applied sciences

Article

Local Differential Privacy-Based Federated Learning under Personalized Settings

Xia Wu [1], Lei Xu [2,*] and Liehuang Zhu [2]

[1] School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China
[2] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
* Correspondence: xu.lei@bit.edu.cn

Abstract: Federated learning is a distributed machine learning paradigm, which utilizes multiple clients' data to train a model. Although federated learning does not require clients to disclose their original data, studies have shown that attackers can infer clients' privacy by analyzing the local models shared by clients. Local differential privacy (LDP) can help to solve the above privacy issue. However, most of the existing federated learning studies based on LDP, rarely consider the diverse privacy requirements of clients. In this paper, we propose an LDP-based federated learning framework, that can meet the personalized privacy requirements of clients. We consider both independent identically distributed (IID) datasets and non-independent identically distributed (non-IID) datasets, and design model perturbation methods, respectively. Moreover, we propose two model aggregation methods, namely *weighted average method* and *probability-based selection method*. The main idea, is to weaken the impact of those privacy-conscious clients, who choose relatively small privacy budgets, on the federated model. Experiments on three commonly used datasets, namely MNIST, Fashion-MNIST, and forest cover-types, show that the proposed aggregation methods perform better than the classic arithmetic average method, in the personalized privacy preserving scenario.

Keywords: federated learning; local differential privacy; personalized privacy preserving; model perturbation; model aggregation

check for updates

Citation: Wu, X.; Xu, L.; Zhu, L. Local Differential Privacy-Based Federated Learning under Personalized Settings. *Appl. Sci.* **2023**, *13*, 4168. https://doi.org/10.3390/app13074168

Academic Editor: Gianluca Lax

Received: 21 February 2023
Revised: 17 March 2023
Accepted: 22 March 2023
Published: 24 March 2023

1. Introduction

The explosive development of big data has brought new challenges to industry, and the problem of data silos is one of them. Companies in different industries have different businesses and systems that are not open to the public, making it difficult to communicate and integrate data and information [1]. Joint modeling needs to overcome many barriers. Currently, many people have transferred their data to cloud platforms [2]. This data may contain sensitive information from individuals, enterprises, and even governments. Solving the data privacy problem in cloud storage is another challenge for big data. Countries around the world have proposed laws to protect data privacy, such as the General Data Protection Regulation (GDPR) [3] implemented by the European Union in 2018. The establishment of these regulations is undoubtedly a measure to improve the level of social civilization, but it also brings new challenges to traditional data processing. Facing the problems of data silos and data privacy, federated learning (FL) provides a feasible solution, which was first proposed by Google Research in 2016 [4]. In federated learning, the data analyst (also known as the server) does not collect the original data of clients (e.g., individuals, enterprises, or institutions), but only collects the local training results of the client, for training a federated model.

Although federated learning does not require clients to disclose their original data, studies have shown that attackers can infer client privacy by analyzing local models shared by clients [5,6]. To enhance privacy protection, researchers have applied secure multi-party computation (SMC) [7,8], homomorphic encryption (HE) [9,10], and differential privacy

(DP) [11] to federated learning. Secure multi-party computing and homomorphic encryption are encryption-based methods, that usually require the design of complex encryption computing protocols to hide the real input and output, where the communication cost and computation cost are high. SMC-based federated learning, generally utilizes the secret sharing strategy. The client needs to divide its local model to several parts and transmit these parts to all other clients, which incurs a high communication cost. In HE-based federated learning, clients send encrypted model parameters to the server, and the ciphertexts are of a larger size than the plaintexts. Differential privacy-based methods mainly hide some sensitive attributes of participants by adding noise to the data, until the attacker cannot distinguish individuals through differential attacks. In DP-based federated learning, the client sends perturbed model parameters to the server, and the communication cost is almost the same as that of ordinary federated learning. Compared with the first two methods, the differential privacy-based method has low computation and communication costs, which makes it more suitable for individual users to deploy.

According to the trust assumption, existing differential privacy mechanisms can be roughly divided into two categories: centralized differential privacy (CDP) and local differential privacy (LDP) [12,13]. In the federated learning framework based on centralized differential privacy [14], the client directly uploads the trained local model to the server, and a *trusted* server aggregates the collected models, and then randomly perturbs the aggregated models. Although such a framework can maintain high accuracy, it relies on the trusted server. The federated learning framework based on local differential privacy [15], assumes that the server is *untrusted*, and it does not care what background knowledge the server has. The client performs model perturbation before sending the local model to the server, to protect its privacy. However, different clients have different definitions of privacy and expect different levels of privacy protection. The local differential privacy mechanism mainly uses the privacy parameter called *privacy budget*, to measure the level of privacy protection. The clients can express different privacy requirements by setting different privacy budgets. A federated learning framework based on local differential privacy should be adapted to these “personalized” privacy settings.

Many of the existing federated learning methods, based on local differential privacy, do not take into account the personalized problem [16,17]. Because of the heterogeneity of data between different clients and the inconsistency of clients’ privacy requirements, it is difficult to meet the requirements of all participants with a single federated model, so a personalized method needs to be considered, to meet the personalized requirements of clients. In federated learning based on personalized local differential privacy, when a client sets a low privacy budget, a large noise will be introduced, and the accuracy of the federated model may be low if the server still treats all clients equally. Therefore, it is necessary for the server to take certain measures to adjust the weights of the clients’ local models.

Following the basic framework of LDP-based federated learning, in this paper we propose several aggregation methods to meet the personalized privacy requirements of clients. According to the characteristics of data distribution, federated learning can be divided into three categories, namely horizontal federated learning, vertical federated learning, and federated transfer learning [18]. Among them, horizontal federated learning is the most common one. The main feature of horizontal federated learning, is that training datasets have the same feature space but different sample spaces. We focus on horizontal federated learning in this paper. We consider both independent identically distributed (IID) datasets and non-independent identically distributed (non-IID) datasets. For IID datasets, we choose the Gaussian mechanism to realize locally differentially private model perturbation. Each client adds Gaussian random noise to the original local model. For non-IID datasets, we choose a probability-based "symbolized" function [19] to realize locally differentially private model perturbation.

The proposed methods are applied on the server side. Specifically, a weighted average method and a probability-based selection method are proposed for model aggregation.

The basic idea of the proposed methods, is to assign different weights to different clients, according to the privacy budgets set by clients. The difference between the two methods, is that in the weighted average method, every client's local model is utilized for model aggregation. If the client chooses a high privacy budget, which means that it does not need a high level of privacy protection, the client will not add too much noise to its local model parameters. Correspondingly, a high weight will be assigned to the client's local model. While in the probability-based selection method, the server randomly selects a few local models for model aggregation. If the client chooses a high privacy budget, a high probability will be assigned to the client's local model, and the client's local model is more likely to be selected by the server. We have conducted experiments on several commonly used datasets, and compared the results of the proposed methods with those of the traditional methods (i.e., the server treats all clients equally). The experimental results show that the weighted average method performs better on IID datasets, and the probability-based selection method performs well on both IID datasets and non-IID datasets.

Our contributions are summarized as follows:

1. Different model perturbation methods are proposed for IID and non-IID data. For IID data, the Gaussian mechanism is applied, to achieve local differential privacy preserving. For non-IID data, the symbolized function is used, to perturb the local model of the client.
2. In order to obtain a federated model under the premise of personalized privacy settings, two model aggregation methods are proposed, namely a weighted average method and a probability-based selection method. The proposed methods can help to reduce the impact of privacy-conscious clients on the federated model, thus can preserve the accuracy of the model.
3. Experiments are conducted on real-world datasets. The experimental results show that, compared with the arithmetic average aggregation method, the proposed aggregation methods can achieve better accuracy on both IID data and non-IID data.

The remainder of this paper is organized as follows. Section 2 introduces some core concepts about local differential privacy and federated learning used in our work. Section 3 describes the system model of the proposed method. The proposed model aggregation methods are presented in Section 4. Experiments and performance analysis of the proposed method are presented in Section 5. Finally, conclusions are drawn in Section 6.

2. Preliminaries

2.1. Local Differential Privacy

Local differential privacy considers the possibility of server stealing, or leaking client's privacy, during data collection. Under this framework, each client first conducts randomized perturbation on the data, and then sends the perturbed data to the server. The server makes statistics on the collected data, to obtain effective analysis results. A formal definition of local differential privacy [12] is given below:

Definition 1 (*Local Differential Privacy (LDP)*)**.** *For a randomized algorithm M, its domain of definition and range are $D(M)$ and $R(M)$, respectively. For any two records l and l' in $D(M)$, their same output of M is S, where $S \subseteq R(M)$. If the following inequality holds, then the randomized algorithm M satisfies (ϵ, δ)-LDP,*

$$Pr[M(l) = S] \leq e^{\epsilon} \cdot Pr[M(l') = S] + \delta. \tag{1}$$

In the above definition, $Pr[M(l) = S]$ represents the probability that the output of algorithm M is S, given the input l, and $Pr[M(l') = S]$ represents the probability that the output of algorithm M is S, given the input l'. The parameter ϵ is called the *privacy budget*, and the privacy protection level of the data can be adjusted through this parameter. Theoretically, the larger the privacy budget ϵ is, the lower the privacy protection level M, can be achieved, and the higher the utility of the output is. The parameter δ, represents the tolerance of some possibility of failure. The client uses a randomized algorithm satisfying LDP to perturb the local model parameters. The algorithm applied by the client is referred to as the perturbation mechanism.

To achieve LDP, a data perturbation mechanism needs to be introduced. At present, random response (RR) [20] is the mainstream perturbation mechanism of LDP. Its main idea is using the uncertainty of response to sensitive issues to protect the privacy of the original data. A Gaussian mechanism [21] is not a typical perturbation mechanism of LDP, but it is often used to realize LDP in federated learning [22]. The basic idea of the Gaussian mechanism is to blur the original data, by adding 0-mean Gaussian noise. The advantage of Gaussian noise is that the sum of two Gaussians is a Gaussian, which means the effects of the Gaussian mechanism on the statistical analysis may be easier to correct for [11].

2.2. Federated Learning

In a federated learning scenario, there are N clients with their own local data $\{D_1, D_2, \ldots, D_N\}$, and a server that wants the participants to jointly train an effective model. The initial model parameter of the server is denoted as $W^{(0)}$. Each client has their own model, whose structure is consistent with that of the server. The client will upload the model to the server every time it trains, which will lead to a high cost of network communication and low training efficiency. In order to reduce the number of communications, the client locally trains their model for multiple rounds, and then uploads the accumulated updates of model parameters to the server. This algorithm is called *FedAvg* [4], which is one of the most widely used federated learning algorithms. In the r-th round of training, the server first randomly selects m clients from N clients, and sends current model parameter vector $W^{(r)}$ to the selected m clients simultaneously. Then, each selected client i, utilizes the model parameter sent by the server and its local data, to train a new model. We denote the new model parameter as $G_i^{(r)}$. The local model update $\Delta_i^{(r)} \doteq G_i^{(r)} - W^{(r)}$ is sent to the server. After receiving the update from each selected client, the server computes the federated model as follows:

$$W^{(r+1)} = W^{(r)} + \frac{\sum_{i=1}^{m}(\Delta_i^{(r)})}{m}. \tag{2}$$

Then the server will send $W^{(r+1)}$ to all clients, and the $r+1$-th round of training begins. Generally, after several rounds of training, the federated model converges.

In the federated learning scenario, the dataset of each client may be IID or non-IID. The FedAvg algorithm can be directly applied to IID scenarios. While for non-IID scenarios, using FedAvg will lead to a decline in the accuracy of the federated model, because the data distributions of different clients are quite different. At present, how to train a federated model on non-IID data is still an open problem in federated learning and needs to be solved urgently. To solve the non-IID problem, Zhao et al. [23] proposed creating a data subset that can be shared on all clients. The shared data subset can narrow the difference between the data distribution of each client and the overall data distribution. The privacy-preserving federate learning framework proposed in this paper, is obtained by incorporating an LDP perturbation mechanism and proposed aggregate methods into federated learning, and better model accuracy is obtained by modifying the model aggregation method.

3. System Model

The notations used in this article are listed in Table 1.

Table 1. Notations.

Symbol	Description	Symbol	Description
$\mathbb{R}$	Real number field	M	Perturbation mechanism
f	Aggregation rule	N	Total number of clients
m	Number of selected clients by using probability parameter	i	The i-th client
R	Maximum number of training rounds	r	The r-th round ($r = 1, 2, \ldots, R$)
d	Dimension of model parameter	η	Learning rate of SGD
U	Maximum number of logistic regression rounds	q	Random sampling probability
ω	A uniformly distributed random number between 0 and 1, generated by the server	D_i	Training dataset of the i-th client
ϵ_i	Privacy budget of the i-th client	c_i	Clipping threshold of the i-th client
δ_i	Inverse of training dataset size (default: 10^{-5})	σ_i	Standard deviation of the i-th client
α_i	Weighting coefficient of the i-th client in the weighted average method	ρ_i	Inverse of the standard deviation of the Gaussian distribution of the i-th client
P_i	Probability parameter of the i-th client	$W^{(0)}$	Initial federated model parameter
$W^{(r)}$	Federated model parameter generated in the r-th round	$g_i^{(r)}$	Gradient of the i-th client in the r-th round
$\Delta_i^{(r)}$	Local model update of the i-th client in the r-th round	$G_i^{(r)}$	Local model parameter of the i-th client generated in the r-th round
$\widehat{G}_i^{(r)}$	Clipped local model parameter of the i-th client generated in the r-th round	$\widetilde{G}_i^{(r)}$	Perturbed local model parameter of the i-th client generated in the r-th round

In this paper, we consider a horizontal federated learning architecture. As shown in Figure 1, there are N clients in total. The server initializes the initial federated model parameter $W^{(0)}$, and sends $W^{(0)}$ to each client i. The parameter $W^{(0)}$ is a d-dimensional vector. Each element in the vector is 0. In the r-th round, each client i, trains the local model $G_i^{(r)}$, based on the federated model parameter of previous round $W^{(r-1)}$ and local training dataset D_i. Then, each client i, generates the clipped local model parameter $\widehat{G}_i^{(r)}$, by the model clipping process. Next, each client i, generates the perturbed local model parameter $\widetilde{G}_i^{(r)} = M(\widehat{G}_i^{(r)})$, by the model perturbation mechanism M, and sends $\widetilde{G}_i^{(r)}$ to the server. In the end of the r-th round, the server generates the federated model parameter of r-th round $W^{(r)} = f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)})$, by the model aggregation rule f, and sends $W^{(r)}$ to each client. Then the $r+1$-th round begins.

3.1. Client Side

There are N clients participating in a federated learning task. The local training data of clients are either IID or non-IID. In each round, each client will update its local model parameter $\widetilde{G}_i^{(r)}$, based on the federated model sent from the server. The client uses stochastic gradient descent (SGD) [24] to optimize the model's parameter, and obtain the local model parameter. Then, the client uses the perturbation mechanism M, to add some random noise or make some perturbation on the local model parameter. Clients can configure their individualized privacy budget as they want. Parameter ϵ_i, is the privacy budget of client i in the mechanism. We define that ϵ_i satisfies $\epsilon_{low} \leq \epsilon_i \leq \epsilon_{high}$, where ϵ_{low} is usually set to 1, and ϵ_{high} is usually set to 10 [25,26]. The privacy budget ϵ_i, of each client i, is known to the server. Next, we will describe the local training process in the r-th round, in detail.

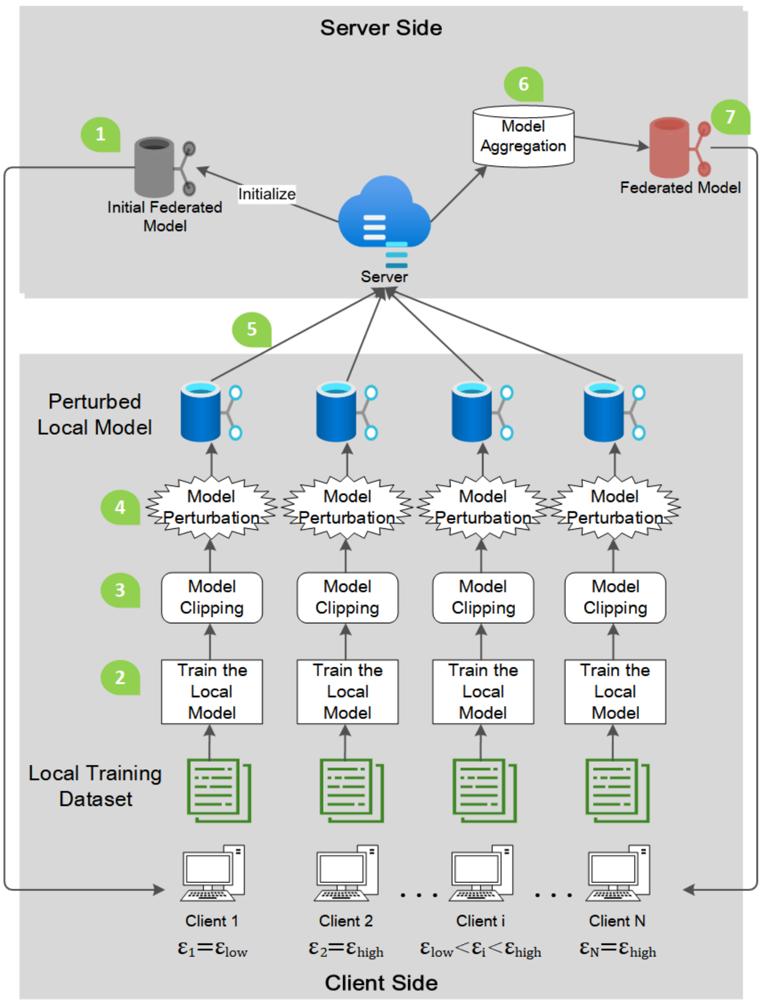

Figure 1. The overall architecture of the federated learning system with personalized local differential privacy.

Train the local model $G_i^{(r)}$. Given the federated model parameter $W^{(r-1)}$ from the previous round (i.e., the $r-1$-th round), each client applies the SGD algorithm on the local training dataset D_i, and obtains the local model parameter $G_i^{(r)}$:

$$G_i^{(r)} = W^{(r-1)} - \eta g_i^{(r)}, \tag{3}$$

where η denotes the learning rate of the SGD algorithm, and $g_i^{(r)}$ denotes the gradient of the i-th client in the r-th round. It should be noted that, when the clients' training data are IID, each client i, will sample from its own training dataset D_i, with the same sampling probability q, before training the local model, and then each client applies the SGD algorithm on the sampled dataset D_i'.

Model clipping. After using SGD to get $G_i^{(r)}$, each client needs to clip the local model parameter. Clipping [27] can prevent excessive noise from being introduced in the subsequent perturbation process, to a certain extent. Clipping can also make the calculation of sensitivity more convenient. To conduct model clipping, the client needs to pre-specify a clipping threshold. The value of the threshold depends on the training dataset. That is to say, different clients may use different clipping thresholds. Let c_i denote the threshold specified by client i. In the r-th round of the training, client i produces the clipped local model parameter $\widehat{G}_i^{(r)}$ as follows. For each element b, in $G_i^{(r)}$, if $b > c_i$, then the client replaces b with c_i, otherwise the client replaces b with $-c_i$.

Model perturbation. Given the clipped parameter $\widehat{G}_i^{(r)}$, the client i, applies the randomized mechanism M, to each element in $\widehat{G}_i^{(r)}$, and obtains a perturbed vector $\widetilde{G}_i^{(r)} \doteq M(\widehat{G}_i^{(r)})$. Depending on the distribution of clients' training data, different perturbation mechanisms are applied.

1. For IID data, the perturbation mechanism M, is to add Gaussian noise to each element p, in $\widehat{G}_i^{(r)}$. Firstly, the client i, calculates the standard deviation σ_i [28], of the Gaussian distribution:
$$\sigma_i = \sqrt{\frac{4q^2R}{1-q}(\frac{2}{\epsilon_i^2}log\frac{1}{\delta_i} + \frac{1}{\epsilon_i})}, \tag{4}$$
where ϵ_i denotes the privacy budget set by the client i, q denotes the probability of randomly sampling the samples in clients i's dataset, and D_i, $\delta_i \doteq \frac{1}{|D_i|}$ denotes the inverse of the dataset size. Secondly, random noise is generated according to the Gaussian distribution $\mathcal{N}(0, \sigma_i^2)$. The probability density function is given by
$$Gauss(x) = \frac{1}{\sqrt{2\pi}\sigma_i} exp(\frac{x^2}{2\sigma_i^2}). \tag{5}$$
For each element p, in $\widehat{G}_i^{(r)}$, the perturbed result $\widetilde{p}$, is given by $p + \mathcal{N}(0, \sigma_i^2)$. Finally, the client i sends $\widetilde{G}_i^{(r)}$ to the server.
2. For non-IID data, the perturbation mechanism M, is to symbolize $\widehat{G}_i^{(r)}$. By symbolize, we mean that only the sign of each element in $\widehat{G}_i^{(r)}$ is retained, making $\widehat{G}_i^{(r)}$ a vector consisting of only $+1$ or -1. As described in [19], each client i, uses the privacy-symbolization function $ldpsign(\cdot)$ to perturb $\widehat{G}_i^{(r)}$, and obtain $\widetilde{G}_i^{(r)}$. Specifically, for each element p, in the vector $\widehat{G}_i^{(r)}$, the perturbed result $\widetilde{p}$, is given by
$$\widetilde{p} = ldpsign(p) = \begin{cases} 1, & \text{with probability } \varphi(\frac{p}{\sigma_i}), \\ -1, & \text{with probability } 1 - \varphi(\frac{p}{\sigma_i}). \end{cases} \tag{6}$$
In the above equation, $\varphi(\cdot)$ denotes the cumulative distribution function of the normalized Gaussian distribution, and σ_i denotes the standard deviation. The standard deviation is computed as:
$$\sigma_i = \frac{\Delta_2}{\epsilon_i}\sqrt{(2ln(1.25/\delta_i))}, \tag{7}$$
where Δ_2 denotes the l_2-sensitivity [11], and $\delta_i \doteq \frac{1}{|D_i|}$ denotes the inverse of the dataset size.

3.2. Server Side

In the proposed federated learning system, the server is an untrusted aggregator, which is responsible for collecting the local model of each client and generating the federated model. First, the server initializes the federated model parameter $W^{(0)}$ and sends $W^{(0)}$ to each client. In the r-th round, the server will collect clients' parameters for federated model optimization. In particular, the server aggregates a set of perturbed parameters of clients' local models and averages the parameters at the end of each round. The new model parameter, after the server's averaging operation, is called the federated model parameter. The server intends to maximize the accuracy of both the federated and local model, while preserving clients' privacy. The federated model parameter is sent back to clients to update their local model parameters. Next, we will describe the model aggregation process in the r-th round, in detail.

Model aggregation. In the r-th round, the server collects the perturbed local parameter $\widetilde{G}_i^{(r)}$, of N clients and performs an aggregation operation, to obtain $W^{(r)}$. Let f denote the model aggregation rule. The following equation is the aggregation process of independent identically distributed data:

$$W^{(r)} = f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)}). \tag{8}$$

In particular, when the training data are non-independent identically distributed, the server needs to symbolize the model parameter after applying the aggregation rule. For each element p, in $f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)})$, the symbolized result $\widetilde{p}$, is given by [19]

$$\widetilde{p} = sign(p) = \begin{cases} -1, & \text{if } p < 0, \\ 0, & \text{if } p = 0, \\ 1, & \text{if } p > 0. \end{cases} \tag{9}$$

We use the following equation to represent the aggregation and symbolization process of non-independent identically distributed data:

$$W^{(r)} = sign(f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)})). \tag{10}$$

We summarize the whole federated learning process on independent identically distributed data and non-independent identically distributed data in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Federated learning with personalized local differential privacy on independent identically distributed data.

Input: The total number of clients N, the maximum number of training rounds R, the maximum number of logistic regression rounds U, the clipping threshold c_i of client i, the learning rate η of SGD, the standard deviation σ_i of client i.

```
1:  **Server Side**
2:  Initialize $W^{(0)}$;
3:  **for** round $r = 1, 2, \ldots, R$ **do**
4:      **for** every client $i \in N$ **in parallel do**
5:          $\widetilde{G}_i^{(r)} \leftarrow LocalTrain(i, W^{(r-1)})$;
6:      **end for**
7:      $W^{(r)} \leftarrow f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)})$;//Model Aggregation
8:  **end for**

9:  **Client Side**
10: **LocalTrain($i, W^{(r-1)}$):**
11: **for** round $u = 1, 2, \ldots, U$ **do**
12:     $G_i^{(r)} \leftarrow W^{(r-1)} - \eta g_i^{(r-1)}$;
13: **end for**
14: $\widehat{G}_i^{(r)} \leftarrow ModelClip(c, G_i^{(r)})$;//Model Clipping
15: $\widetilde{G}_i^{(r)} \leftarrow M(\widehat{G}_i^{(r)})$//Model Perturbation
```

Algorithm 2 Federated learning with personalized local differential privacy on non-independent identically distributed data.

```
Input: The total number of clients $N$, the maximum number of training rounds $R$, the maximum number of logistic regression rounds $U$, the clipping threshold $c_i$ of client $i$, the learning rate $\eta$ of SGD, the standard deviation $\sigma_i$ of client $i$.
1: Server Side
2: Initialize $W^{(0)}$;
3: for round $r = 1, 2, \ldots, R$ do
4:     for every client $i \in N$ in parallel do
5:         $\widetilde{G}_i^{(r)} \leftarrow LocalTrain(i, W^{(r-1)})$;
6:     end for
7:     $W^{(r)} \leftarrow sign(f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)}))$;//Model Aggregation
8: end for

9: Client Side
10: LocalTrain($i, W^{(r-1)}$):
11: for round $u = 1, 2, \ldots, U$ do
12:     $G_i^{(r)} \leftarrow W^{(r-1)} - \eta g_i^{(r-1)}$;
13: end for
14: $\widehat{G}_i^{(r)} \leftarrow ModelClip(c, G_i^{(r)})$;//Model Clipping
15: $\widetilde{G}_i^{(r)} \leftarrow M(\widehat{G}_i^{(r)}) = ldpsign(\sigma_i, \widehat{G}_i^{(r)})$//Model Perturbation
```

4. Improved Aggregation Method

In the above section, we describe a federated learning process where clients perturb their local models to preserve privacy. It should be noted that, though different clients are allowed to choose different privacy budgets, when computing the federated model, the server treats all local models equally. While in fact, if some clients choose low privacy budgets, their local models will contain much noise. As a result, if all local models are treated equally, it can be foreseen that the accuracy of the federated model will be low. Therefore, if the client is allowed to choose a personalized privacy budget, it is necessary for the server to take some measures to adjust the weight of the client, in order to obtain a federated model with high accuracy. In this section, we propose two methods for the server to aggregate the local models. It is assumed that every client notifies the server of the privacy budget it has chosen before the iterative training process starts.

4.1. Weighted Average Method

In order to make the local models of clients with high privacy budgets contribute more to the federated model, and at the same time ensure the accuracy of the federated model, the first strategy we consider is to determine the weight of each client's contribution in the federated learning, according to the privacy budget set by each client. In the r-th round of the training, after collecting the perturbed local parameter $\widetilde{G}_i^{(r)}$ from each client i, the server first utilizes the privacy budget to compute a weighting coefficient α_i, for each client i, and then performs a weighted average operation to obtain $W^{(r)}$. Intuitively, α_i represents the importance of client i's local model in this round of learning. We compute α_i as follows:

$$\alpha_i = \frac{\rho_i}{\sum_{i=1}^{N} \rho_i}, \tag{11}$$

where ρ_i denotes the inverse of the standard deviation σ_i, of the Gaussian distribution of the i-th client, and $\alpha_i \leq 1$. The larger the standard deviation σ_i, the smaller the coefficient α_i is, and the smaller the influence of $\widetilde{G}_i^{(r)}$ on $W^{(r)}$.

Specifically, for independent identically distributed data, the model aggregation rule (line 7 in Algorithm 1) is now given by

$$W^{(r)} = f(\widetilde{G}_1^{(r)}, G_2^{(r)}, \dots, G_N^{(r)}) = \frac{\sum_{i=1}^{N} \alpha_i \widetilde{G}_i^{(r)}}{\sum_{i=1}^{N} \alpha_i}. \tag{12}$$

For non-independent identically distributed data, the model aggregation rule (line 7 in Algorithm 2) is now given by

$$W^{(r)} = sign(f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \dots, \widetilde{G}_N^{(r)})) = sign(\frac{\sum_{i=1}^{N} \alpha_i \widetilde{G}_i^{(r)}}{\sum_{i=1}^{N} \alpha_i}). \tag{13}$$

4.2. Probability-Based Selection Method

In the above method, we assign different weights to different clients, so that the client who has chosen a higher privacy budget will have a greater impact on the federated model. To further weaken the impact of clients who have chosen a small privacy budget, here we propose another strategy. The basic idea is to randomly select a subset of clients to participate in one round of model aggregation. For a client, the probability of being selected is determined by the privacy budget chosen. In short, choosing a high privacy budget means that the client does not need a high level of privacy protection, and it will not add too much noise to the local model's parameters. Correspondingly, a high probability will be assigned to the client's local model, and the client's local model is more likely to participate in this round of federated learning.

In the r-th round of training, after collecting the perturbed local parameter $\widetilde{G}_i^{(r)}$, of each client i, the server first randomly selects a subset of m clients, and then performs an arithmetic average operation on the local models from the selected clients, to obtain $W^{(r)}$. Let P_i denote the probability of client i being selected in this round. The probability P_i, is computed as:

$$P_i = \frac{\rho_i}{\sum_{i=1}^{N} \rho_i}, \tag{14}$$

where ρ_i denotes the inverse of the standard deviation σ_i, of the Gaussian distribution of the i-th client, and $0 < P_i \leq 1$. The larger the standard deviation σ_i, the smaller the probability P_i.

Before the iterative training process starts, the server will utilize the privacy budget to compute the probability P_i, for each client. Then, in each round of training, the server takes the following steps to select m clients. The server first generates a uniformly distributed random number $\omega \in [0, 1]$. For each client i, if there is $P_i > \omega$, then the client is selected. Otherwise, the client will not be selected.

Specifically, for independent identically distributed data, the model aggregation rule (line 7 in Algorithm 1) is now given by

$$W^{(r)} = f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \dots, \widetilde{G}_N^{(r)}) = \frac{1}{m} \sum_{i=1}^{m} \widetilde{G}_i^{(r)}. \tag{15}$$

For non-independent identically distributed data, the model aggregation rule (line 7 in Algorithm 2) is now given by

$$W^{(r)} = sign(f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \dots, G_N^{(r)})) = sign(\frac{1}{m} \sum_{i=1}^{m} \widetilde{G}_i^{(r)}). \tag{16}$$

5. Experiments

To evaluate the performance of the methods proposed in the above section, we conduct a series of experiments on real-world datasets. In general, we have verified the effectiveness of the proposed method on independent identically distributed data and non-independent identically distributed data. For all experiments, we use the federated model accuracy as the evaluation criterion, which is defined as the ratio of correctly classified samples to the total number of samples.

5.1. Datasets

All experiments are carried out on the MNIST (http://yann.lecun.com/exdb/mnist/, accessed on 25 November 2022), Fashion-MNIST (https://github.com/zalandoresearch/fashion-mnist, accessed on 25 November 2022), and forest cover-types (https://archive.ics.uci.edu/ml/datasets/Covertype, accessed on 25 November 2022) datasets. These three datasets are commonly used for classification tasks in federated learning [19,26].

The MNIST hand-written digits dataset, is composed of 0 to 9 handwritten digit pictures and digit labels. It consists of 60,000 training samples and 10,000 test samples, each of which is a 28 $\times$ 28 pixel grayscale handwritten digit picture. We use the digits dataset as well, which is the built-in dataset of scikit-learn. It is a copy of the test set of the MNIST hand-written digits dataset, which consists of 1797 samples with 64 attributes. Each sample is an 8 $\times$ 8 size image of integer pixels, in the range 0 to 16.

The Fashion-MNIST dataset consists of 60,000 training examples and 10,000 testing examples. Each example is a 28 $\times$ 28 size gray-level image, depicting an item from one of ten different fashion classes.

The forest cover-types dataset consists of 581,012 samples with 54 features. Samples correspond to 30 $\times$ 30 m^2 patches of forest in the US, collected for the task of predicting each patch's cover type, i.e., the dominant species of tree. There are seven cover types: spruce/fir, Lodgepole pine, Ponderosa pine, cottonwood/willow, aspen, Douglas-fir and Krummholz.The original dataset does not differentiate between training data and testing data. In our experiments, we randomly selected 500,000 samples as the training data, and 30,000 samples were used as testing data. The training data was evenly distributed to 10 clients.

5.2. Experiments on Independent Identically Distributed Data

We first conduct experiments to evaluate the performance of the proposed methods on IID data. Two scenarios are simulated. One is the non-personalized scenario, where all clients set the same privacy budget or do not add any Gaussian noise, and the other is the personalized scenario, where different clients set different privacy budgets. We use the FedAvg algorithm in the non-personalized scenario. The proposed methods are evaluated and compared with the arithmetic average method in the personalized scenario. The aggregation process of arithmetic average method is: $W^{(r)} = f(\widetilde{G}_1^{(r)}, \widetilde{G}_2^{(r)}, \ldots, \widetilde{G}_N^{(r)}) = \frac{1}{N}\sum_{i=1}^{N} \widetilde{G}_i^{(r)}$.

5.2.1. Experiment Setting

In the experiment, we train a logistic regression [29] model to solve the classification task. The algorithm used to minimize the loss function of logistic regression is SGD. The learning rate η, in the SGD algorithm, is calculated using the "optimal" strategy (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.\SGDClassifier.html, accessed on 25 November 2022) and all clients use the same learning rate.

1. MNIST: Experiments are configured with $N = 3$, $R = 10$, $batch = 50$, *test dataset size* $= 300$, $\delta_i = 1/500$, $q = 0.8$, $c = 200$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 1, 5, or 10. When simulating the personalized scenario, the privacy budgets of the three clients are set as $\{1, 1, 10\}$, $\{1, 5, 10\}$, or $\{1, 10, 10\}$.
2. Fashion-MNIST: Experiments are configured with $N = 3$, $R = 20$, $batch = 50$, *test dataset size* $= 600$, $\delta_i = 1/1000$, $q = 0.9$, $c = 2$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 0.05, 0.5, or 1. When simulating the personalized scenario, the privacy budgets of the three clients are set as $\{0.05, 0.05, 1\}$, $\{0.05, 0.5, 1\}$, or $\{0.05, 1, 1\}$.

3. Forest cover-types: Experiments are configured with $N = 10$, $R = 50$, $batch = 1000$, $test\ dataset\ size = 30000$, $\delta_i = 1/50000$, $q = 0.8$, $c = 2$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 1, 5, or 10. When simulating the personalized scenario, the privacy budgets of the ten clients are set as $\{1, 1, 1, 1, 1, 10, 10, 10, 10, 10\}$ or $\{1, 1, 1, 5, 5, 5, 5, 10, 10, 10\}$.

5.2.2. Results Discussion

The experiment results are shown in Figures 2–4. We denote the arithmetic average method as 'A'. We denote the weighted average method as 'B'. We denote the probability-based selection method as 'C'. In the non-personalized scenario, as we can see from Figures 2a, 3a, and 4a, when all clients set the same privacy budget, the larger the privacy budget, the higher accuracy of the federated model. The accuracy is highest when Gaussian noise is not added, because the local models from clients are not perturbed. When clients do not add Gaussian noise to the local model, the federated model accuracies for those three datasets are 88.7%, 73.1%, and 49%, respectively, and this shows that for IID data, the LDP-based federated framework is more applicable and effective for the MNIST and Fashion-MNIST datasets, than for the forest cover-types dataset.

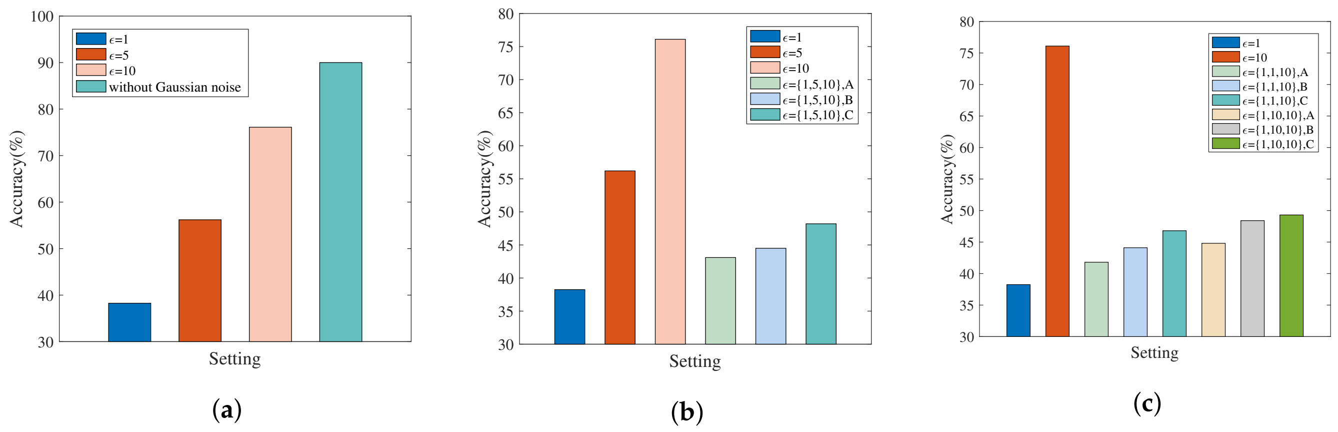

Figure 2. Simulation results on MNIST data. Clients' training data are independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting.

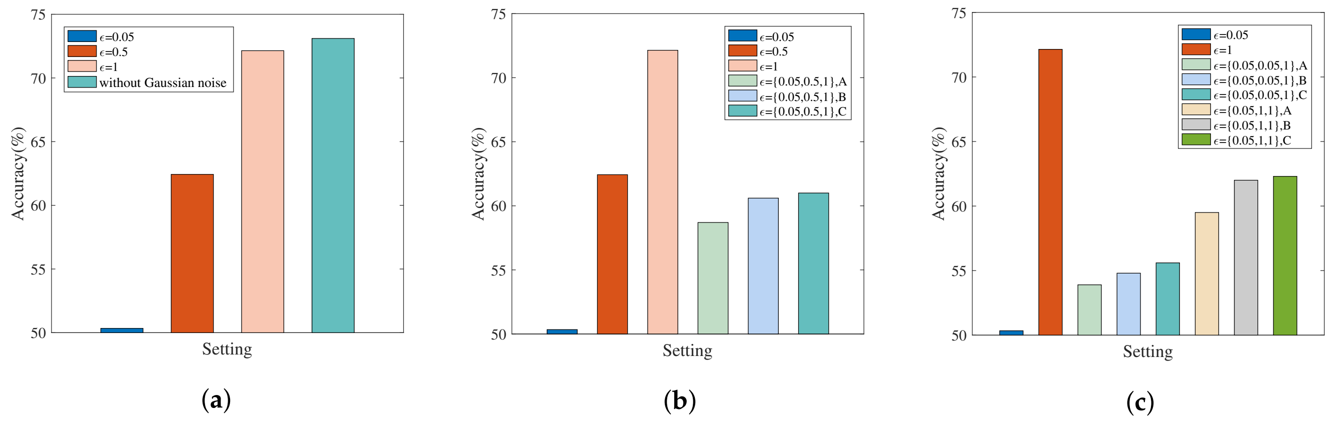

Figure 3. Simulation results on Fashion-MNIST data. Clients' training data are independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting.

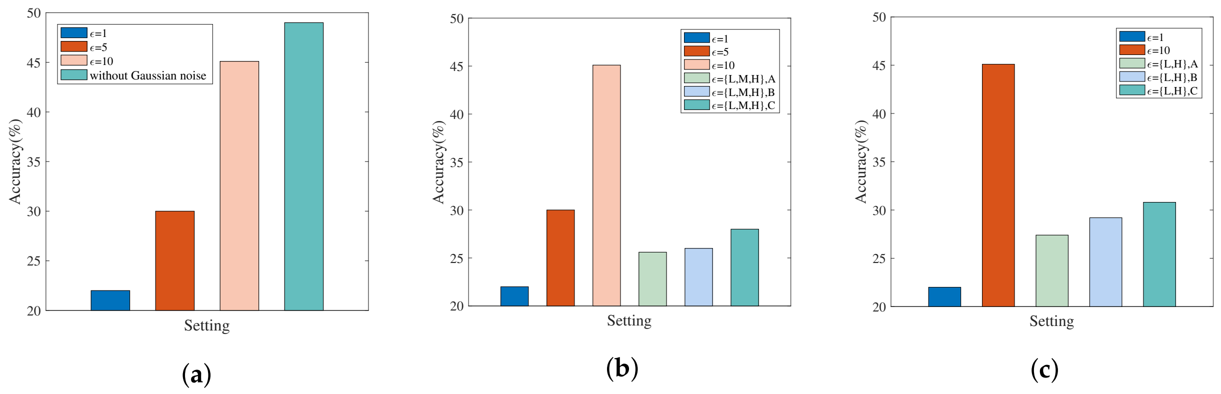

Figure 4. Simulation results on forest cover-types data. Clients' training data are independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting. We nominate $\{1, 1, 1, 5, 5, 5, 5, 10, 10, 10\}$ as $\{L, M, H\}$ (i.e., low, middle, high), and nominate $\{1, 1, 1, 1, 1, 10, 10, 10, 10, 10\}$ as $\{L, H\}$ (i.e., low, high).

In the personalized scenario, as can be seen from Figures 2b,c, 3b,c, and 4b,c, when more clients set large privacy budgets, the accuracy of the federated model increases. In addition, it is worth noting that under the same setting of privacy budgets, the accuracy of the probability-based selection method performs best among the three model aggregation methods, and both the probability-based selection method and the weighted average method perform better than the arithmetic average method. Compared with the arithmetic average method, those three datasets obtain accuracy gains in the weighted average method and the probability-based selection method. For the MNIST dataset, the accuracy gains obtained by the weighted average method and the probability-based selection method are 5.57% and 11.28%, respectively. For the Fashion-MNIST dataset, the accuracy gains obtained by the weighted average method and the probability-based selection method are 3.04% and 3.93%, respectively. For the forest cover-types dataset, the accuracy gains obtained by the weighted average method and the probability-based selection method are 4.07% and 10.90%, respectively. In general, both the weighted average method and the probability-based selection method perform best on the MNIST dataset.

5.3. *Experiments on Non-Independent Identically Distributed Data*

Our idea is to try to make the sample label distribution of each client different, that is, the samples of each category label need to be divided on different clients according to different proportions. Our strategy is to divide the samples according to the Dirichlet distribution.

In the non-IID experiment, the scenarios of the experiment and the methods used in each scenario are the same as the IID experiment. In particular, for non-IID experiments, the method without privacy (i.e., symbolized perturbation) is to change the 7th line of Algorithm 2, to $W^{(r)} \leftarrow sign(\frac{1}{N}\sum_{i=1}^{N}\widetilde{G}_i^{(r)})$, remove the 13th line, and change the 14th line to $\widetilde{G}_i^{(r)} \leftarrow sign(G_i^{(r)})$.

5.3.1. Experiment Setting

The logistic regression model, SGD algorithm, and learning rate η, used in the non-IID experiment, were the same as those in the IID experiment.

1. MNIST: Experiments are configured with $N = 10$, $R = 10$, $batch = 400$, $test\ dataset\ size = 2400$, $\delta_i = 10^{-5}$, $c = 4$, $\Delta_2 = 8$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 5, 10, or 15. When simulating the personalized scenario, the privacy budgets of the ten clients are set as $\{5, 5, 5, 5, 5, 15, 15, 15, 15, 15\}$ or $\{5, 5, 5, 10, 10, 10, 10, 15, 15, 15\}$.

2. Fashion-MNIST: Experiments are configured with $N = 10$, $R = 10$, $batch = 200$, $test\ dataset\ size = 1200$, $\delta_i = 10^{-5}$, $c = 4$, $\Delta_2 = 8$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 5, 10, or 15. When simulating the personalized scenario, the privacy budgets of the ten clients are set as $\{5,5,5,5,5,15,15,15,15,15\}$ or $\{5,5,5,10,10,10,10,15,15,15\}$.
3. Forest cover-types: Experiments are configured with $N = 10$, $R = 20$, $batch = 2500$, $test\ dataset\ size = 30{,}000$, $\delta_i = 10^{-5}$, $c = 2$, $\Delta_2 = 4$. When simulating the non-personalized scenario, each client's privacy budget ϵ_i, is set to 1, 5, or 10. When simulating the personalized scenario, the privacy budgets of the ten clients are set as $\{1,1,1,1,1,10,10,10,10,10\}$ or $\{1,1,1,5,5,5,5,10,10,10\}$.

5.3.2. Results Discussion

The experiment results are shown in Figures 5–7. We denote the arithmetic average method as 'A'. We denote the weighted average method as 'B'. We denote the probability-based selection method as 'C'. In the non-personalized scenario, as we can see from Figures 5a, 6a, and 7a, when all clients set the same privacy budget, the larger the privacy budget set, the higher the accuracy of the federated model; the accuracy is highest when there is no symbolized perturbation. When clients do not make symbolized perturbation to the local model, the federated model accuracies for those three datasets are 65.2%, 49.8%, and 62.6%, respectively, and this shows that for non-IID data, the LDP-based federated framework is more suitable and usable on the MNIST and forest cover-types datasets, than the Fashion-MNIST dataset.

In the personalized scenario, as we can see from Figures 5b,c, 6b,c, and 7b,c, when more clients set large privacy budgets, the accuracy of the federated model increases. In addition, it is worth noting that under the same setting of privacy budgets, the probability-based selection method still performs best on the MNIST and Fashion-MNIST datasets, as in the IID experiments. The MNIST and the Fashion-MNIST datasets obtain accuracy gains in the probability-based selection method, compared with the arithmetic average method. For the MNIST dataset, the accuracy gain obtained by the probability-based selection method is 6.97%. For the Fashion-MNIST dataset, the accuracy gain obtained by the probability-based selection method is 19.85%. The forest cover-types dataset does not obtain accuracy gains from the weighted average method and the probability-based selection method. In general, the probability-based selection method performs best on the Fashion-MNIST dataset.

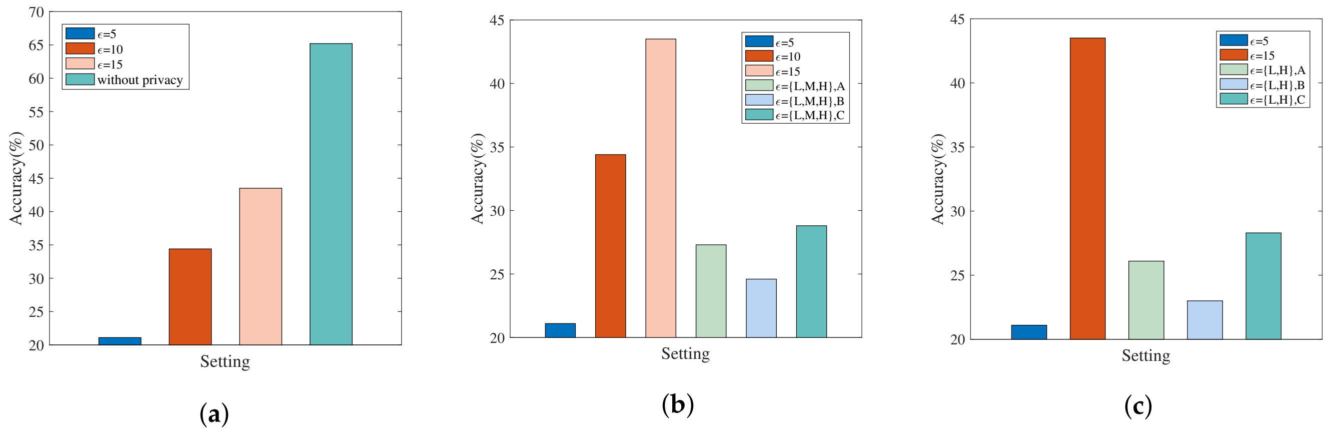

Figure 5. Simulation results on MNIST data. Clients' training data are non-independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting. We nominate $\{5,5,5,10,10,10,10,15,15,15\}$ as $\{L,M,H\}$ (i.e., low, middle, high), and nominate $\{5,5,5,5,5,15,15,15,15,15\}$ as $\{L,H\}$ (i.e., low, high).

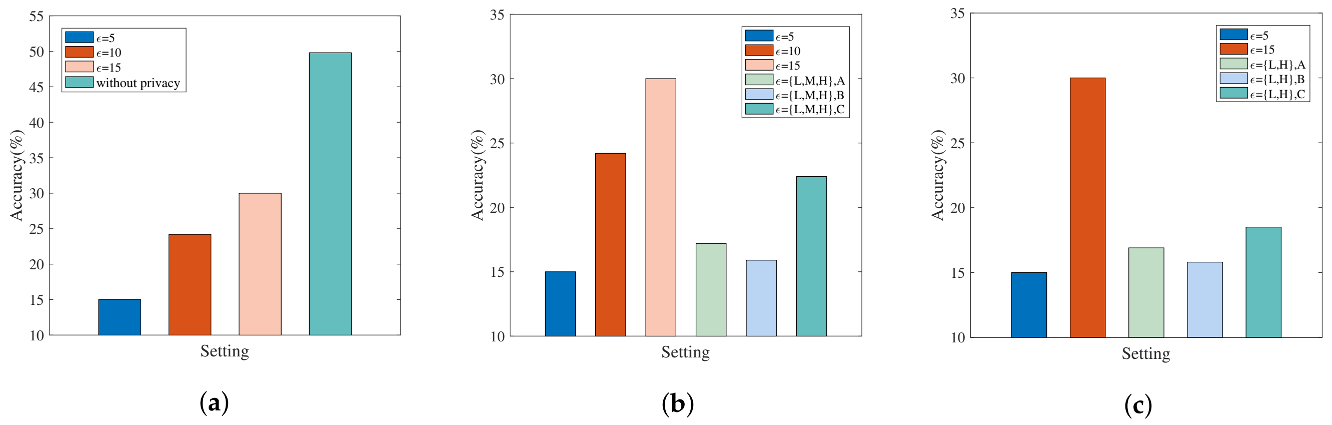

Figure 6. Simulation results on Fashion-MNIST data. Clients' training data are non-independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting.. We nominate $\{5,5,5,10,10,10,10,15,15,15\}$ as $\{L,M,H\}$ (i.e., low, middle, high), and nominate $\{5,5,5,5,5,15,15,15,15,15\}$ as $\{L,H\}$ (i.e., low, high).

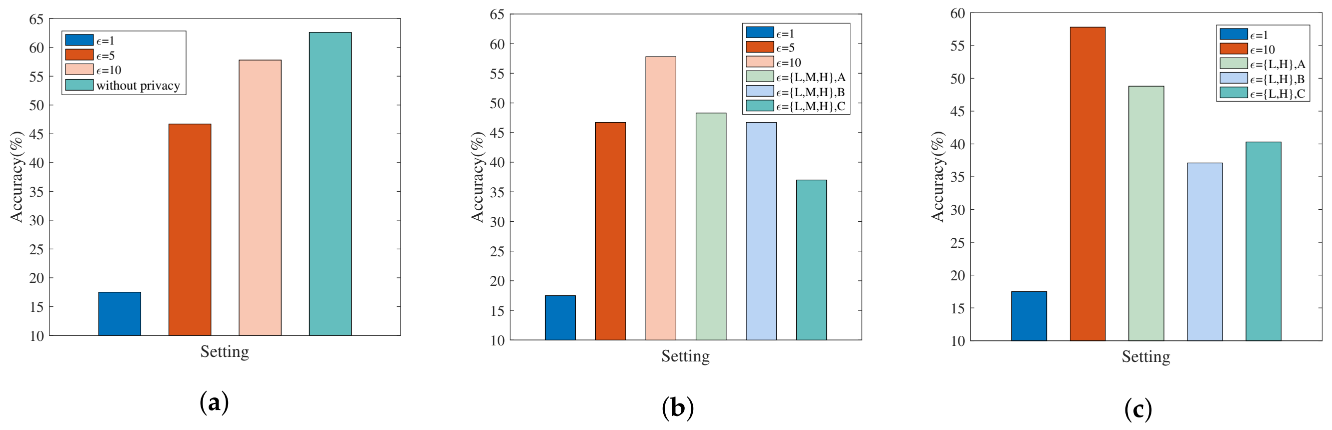

Figure 7. Simulation results on forest cover-types data. Clients' training data are non-independent and identically distributed. (**a**) The non-personalized scenario. (**b**) The personalized scenario under the first privacy setting. (**c**) The personalized scenario under the second privacy setting. We nominate $\{1,1,1,5,5,5,5,10,10,10\}$ as $\{L,M,H\}$ (i.e., low, middle, high), and nominate $\{1,1,1,1,1,10,10,10,10,10\}$ as $\{L,H\}$ (i.e., low, high).

6. Conclusions

The emergence of federated learning, breaks the barriers of joint training models between different parties, and also provides protection for sensitive data. In the client-server framework of federated learning, the server does not access the clients' original data. However, the clients still face the risk of privacy violation. Differential privacy is a kind of privacy protection technology based on strict mathematical theory. Previous studies have applied differential privacy to federated learning, to enhance privacy protection. Different clients have different privacy requirements. Previous DP-based approaches usually set the same privacy budget for all clients and cannot provide personalized privacy protection. In this paper, under the basic framework of LDP-based federated learning, we propose a weighted average method and a probability-based selection method for model aggregation, to obtain the trade-off between privacy and accuracy. The basic idea of the proposed methods is to assign different weights to different clients according to the privacy budgets set by clients. We consider two cases, where clients' training data are IID or non-IID. Different model perturbation methods are applied to satisfy the constraint of LDP. The experiments show that the performances of the proposed aggregation methods are better than that of the arithmetic average method. In the future, we plan to further explore the diversity of privacy requirement and design better model perturbation methods. We also plan to investigate other solutions for federated learning with non-IID data.

Author Contributions: Conceptualization, X.W. and L.X.; data curation, X.W.; formal analysis, X.W.; funding acquisition, L.X. and L.Z.; investigation, X.W.; methodology, X.W., L.X. and L.Z.; project administration, L.X. and L.Z.; resources, L.X.; software, X.W.; supervision, L.X. and L.Z.; validation, X.W.; visualization, X.W.; writing—original draft, X.W.; writing—review and editing, L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Natural Science Foundation of China under Grant No. 62272039 and in part by the Beijing Natural Science Foundation under Grant No. M21035. The APC was funded by the National Natural Science Foundation of China under Grant No. 62272039.

Data Availability Statement: MNIST dataset (http://yann.lecun.com/exdb/mnist/), Fashion-MNIST dataset (https://github.com/zalandoresearch/fashion-mnist), Forest Cover-types dataset (https://archive.ics.uci.edu/ml/datasets/Covertype).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Abdulrahman, S.; Tout, H.; Ould-Slimane, H.; Mourad, A.; Guizani, M. A Survey on Federated Learning: The Journey From Centralized to Distributed On-Site Learning and Beyond. *IEEE IoT* **2020**, *8*, 5476–5497. [CrossRef]
2. Fu, Y.; Luo, S.; Shu, J. Survey of Secure Cloud Storage System and Key Technologies. *J. Comput. Res. Dev.* **2013**, *50*, 136–145.
3. General Data Protection Regulation. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46. *Off. J. Eur. Union* **2016** . [CrossRef]
4. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient Learning of Deep Networks from Decentralized Data. *Artif. Intell. Stat.* **2017**, *54*, 1273–1282.
5. Shokri, M.N.R.; Houmansadr, A. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In Proceedings of the IEEE Symposium on Security and Privacy, San Francisco, CA, USA, 16 September 2019. [CrossRef]
6. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A Survey on Security and Privacy of Federated Learning. *FGCS* **2021**, *115*, 619–640. [CrossRef]
7. Yao, A.C. Protocols for Secure Computations. In Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science, Chicago, IL, USA, 18 July 2008. [CrossRef]
8. Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; Seth, K. Practical Secure Aggregation for Federated Learning on User-Held Data. *arXiv* **2016**, arXiv:1611.04482.
9. Abrego, L. On Data Banks and Privacy Homomorphisms . *Found. Secur. Comput.* **1978**, *76*, 169–179. [CrossRef]
10. Zhang, J.; Chen, B.; Yu, S.; Deng, H. PEFL: A Privacy-Enhanced Federated Learning Scheme for Big Data Analytics. In Proceedings of 2019 IEEE Global Communications Conference, Waikoloa, HI, USA, 9–13 December 2019. [CrossRef]
11. Dwork, C.; Roth, A. The Algorithmic Foundations of Differential privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. [CrossRef]
12. Kasiviswanathan, S.P.; Lee, H.K.; Nissim, K.; Raskhodnikova, S.; Smith, A.D. What Can We Learn Privately? In Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, Philadelphia, PA, USA, 2 December 2008. [CrossRef]
13. Duchi, J.C.; Jordan, M.I.; Wainwright, M.J. Local Privacy and Statistical Minimax Rates. In Proceedings of the 54th IEEE Annual Symposium on Foundations of Computer Science, Berkeley, CA, USA, 19 December 2013. [CrossRef]
14. McMahan, H.B.; Ramage, D.; Talwar, K.; Zhang, L. Learning Differentially Private Recurrent Language Models. *arXiv* **2018**, arXiv:1710.06963.
15. Wang, N.; Xiao, X.; Yang, Y.; Zhao, J.; Hui, S.C.; Shin, H.; Shin, J.; Yu, G. Collecting and Analyzing Multidimensional Data with Local Differential Privacy. In Proceedings of the 35th IEEE International Conference on Data Engineering, Macao, China, 6 June 2019. [CrossRef]
16. Truex, S.; Liu, L.; Chow, K.H.; Gursoy, M.E.; Wei, W. LDP-Fed: Federated Learning with Local Differential Privacy. *arXiv* **2020**, arXiv:2006.03637.
17. Mugunthan, V.; Peraire-Bueno, A.; Kagal, L. PrivacyFL: A Simulator for Privacy-preserving and Secure Federated Learning. *arXiv* **2020**, arXiv:2002.08423.
18. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated Machine Learning: Concept and Applications. *ACM TIST* **2019**, *10*, 1–19. [CrossRef]

19. Jin, R.; Huang, Y.; He, X.; Dai, H.; Wu, T. Stochastic-Sign SGD for Federated Learning with Theoretical Guarantees. *arXiv* **2020**, arXiv:2002.10940.
20. Warner, S.L. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *JASA* **1965**, *60*, 63–69. [CrossRef] [PubMed]
21. Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; Naor, M. *Our Data, Ourselves: Privacy Via Distributed Noise Generation*, 1st ed.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 486–503. [CrossRef]
22. Liu, R.; Cao, Y.; Chen, H.; Guo, R.; Yoshikawa, M. FLAME: Differentially Private Federated Learning in the Shuffle Model. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtually, 2–9 February 2021.
23. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated Learning with Non-IID Data. *arXiv* **2018**, arXiv:1806.00582.
24. Robbins, H.; Monro, S. A Stochastic Approximation Method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
25. Kim, M.; Gunlu, O.; Schaefer, R.F. Federated Learning with Local Differential Privacy: Trade-offs between Privacy, Utility, and Communication. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Toronto, ON, Canada, 13 May 2021. [CrossRef]
26. Sun, L.; Qian, J.; Chen, X. LDP-FL: Practical Private Aggregation in Federated Learning with Local Differential Privacy. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, Montreal, QC, Canada, 19–27 August 2021. [CrossRef]
27. Zhang, X.; Chen, X.; Hong, M.; Wu, S.; Yi, J. Understanding Clipping for Federated Learning: Convergence and Client-Level Differential Privacy. *arXiv* **2021**, arXiv:2106.13673.
28. Abadi, M.; Chu, A.; Goodfellow, I.; McMahan, H.B.; Mironov, I.; Talwar, K.; Zhang, L. Deep Learning with Differential Privacy. In Proceedings of the ACM Conference on Compututer and Communication Security, Vienna, Austria, 24–28 October 2016. [CrossRef]
29. Hosmer, D.W. Stanley Lemeshow. In *Applied Logistic Regression*, 1st ed.; Wiley: Hoboken, NJ, USA, 2000.