuming the set of clients S_w is $\{\|\Delta_i^1\|_2, \|\Delta_i^2\|_2, \ldots, \|\Delta_i^t\|_2\}$, where t represents the number of norms. The mean of S_w is denoted as μ, and standard deviation s can be calculated using the following equation:

$$s = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\left\|\Delta_i^t\right\|_2 - \mu\right)^2} \tag{9}$$

A larger standard deviation implies significant fluctuations in parameter updates during the model training process, indicating that the parameter values change substantially with each update. Conversely, if the fluctuations in parameter updates are small, it suggests that the parameter values remain relatively stable across different training iterations. Therefore, the standard deviation reflects the range of adjustments that clients desire, while the mean reflects how clients want to adjust. The equation for calculating the desired change value from clients is as follows:

$$s = \begin{cases} s, & \mu < C^{t-1} \\ -s, & \mu \geq C^{t-1} \end{cases} \tag{10}$$

At this point, the calculated s is the change value expected by the current client. Then, all clients' s values are uploaded to a trusted server to calculate the mean s_{avg}. The clipping threshold is updated using the following Equation (11):

$$C^t = C^{t-1} - s_{avg} \tag{11}$$

Algorithm 2. Adaptive Clipping Strategy

Input: update Δ_i^t, standard deviation s, sliding window s_w
Output: clipping threshold C^t
function Client (Δ_i^t,C^{t-1})
  for i =1,2,…,n **do**
    $s_{w,i}$ ← add ($\left\|\Delta_i^t\right\|_2$)
    if s_w = full :
      μ ← Calculate the mean using s_w
      s ← Calculate the standard deviation using s_w
      if $\mu < C^{t-1}$:
        $s = s$
      else
        $s = -s$
  end for
return s
function Trusted server (s)
  for i =1,2,…,n **do**
    sum ←cumulative sum of s
  end for
  s_{avg} ←sum / n
  $C^t = C^{t-1} - s_{avg}$
return: C^t

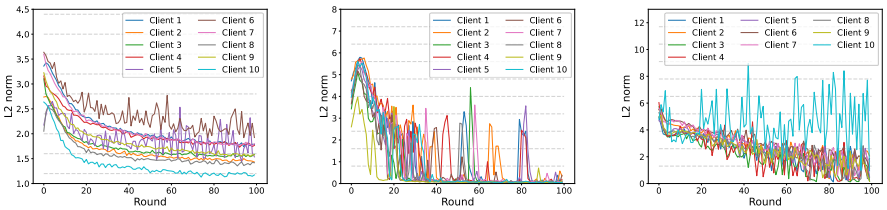
The specific process is shown in Algorithm 2. Each client first adds the L2 norm of the update to a sliding window (line 3). When the sliding window reaches its full capacity, the mean and standard deviation of the norm are calculated (lines 5-6). Meanwhile, the standard deviation is determined based on a comparison between the mean and the clipping threshold (lines 7-10). The resulting standard deviations are sent to a trusted server, which computes the sum and takes the average as the desired change value (lines 14-17). Finally, the clipping threshold is calculated using the Equation (11) (line 18).

4.3 Privacy Budget Allocation

Based on Algorithm 2, we have found a clipping threshold that matches the local update trend of all clients as closely as possible. However, in scenarios with heterogeneous data, some individual clients may still be significantly affected.

In Fig. 3, Fig. 3(a) shows the trend of L2 norm changes in the update across different clients for a CNN model on the MNIST dataset. It is observed that although there are some differences in norms among clients, the overall trend is consistently downward, ultimately stabilizing and becoming smooth. This indicates that on small models and simple datasets, the learning processes of different clients tend to be more consistent.

In contrast, Fig. 3(b) and 3(c) illustrate the norm trends of ResNet-18 and ResNet-34 on the CIFAR-10 dataset, respectively. For ResNet-18, the norm of client updates showed a significant increase in the early stages of training and then began to decrease. However, there were fluctuations during the decline process, and even after stabilizing, the norm of some clients still showed significant fluctuations. In the case of ResNet-34, the trend is different, the norm of client update does not gradually become smoother but resembles a gradually decreasing straight line. Moreover, we can observe that the norm change of one client is opposite to that of other clients, not only does it not show a gradually decreasing trend, but it also fluctuates greatly.



(a) MNIST (CNN) (b) CIFAR-10 (ResNet-18) (c) CIFAR-10 (ResNet-34)

Fig. 3. The L2 norm of each client

Algorithm 3. Privacy Budget Allocation Strategy

Input: privacy budget ε^t, clipping threshold C^t, update norm $\left\|\Delta_i^t\right\|_2$

Output: privacy budget ε_i^t

1: for i =1,2,…,n **do**

2: **if** $C^t > \left\|\Delta_i^t\right\|_2$:

3: $\varepsilon_i^t = \varepsilon^t + \frac{C^t - \left\|\Delta_i^t\right\|_2}{C^t - min} \times (1-\alpha)\varepsilon^t$

4: **else:**

5: $\varepsilon_i^t = a\varepsilon^t$

6: end for

7: return: ε_i^t

Therefore, this paper solves this problem by allocating privacy budgets, the specific calculation process is as follows. Given the clipping threshold provided by the trusted server and the norm of parameter updates for the current iteration, we can calculate the privacy budget of the client. The formula is as follows:

$$\varepsilon_i^t = \begin{cases} a\varepsilon^t + \dfrac{C^t - \|\Delta_i^t\|_2}{C^t - min} \times (1-\alpha)\varepsilon^t, & C^t > \|\Delta_i^t\|_2 \\ a\varepsilon^t \quad , & C^t \le \|\Delta_i^t\|_2 \end{cases} \tag{12}$$

where, C^t represents the clipping threshold provided by the trusted server, min represents the minimum value in $\{\|\Delta_1^t\|_2, \|\Delta_2^t\|_2, \ldots, \|\Delta_n^t\|_2\}$. ε^t is the privacy budget for the current round, a is a hyperparameter used to limit the privacy budget and prevent adjustments from being too small, ε_i^t denotes the privacy budget for client i in round t.

The specific process is shown in Algorithm 3. For each client, their clipping thresholds and norms are first compared and then substituted into the calculation of the privacy budget (lines 2-5).

4.4 Privacy Analysis

In this section, we analyze the privacy guarantees of the proposed ADP-FL algorithm.

Theorem 1 [28]: If Δ_i is clipped by C, the sensitivity of M is bounded as $S < 2C/n$.

Theorem 2: Let S be the sensitivity, when $\sigma \geq S\sqrt{2\log(1/\delta)}\,/\varepsilon$, for any $\delta \in (0,1)$ and $\varepsilon \leq 2\log(1/\delta)$, the output of Algorithm 1 satisfies $(\varepsilon, \delta) - DP$.

Proof. Suppose two neighboring datasets, D and D', differ by one user's sample. By theorem 1 the sensitivity S has

$$S = \|M(1), -M(D')\| \leq 2C^t/n \tag{13}$$

According to the Gaussian mechanism of RDP (Proposition 1) and sequence composition theorem (Proposition 3), for $\forall\delta \in (0,1)$ we have

$$D_\alpha\big(N(0,\sigma^2)||N(S,\sigma^2)\big) = \alpha S^2/(2\sigma^2) \tag{14}$$

It follows from $\sigma = S\sigma_s$ [28] that the client's upload process satisfies $(\alpha, \alpha/2{\sigma_s}^2)$-RDP, i.e., $(\alpha, \varepsilon) - RDP$.

According to the Postprocessing mechanism (Proposition 5), when Algorithm 1 processes the output of the differential privacy mechanism, the output still satisfies differential privacy as long as there is no new interaction with the private dataset. Thus, multiple local models still satisfy differential privacy after server aggregation.

In Algorithm 3, a privacy budget allocation strategy is employed, which allocates different privacy budgets to different clients. According to parallel combinatoriality (Proposition 4), the global federated learning model satisfies $(\alpha, \varepsilon) - RDP$, where $\varepsilon = \max(\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k)$.

Now we have proved that Algorithm 1 satisfies $(\alpha, \alpha/2{\sigma_s}^2) - RDP$. in order to make it equivalent to $(\varepsilon, \delta) - DP$, according to the theorem on the conversion of RDP to $(\varepsilon, \delta) - DP$ (Proposition 2), we need the following equation:

$$\frac{\alpha}{2{\sigma_s}^2} + \frac{\log(1/\delta)}{\alpha - 1} \leq \varepsilon \tag{15}$$

Choose $\alpha = 1 + \frac{2\log(1/\delta)}{\varepsilon}$ and rearrange Equation (15) we need the following equation:

$${\sigma_s}^2 \geq \frac{\varepsilon + 2\log(1/\delta)}{\varepsilon^2} \tag{16}$$

Based on the constraints on ε, the proof is derived.

5 Experiments

In this section, we will evaluate the proposed algorithm ADP-FL. To compare with federated learning algorithms such as FedPer, FedAvg, and LG-Fed, we will evaluate their model accuracy on the CIFAR-10 and MNIST datasets. In addition, we will also compare other adaptive clipping strategies.

5.1 Datasets and Models

Dataset. The CIFAR-10 dataset consists of 60,000 32×32 color images, with 6,000 images in each of the 10 different categories. Similarly, the CIFAR-100 dataset contains 600 images per category from 100 different classes, with 500 images used for training and 100 images used for testing. The MNIST dataset comprises 60,000 training images of handwritten digits and 10,000 test images, each with a size of 28×28 pixels and in grayscale.

Models. The basic structure of ResNet is composed of multiple residual blocks with convolutional layers. ResNet has different network configurations with various numbers of layers, such as 18, 34, 50, and 101. In this paper, we utilized ResNet-18 and ResNet-34 models, as well as the Convolutional Neural Network (CNN).

5.2 Implementation Details

Parameter Settings. In the experiment, the local epoch is set to 4, with a total of 100 communication rounds and 10 client participants. The local batch size is set to 128, and the learning rate is set to 0.01. The distillation epoch is set to 4, and the distillation batch size is set to 128, the hyperparameter a is 0.8. After constructing the local dataset for each user, we randomly selected 60% of the local dataset as the local training dataset. In the experiments with ResNet, we consider the last fully connected layer and basic blocks as personalized layers, and the number of personalized layers is set to 2. When evaluating the accuracy, the final accuracy performance is the average accuracy of local models on local test datasets for each client. The comparative federated learning methods include FedAvg [1], FedProx [22], FedPer [23], LD-Fed [24], and FedVF [25].

Dataset Partition. We use the Dirichlet distribution [26] to model the phenomenon of label distribution bias among clients. Experiments on the MNIST, CIFAR-10, and CIFAR-100 datasets utilize Dirichlet distribution Dir (α) to sample the number of samples per category for each client, where α is used to adjust the degree of class imbalance in local data, with smaller values of α indicating more severe class imbalances. When the dataset is partitioned using the Dirichlet distribution, α is set to 0.1, 0.5, and 1.0.

5.3 Experimental Results and Analysis

The algorithm proposed in this paper is called ADP-FL, and the part without noise is referred to as PFL. Table 2 presents a comparison of the accuracy between PFL and other methods. The bold numbers indicate the highest accuracy achieved.

Table 2 provides a detailed display of the model accuracy for different models on different datasets with varying degrees of heterogeneity. On the CIFAR-100 dataset, the paper first compares the accuracy of ResNet-18 and ResNet-34. When α=1, PFL achieves a higher accuracy rate than FedVF by approximately 3% for ResNet-18, and

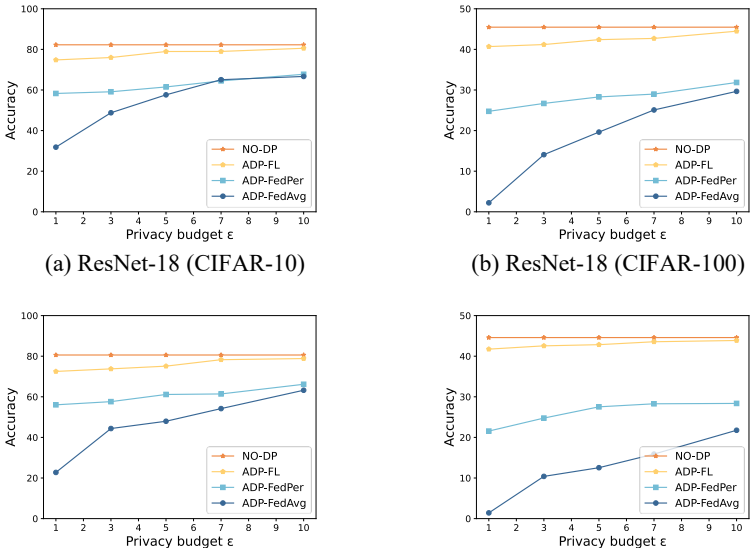
approximately 1% for ResNet-34. As the heterogeneity degree gradually increases, the personalized models show increasing accuracy, while FedAvg and Fedprox exhibit a decreasing trend. On the CIFAR-10 dataset, the paper compares the accuracy of three models. Among the residual networks, PFL achieves the highest accuracy, while for the CNN network, FedVF has the highest accuracy when α=1. Among the CNN models on the MNIST dataset, FedVF and FedPer achieve the highest accuracy. Although PFL does not have the same high accuracy as these two algorithms, its accuracy still reaches over 98%, with a difference of only about 1%.

Table 2. Test the average accuracy of different models.

Dataset	Model	α	FedAvg	FedProx	FedPer	LG-Fed	FedVF	PFL
CIFAR-100	ResNet-18	0.1	33.72	34.11	54.66	51.38	58.26	**63.12**
		0.5	36.74	35.66	38.78	33.59	44.74	**45.47**
		1	36.77	37.39	32.99	27.10	38.87	**41.44**
	ResNet-34	0.1	34.68	33.92	54.85	50.40	59.14	**61.15**
		0.5	36.50	36.92	41.45	31.45	42.43	**44.59**
		1	37.99	37.66	35.63	26.22	40.33	**41.00**
CIFAR-10	ResNet-18	0.1	60.32	66.30	87.20	85.41	89.05	**91.15**
		0.5	71.11	68.81	73.98	67.61	80.85	**82.26**
		1	70.87	70.69	66.71	61.38	73.37	**80.16**
	ResNet-34	0.1	49.35	61.67	86.66	83.40	90.43	**90.54**
		0.5	72.66	72.66	75.41	65.33	77.47	**80.59**
		1	73.13	73.13	74.41	56.75	75.58	**77.54**
	CNN	0.1	55.70	53.15	80.34	82.88	78.22	**83.47**
		0.5	55.46	54.71	64.90	62.77	65.80	**65.85**
		1	57.32	56.32	59.21	59.71	**61.34**	58.76
MNIST	CNN	0.1	98.32	98.50	99.54	99.46	**99.61**	98.80
		0.5	98.77	98.86	**99.25**	98.69	99.16	98.90
		1	98.77	98.79	**98.99**	98.46	98.97	98.26

Fig. 4 compares the model accuracy under different privacy budgets (ε = 1, 3, 5, 7, 10). Fig. 4(a) and 4(c) display the accuracy of ResNet-18 and ResNet-34 on the CIFAR-10 dataset, respectively. It can be observed that as the privacy budget increases, the accuracy of all algorithms gradually improves. This is because a higher privacy budget leads to less noise being added, which in turn improves model accuracy. Similarly, Fig. 4(b) and 4(d) demonstrate that on the CIFAR-100 dataset, the accuracy of ResNet-18 and ResNet-34 also increases with higher privacy budgets. Moreover, the ADP-FL algorithm consistently outperforms the other two algorithms in all experiments.

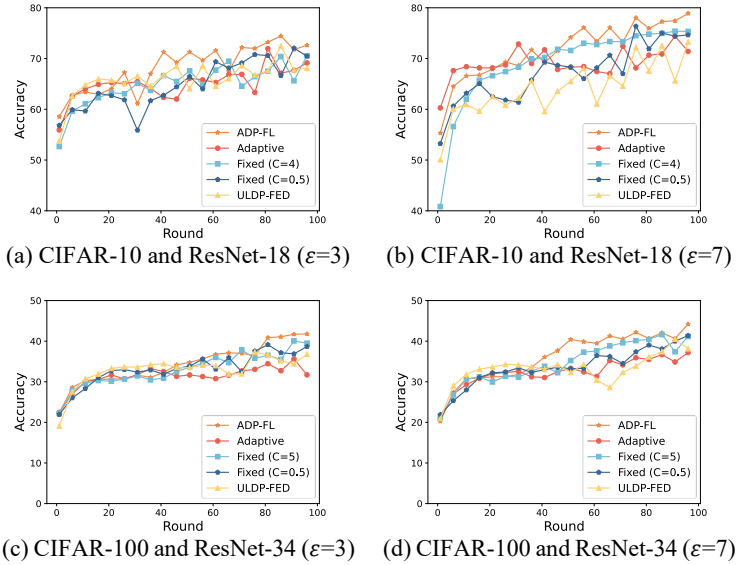
Fig. 5 presents a comparison of accuracy among different clipping strategies. The orange starred line represents the clipping strategy proposed in this paper. The light blue square line and the dark blue pentagon line represent fixed clipping thresholds. The red circular line represents the adaptive clipping strategy [15], and the yellow triangular line represents the gradually decaying strategy [27]. Fig. 5(a) shows that in



(a) ResNet-18 (CIFAR-10) (b) ResNet-18 (CIFAR-100)

(c) ResNet-34 (CIFAR-10) (d) ResNet-34 (CIFAR-100)

Fig. 4. Accuracy of models with different privacy budgets



(a) CIFAR-10 and ResNet-18 (ε=3) (b) CIFAR-10 and ResNet-18 (ε=7)

(c) CIFAR-100 and ResNet-34 (ε=3) (d) CIFAR-100 and ResNet-34 (ε=7)

Fig. 5. Accuracy of different clipping strategies

the CIFAR-10 dataset, when the privacy budget is 3, the clipping strategy proposed in this paper improves by about 3% compared to the fixed strategy (C = 4), by about 4%

compared to the adaptive strategy, and by about 2% compared to the decay strategy. In Fig. 5(b), the proposed clipping strategy outperforms the adaptive strategy by approximately 0.6%, the fixed strategy (C = 4) by approximately 1%, and the decaying strategy by approximately 3%. In Fig. 5(c) and 5(d), which show the results on the CIFAR-100 dataset using a ResNet-34 model, the ADP-FL strategy still exhibits the best performance.

Table 3. The impact of privacy budget on adaptive clipping.

Dataset (Model)	α	ε	Adaptive C	Adaptive C (Fixed epsilon)
CIFAR-10 (ResNet-18)	0.5	3	**75.99**	75.19
		5	**78.95**	76.08
		7	**78.99**	76.28
	0.1	3	**87.11**	84.05
		5	**88.43**	87.94
		7	**91.82**	91.11
CIFAR-100 (ResNet-34)	0.5	3	**42.55**	41.92
		5	**42.84**	42.09
		7	**43.56**	43.36
	0.1	3	**56.13**	52.45
		5	**58.66**	58.11
		7	**60.25**	58.62

Table 3 presents the impact of the privacy budget allocation module in the adaptive clipping strategy. On the CIFAR-10 dataset, when the parameter $\alpha = 0.5$ and the privacy budget is set to 3, compared to a fixed privacy budget, the accuracy improves by 0.8%. For privacy budgets of 5 and 7, the accuracy improvement is close to 2%. When $\alpha = 0.1$, indicating a higher degree of data heterogeneity, with a privacy budget of 3, the accuracy increases by 3% compared to the fixed privacy budget. However, when the privacy budget increases to 5 and 7, the accuracy improvement is less than 1%. In the CIFAR-10 dataset, the improvement is more pronounced when $\alpha = 0.1$, with the highest accuracy improvement being around 3.6%. Overall, the privacy budget allocation enhances the effectiveness of the adaptive clipping strategy, thereby playing a significant role in improving model accuracy.

6 Conclusion

This paper proposes a federated learning based on adaptive differential privacy. By analyzing the dynamic change in the L2 norm of the update from each client, the algorithm is able to adjust the clipping threshold for each round, to balance the requirements of all clients. Additionally, based on the adaptive clipping strategy, a privacy budget allocation strategy is proposed to address the issue of imbalanced privacy protection among clients caused by a unified clipping threshold. With this approach, we ensure the privacy of each user while optimizing the overall performance of the model.

References

1. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. pp. 1273–1282. PMLR (2017)
2. Wu, J., Si, S., Wang, J., Xiao, J.: Threats and defenses of federated learning: a survey. Big Data Research 8(5), 12 (2022)
3. Nasr, M., Shokri, R., Houmansadr, A.: Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In: 2019 IEEE symposium on security and privacy (SP). pp. 739–753. IEEE (2019)
4. Fredrikson, M., Jha, S., Ristenpart, T.: Model inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACMSIGSAC conference on computer and communications security. pp. 1322–1333(2015)
5. Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q., Qi, H.: Beyond inferring class representatives: User-level privacy leakage from federated learning. In: IEEE INFOCOM 2019-IEEE conference on computer communications. pp. 2512–2520. IEEE (2019)
6. Yan, X., Cui, B., Xu, Y., Shi, P., Wang, Z.: A method of information protection for collaborative deep learning under gan model attack. IEEE/ACM Transactions on Computational Biology and Bioinformatics 18(3), 871–881 (2019)
7. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. Advances in neural information processing systems 32 (2019)
8. Zhang, C., Li, S., Xia, J., Wang, W., Yan, F., Liu, Y.: {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In: 2020 USENIX annual technical conference (USENIX ATC 20). pp. 493–506 (2020)
9. Hao, M., Li, H., Luo, X., Xu, G., Yang, H., Liu, S.: Efficient and privacy-enhanced federated learning for industrial artificial intelligence. IEEE Transactions on Industrial Informatics 16(10), 6532–6542 (2019)
10. Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L.: Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. pp. 308–318(2016)
11. Dwork, C.: Differential privacy. In: International colloquium on automata, languages, and programming. pp. 1–12. Springer (2006)
12. McMahan, H.B., Ramage, D., Talwar, K., Zhang, L.: Learning differentially private recurrent language models. arXiv preprint arXiv:1710.06963 (2017)
13. Zhang, X., Chen, X., Hong, M., Wu, Z.S., Yi, J.: Understanding clipping for federated learning: Convergence and client-level differential privacy. In: International Conference on Machine Learning, ICML 2022 (2022)
14. Geyer, R.C., Klein, T., Nabi, M.: Differentially private federated learning: A client level perspective. arXiv preprint arXiv:1712.07557 (2017)
15. Andrew, G., Thakkar, O., McMahan, B., Ramaswamy, S.: Differentially private learning with adaptive clipping. Advances in Neural Information Processing Systems 34, 17455–17466 (2021)
16. Noble, M., Bellet, A., Dieuleveut, A.: Differentially private federated learning on heterogeneous data. In: International Conference on Artificial Intelligence and Statistics. pp. 10110–10145. PMLR (2022)
17. Xu, Z., Shi, S., Liu, A.X., Zhao, J., Chen, L.: An adaptive and fast convergent approach to differentially private deep learning. In: IEEE INFOCOM 2020-IEEE Conference on Computer Communications. pp. 1867–1876. IEEE (2020)

18. Wang, N., Xiao, Y., Chen, Y., Zhang, N., Lou, W., Hou, Y.T.: Squeezing more utility via adaptive clipping on differentially private gradients in federated meta-learning. In: Proceedings of the 38th Annual Computer Security Applications Conference. pp. 647–657 (2022)
19. Mironov, I.: Rényi differential privacy. In: 2017 IEEE 30th computer security foundations symposium (CSF). pp. 263–275. IEEE (2017)
20. Zhang, P.: Research on Rényi Differential Privacy Protection Algorithm in Deep Learning. Master's thesis, Dalian Maritime University (2022) (in Chinese)
21. Dwork, C., Roth, A., et al.: The algorithmic foundations of differential privacy. Foundations and Trends® in Theoretical Computer Science 9(3–4), 211–407 (2014)
22. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems 2, 429–450 (2020)
23. Arivazhagan, M.G., Aggarwal, V., Singh, A.K., Choudhary, S.: Federated learning with personalization layers. arXiv preprint arXiv:1912.00818 (2019)
24. Liang, P.P., Liu, T., Ziyin, L., Allen, N.B., Auerbach, R.P., Brent, D., Salakhutdinov, R., Morency, L.P.: Think locally, act globally: Federated learning with local and global representations. arXiv preprint arXiv:2001.01523 (2020)
25. Mei, Y., Guo, B., Xiao, D., Wu, W.: Fedvf: Personalized federated learning based on layer-wise parameter updates with variable frequency. In: 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC). pp. 1–9. IEEE (2021)
26. Hsu, T., Qi, H., Brown, M.: Measuring the effects of non-identical data distribution for federated visual classification. arxiv 2019. arXiv preprint arXiv:1909.06335
27. Zhang, X., Zhang, L., Zhang, Z.: Federated learning method based on user-level localized differential privacy. Computer Research and Development pp. 1–16 (in Chinese)
28. Pan, Y., Ni, J., Su, Z.: Fl-pate: Differentially private federated learning with knowledge transfer. In: 2021 IEEE Global Communications Conference (GLOBECOM). pp. 1–6. IEEE (2021)