entropy

MDPI

Article

PLDP-FL: Federated Learning with Personalized Local Differential Privacy

Xiaoying Shen [1,2], Hang Jiang [1], Yange Chen [3], Baocang Wang [1,*] and Le Gao [4]

[1] The State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China
[2] The Key Laboratory of Cryptography of Zhejiang Province, Hangzhou Normal University, Hangzhou 310030, China
[3] The School of Information Engineering, Xuchang University, Xuchang 461002, China
[4] The Faculty of Intelligent Manufacturing, Wuyi University, Jiangmen 529000, China
* Correspondence: bcwang79@aliyun.com

Abstract: As a popular machine learning method, federated learning (FL) can effectively solve the issues of data silos and data privacy. However, traditional federated learning schemes cannot provide sufficient privacy protection. Furthermore, most secure federated learning schemes based on local differential privacy (LDP) ignore an important issue: they do not consider each client's differentiated privacy requirements. This paper introduces a perturbation algorithm (PDPM) that satisfies personalized local differential privacy (PLDP), resolving the issue of inadequate or excessive privacy protection for some participants due to the same privacy budget set for all clients. The algorithm enables clients to adjust the privacy parameters according to the sensitivity of their data, thus allowing the scheme to provide personalized privacy protection. To ensure the privacy of the scheme, we have conducted a strict privacy proof and simulated the scheme on both synthetic and real data sets. Experiments have demonstrated that our scheme is successful in producing high-quality models and fulfilling the demands of personalized privacy protection.

Keywords: federated learning; privacy protection; local differential privacy; personalization

Citation: Shen, X.; Jiang, H.; Chen, Y.; Wang, B.; Gao, L. PLDP-FL: Federated Learning with Personalized Local Differential Privacy. *Entropy* **2023**, *25*, 485. https://doi.org/10.3390/e25030485

Academic Editor: Wray Buntine

Received: 19 January 2023
Revised: 2 March 2023
Accepted: 9 March 2023
Published: 10 March 2023

1. Introduction

In recent years, artificial intelligence (AI) technology has made great strides, bringing about positive changes in areas such as finance, healthcare, and education [1]. However, owing to the heightened awareness of privacy protection and the implementation of laws and regulations, the problem of data islands has become increasingly noticeable, thus impeding the training of AI models. The introduction of Federated Learning (FL) [2,3] has been a critical technology for AI development, as it allows sharing data between multiple parties while guaranteeing user privacy and data security. Federated learning aims to establish a multi-party collaborative machine learning (ML) model, where the data of participants are kept in their local area, and only the intermediate parameters are exchanged and transferred to train the final model.

It has been noted in prior studies [4,5] that federated learning cannot provide sufficient privacy protection. External adversaries and internal adversaries can still infer clients' private information through gradient parameters or model parameters, such as model inversion attacks [6] and membership inference attacks [7]. Currently, there are many approaches to achieve secure federated learning, such as a combination of federated learning and secure multi-party computation (SMC) [8] or homomorphic encryption (HE) [9] schemes. They can offer robust protection measures. However, they also incur heavy computational and communication overhead. Differential privacy (DP) [10], a lightweight privacy protection technology with low computational and communication costs, has been widely applied in machine learning to protect privacy. A federated optimization algorithm for client-level differential privacy preservation has been presented in [11], which aims to hide the client's

contribution during the training process, while still achieving a balance between privacy loss and model performance. Local differential privacy has been suggested in [12–14] to address the privacy leakage problem in federated learning. As federated learning and local differential privacy are suitable for distributed architectures, their integration is a natural choice. Meanwhile, local differential privacy can ensure stringent privacy protection for federated learning, and effectively resist membership inference attacks.

However, the above-mentioned federated learning schemes based on local differential privacy have a common problem, which is that they assume that all local clients have the same privacy protection requirements, thus each local client has the same privacy budget. Considering the differences in privacy preferences and data sensitivity of each client in the real world, simply setting the same level of privacy protection will lead to a situation where some clients have insufficient privacy protection and some clients have excessive privacy protection. In light of this, this paper proposes a federated learning scheme based on personalized local differential privacy, to satisfy the local clients' differential privacy requirements. In summary, our main contributions are as follows:

- **Personalized perturbation mechanism**. We provide a novel personalized local differential privacy data perturbation mechanism. By adjusting the privacy budget parameters and introducing distinct security range parameters for each client, this perturbation mechanism enables clients to articulate their different privacy requirements and obtain various privacy protection.
- **Privacy-preserving framework**. We propose a privacy-preserving federated learning method to address the demand for personalized local differential privacy. By employing the personalized differential privacy perturbation algorithm to process the intermediate parameters, the server cannot deduce the participant's privacy information from the intermediate parameters, thereby achieving the personalized privacy protection requirements in the federated learning process.
- **High-quality model**. Extensive experimental results show that our scheme can obtain a high-quality model when the clients have privacy-preserving requirements. Thus, it is more suitable for applications with device heterogeneity in the real world.

Organization. In the following, we introduce related work in Section 2. Then, we introduce the preliminaries in Section 3. Section 4 gives the system overview and Section 5 gives the details of the PLDP-FL. The performance evaluation and experimental results are shown in Section 6. Section 7 concludes the paper. A summary of basic concepts and notations is provided in Table 1.

Table 1. Summary of Main Notation.

$\mathcal{M}$	A randomized algorithm for DP
v, v'	Any pair of input values
$\widetilde{v}$	The perturbed value of the input value v
ϵ_i, τ_i	The privacy parameters of client P_i
L_i, c_i	The length and midpoint of the safe range, respectively
P_i	The i-th client
D_i	The dataset held by client P_i
x_i, y_i	The input data and true label in the dataset D_i, respectively
N	The number of all clients
$\lvert\mathcal{P}_t\rvert$	The number of chosen clients
γ	The local client selection factor
T	The number of aggregation rounds
$\mathcal{L}$	Loss function
w^*	The optimal model parameters that minimize $\mathcal{L}$
$w_{i,t}$	Local parameters of the i-th client in the t-th round
$\widetilde{w}_{i,t}$	Local parameters after perturbation of the i-th client in the t-th round
$\overline{w}_t$	Global parameters after aggregation

2. Related Work

Federated learning ensures the security of the original data by transferring only the intermediate calculation results. However, in some cases, the original data can still be retrieved if the intermediate parameters are attacked [4–7]. There are still security risks associated. To bridge the gap of common federated learning techniques, secure federated learning has been proposed in academia and industry. Cryptographic schemes based on homomorphic encryption and secure multi-party computation [8,9,15–17] can safeguard the intermediate parameters and results from attacks. The schemes based on differential privacy [11–14,18–27] ensure the security of process and result, but also introduce a certain amount of computational error.

Cryptographic techniques. Truex et al. [8] combined secure multi-party computation and differential privacy to defend against membership inference attacks and protected data privacy in federated learning. Threshold pailier encryption is employed to implement secure multi-party computation [28]. The noise of differential privacy injection is reduced by introducing a trusted parameter, while solving the problem of the existence of untrustworthy clients. Xu et al. [16] to further reduce the communication costs in [8], they combined a secure multi-party computation protocol implemented based on function encryption and differential privacy to protect data privacy in federated learning. Xu et al. [9] proposed a federated deep learning framework to address the issue of untrusted clients, utilizing Yao's obfuscation circuit and additive homomorphic encryption to protect the private data of clients. Phong et al. [15] combined additive homomorphic encryption and stochastic gradient descent algorithm to ensure that the server will not be aware of the client's private information. The homomorphic encryption method in their scheme provided two implementations, one based on LWE and the other based on Pallier.

Differential privacy. Cryptography-based schemes suffer from low computational performance and high communication costs, so methods based on differential privacy technology have received increasing attention. Shokri et al. [18] first applied differential privacy to privacy protection in deep learning. Nevertheless, it required a large amount of privacy budget to ensure the model's utility. Thus its privacy protection is not very adequate. Subsequently, Abadi et al. [19] also employed differential privacy to protect the gradient information in the random gradient descent algorithm, and proposed a novel algorithm to improve the privacy cost in the learning process. Hu et al. [20] considered the practical problem of user heterogeneity in the distributed learning system, and proposed a privacy-preserving method that satisfies differential privacy, which is used to learn a personalized model on distributed user data.

Local differential privacy. Chamikara et al. [27] proposed a new local differential privacy algorithm using a random response technique, which was inserted between the convolution layer and full connection layer of the convolutional neural network during training, to achieve privacy protection by perturbing the binary encoding of the input data. Truex et al. [12] proposed a new federated learning system for protecting privacy in deep neural networks by using a variant of the LDP algorithm CLDP. Although their approach can handle complex models, its corresponding privacy budget parameter is too large to be practical. Sun et al. [26] introduced a noise-free differential privacy concept that employs sampling (with or without replacement) of the client's data samples during the FL process, achieving improved performance. Sun et al. [14] proposed a federated learning scheme based on local differential privacy. This approach considers different value ranges of model parameters, and designs corresponding perturbation algorithms to ensure parameter privacy. At the same time, this scheme introduces a parameter transformation process that rearranges and shuffles the parameters uploaded by the client, thus providing enhanced privacy protection. However, the shuffling mechanism in this method is equivalent to introducing another third party, which not only increases the security burden of the system, but also increases the communication overhead of the system.

Personalized privacy models. With the emerging demand for privacy, personalized privacy research [29–34] has attracted more and more attention. Nie et al. [29] first proposed

a utility optimization framework, for histogram estimation with personalized multilevel privacy. To clarify the goal of privacy protection, they personalized the traditional definition of LDP. Shen et al. [30] considered the personality of data owners in protecting and utilizing multidimensional data by introducing the concept of personalized local differential privacy. They designed personalized multiple optimized unary encoding to perturb data owners' data, and proposed an aggregation algorithm for frequency estimation of multidimensional data under personalized local differential privacy. Xue et al. [31] used a personalized local differential privacy model based on previous mean estimation schemes under LDP to design novel methods to provide users with personalized privacy. Yang et al. [34] first proposed an algorithm PLU-FedOA to optimize horizontal federated learning with personalized local differential privacy, which allows clients to select the privacy level. Nevertheless, this method has two drawbacks. Firstly, the method only has one privacy parameter, and the range of parameters and parameter values is too wide, resulting in inadequate privacy protection. Secondly, the method enforces differential privacy protection by incorporating Laplace noise. When the privacy budget parameter is small, the Laplace noise will generate a large variance, thus significantly impacting the model performance. To solve the above problems, we propose a novel federated learning scheme based on personalized local differential privacy.

3. Preliminaries

3.1. Local Differential Privacy

Differential privacy [35] is a privacy-preserving technique that does not limit adversary capabilities, and can accurately define and evaluate privacy protection quantitatively. Differential privacy can ensure two security objectives: first, the adversary cannot carry out the link attack. Second, even if the attacker can verify that the target is in the data set, they will not be able to gain accurate information about the target through the differential output result. Traditional differential privacy, also known as global differential privacy (GDP), and its implementation depends on a trusted third party. However, in reality, it is difficult to find a completely trusted third party, so there are limitations in applying global differential privacy. Local differential privacy is a distributed variant of GDP that allows each client to perturb its data locally and then upload them to the server, so the server never has access to the client's real data.

Definition 1 ((ϵ-LDP) [36]). *Assuming that $\mathcal{M}$ is a randomized algorithm, D and D' are two separate datasets with only one dissimilar datum, if and only if for any output $O \in Range(\mathcal{M})$ of the randomized algorithm $\mathcal{M}$, the following inequality is hold.*

$$\Pr[\mathcal{M}(v) = O] \leq e^{\epsilon} \cdot \Pr\left[\mathcal{M}\left(v'\right) = O\right], \tag{1}$$

The randomized algorithm $\mathcal{M}$ is said to satisfy ϵ-LDP, where $\epsilon \geq 0$, $Range(\mathcal{M})$ denotes the set of all possible outputs of $\mathcal{M}$, $\Pr[\cdot]$ denotes the probability, and ϵ denotes the privacy budget.

Differential privacy has become one of the more popular privacy-preserving techniques due to its remarkable features, such as post-processing invariance and composability, which are essential for its transition from theory to practical application.

Proposition 1 (Post-processing). *Given a randomized algorithm $\mathcal{M}_1$ satisfies ϵ-DP, for any randomized algorithm $\mathcal{M}_2$, the combination $\mathcal{M}_2(\mathcal{M}_1)$ of $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfies ϵ-DP.*

Proposition 2 (Sequential composition). *Assuming that each $\mathcal{M}_i$ provides ϵ_i-DP, the sequence combination of $\mathcal{M}_i(D)$ provides $\sum_i \epsilon_i$-DP. Where $\mathcal{M}_i$ denotes a differential privacy algorithm satisfies ϵ_i, and $\mathcal{M}_i(D)$ denotes all the algorithms act on the same dataset D.*

3.2. Personalized Local Differential Privacy

A significant characteristic of local differential privacy is that individuals can independently disturb data to protect their privacy. As participants are the ones who understand their privacy requirements best, there may have varying privacy requirements dependent on the sensitivity of their data. Therefore, it is necessary to promote the concept of personalized privacy in local settings. Chen et al. [37] suggested a novel personalized local differential privacy model for learning the distribution of users' spatial data. This paper utilizes the same method found in [37] to characterize the various privacy preferences of clients.

Definition 2 ((τ, ϵ)-PLDP) [37]**).** *Assuming that $\mathcal{M}$ is a randomized algorithm, given the personalized privacy parameters (τ, ϵ) of the client and any pair of input values $v, v' \in \tau$, if and only if for any output $O \subseteq Range(\mathcal{M})$ of the randomized algorithm $\mathcal{M}$, the following inequality holds.*

$$\Pr[\mathcal{M}(v) \in O] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(v') \in O]. \tag{2}$$

Then the randomized algorithm $\mathcal{M}$ is said to satisfy (τ, ϵ)-PLDP, where τ denotes safe range, $Range(\mathcal{M})$ denotes the set of all possible outputs of $\mathcal{M}$, $\Pr[\cdot]$ denotes the probability, and ϵ denotes the privacy budget.

3.3. Random Response

3.3.1. Random Response

Random response (RR) [38] began as a method for sampling and has since become a popular technology for implementing local differential privacy. The main idea of random response is to protect users' privacy by providing a plausible answer to the sensitive question. Flipping a coin is the most common method for answering a binary value. Utilizing coin flip technology, it can answer privacy-related questions. If the result is positive, the user will answer truthfully, and a second coin flip will be executed if the result is negative. If the result is positive, the answer is yes, and if the result is negative, the answer is no. For the data collector, he knows the probability of the user's true answer. For the user, the data collector does not know the value of his answer, so the user's privacy is protected to some extent.

This random response mechanism fulfills $(ln3, 0)$ differential privacy. It is obvious that the probability of users giving a YES answer when the true value is Yes is 0.75, and the chance of a YES response when the real value is No is 0.25. When the response result is determined, the following equation holds.

$$\frac{\Pr[Ans = YES \mid Truth = Yes]}{\Pr[Ans = YES \mid Truth = No]} = 3. \tag{3}$$

We can also understand the random response in another way, i.e., the user having a probability of p to answer the true value and a probability of $1 - p$ to answer the opposite value. Given a response result, we can obtain the ratio between the probability of responding to the true result and the probability of responding to other results $e^{\epsilon} = \frac{p}{1-p}$. Further, we get $p = \frac{e^{\epsilon}}{e^{\epsilon}+1}$, it can provide $ln(\frac{p}{1-p})$-LDP. To ensure that the privacy budget is greater than 0, it is generally recommended that the value of p be higher than 0.5.

3.3.2. Generalized Random Response

Generalized random response(GRR) [39] is a generalization of the RR for the case of answering multiple choice, also known k-RR. The idea is similar to that of RR, i.e., there is a large probability of providing an honest answer, whereas the probability of giving a false answer is small. On the one hand, The total of all probability values should be 1. On the other hand, the ratio of large probability to small probability needs to satisfy the definition

of $\epsilon-$LDP, where $\epsilon = ln(\frac{(k-1)p}{1-p})$. Currently, the values of p and q fulfill the following equation under optimal local difference privacy.

$$\Pr[\mathcal{M}(v) = \widetilde{v}] = \begin{cases} p = \frac{e^{\epsilon}}{e^{\epsilon}+k-1}, \text{ if } \widetilde{v} = v, \\ q = \frac{1}{e^{\epsilon}+k-1}, \text{ if } \widetilde{v} \neq v, \end{cases} \tag{4}$$

where v is the original input value, $\widetilde{v}$ is the perturbed value, p is the probability of obtaining the true response, and q is the probability of responsing the other values.

3.4. Federated Deep Learning

Federated learning is a distributed machine learning approach that allows sharing of data and training models while protecting privacy. Google's McMahan et al. [2] pioneered the use of federated learning to modify the language prediction model on smartphones. This approach focuses on achieving a balance between data privacy protection and sharing by exchanging the intermediate parameters instead of the original data when multiple participants collaborate to train the model.

Federated Average Algorithm (FedAvg) is the core algorithm employed in federated learning. FL generally involves local clients and parameter server, and each client possesses a private dataset $D_i = (x_i, y_i)_{i=1}^{N}$, where x_i and y_i are the input data and true label, respectively, and N is the number of clients. Suppose the loss function of a learning task is $\mathcal{L}(w; x, y)$, where w is the weight parameter, x and y are the training sample. The learning goal is to construct an empirical minimization such as $w^* = argmin_w \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(w; x_i, y_i)$ on data from N clients. During each round of $t \in [0, T]$, the participant P_i performs the model update in parallel on the local dataset using the stochastic gradient descent (SGD) algorithm,

$$w_{t+1}^{i} = w_t^i - \eta \frac{\partial \mathcal{L}_i}{\partial w_t^i}. \tag{5}$$

Then sends the local update parameters to the server for global aggregation,

$$\overline{w}_{t+1} = \frac{1}{N} \sum_{i=1}^{N} w_{t+1}^{i}. \tag{6}$$

Finally, after aggregating the model parameters, send them back to the client and repeat the process until either the maximum number of training rounds T is reached or the model converges.

4. System Overview

4.1. Problem Definition

This paper focuses on the federated learning problem in the context of personalized local differential privacy. Its goal is to collaborate with N clients and a semi-honest server to train a global model, while respecting the varying levels of privacy protection desired by each participant. Each client has pairs of privacy parameters (τ_i, ϵ_i) that characterize its privacy preferences. ϵ_i is the privacy budget, and τ_i reflects the personalized privacy requirements. Without loss of generality, if the client's original data range is assumed to be $[-1, 1]$ and is uploaded to the server without any processing, it is vulnerable to certain attacks that could expose the victim's private information. This paper ensures the security of the entire process by introducing personalized local differential privacy to perturb the data uploaded by the client.

4.2. System Model

Federated learning scheme based on personalized local differential privacy (PLDP-FL) involves local clients and parameter server, and its system model is shown in Figure 1. In the following, we will explain the functions of each entity within the system.

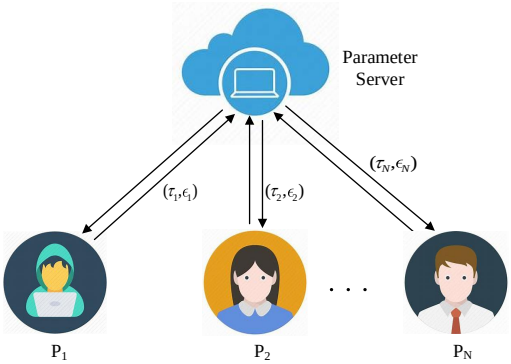


Figure 1. PLDP-FL Framework.

Local clients: Assume that there are N local clients $P = \{P_i | 1 \leq i \leq N\}$ in the system, and each client P_i possesses its private dataset, while they have full authority over their local data. During each round of iterative updates, clients use their local data for model parameter updates, and then upload them to the server for global updates. To ensure the security of their private data during the interaction of intermediary parameters, each client P_i perturbs the intermediate parameters using the local differential privacy algorithm before uploading them to the server. At the same time, each client P_i can set its own desired privacy parameters according to the sensitivity of its data and privacy preference. Finally, each client receives the update results from the server for a new round of local updates.

Parameter server: Assuming that the system is only equipped with a parameter server, it is endowed with powerful computing and storage capabilities. This server acts as a collaborator among all participants, receiving the intermediate parameters uploaded by them for global update operations and enabling data sharing among them.

4.3. Threat Model

In this scenario, we assume that the parameter server is honest and curious, meaning it will strictly execute the process of federated learning accurately, and return the correct results to all clients after global aggregation of the received intermediate parameters. Furthermore, it will be curious to the participants' private information, and it can obtain this information by analyzing the participants' intermediate parameters. All local participants are expected to be honest and curious. They will adhere to the model training process and not disrupt the learning process with malicious intent. Additionally, they shall attempt to acquire the private information of other local participants through aggregation results or public channels. We further assume that the parameter server will collaborate with some local participants, but only with some. There is an active adversary $\mathcal{A}$ in our proposed scheme. The goal of $\mathcal{A}$ is to infer the private information of participants by utilizing intermediate parameters or aggregation results. Specifically, the adversary $\mathcal{A}$ in PLDP-FL has the following capabilities.

- The adversary $\mathcal{A}$ can eavesdrop on intermediate parameters in the interaction of each entity via the public channel, thus inferring the client's private information from them.
- The adversary $\mathcal{A}$ can compromise the parameter server, and exploit the server's intermediate parameters or aggregation results to deduce the private information of the local participant.
- The adversary $\mathcal{A}$ can corrupt one or more local participants, and use their information to infer the privacy details of other local participants. It is not permissible to corrupt all local participants simultaneously.

4.4. Design Goal

According to the above system model and threat model. The proposed scheme must ensure data privacy, local participants' personalized privacy functionality, and the model's accuracy.

- **Data privacy**: Our scheme should guarantee that adversaries cannot access local participants' private data, either directly or by using the intermediate parameters and aggregated results.
- **Local participants' personalized privacy functionality** Considering the varying sensitivities of each participant's private data and their privacy preferences. Our scheme should ensure that all participants can adjust their privacy parameters to satisfy their personalized privacy protection requirements.
- **Model's accuracy** Our scheme should ensure that the model converges in theory, while also ensuring its practical feasibility, i.e., the privacy-protected and non-privacy-protected federated learning should be able to train models that are almost the same.

5. Proposed Scheme

In this section, we will discuss the implementation details of PLDP-FL, including the steps of implementation and the critical personalized perturbation algorithm. Specifically, the tasks of the parameter server and local participants, as well as the implementation steps of the scheme, will be outlined, and a corresponding implementation flow chart will be presented. Moreover, the perturbation algorithm is discussed in detail, including its algorithm description and privacy proof.

5.1. Steps of Implementation

The PLDP-FL scheme involves N local participants $P = \{P_1, P_2, \ldots, P_N\}$, and a parameter server. Each participant has their own private data set, with varying privacy sensitivity. The aim is to collaboratively train a global model with their local data, without compromising any privacy. Algorithm 1 describes the overall process of PLDP-FL, which mainly consists of two phases: server update and client update.

Algorithm 1 PLDP-FL

Require: N is the number of local clients, γ is the local client selection factor, $0 < \gamma \leq 1$, E is the number of local epochs, B is the local mini-batch size, η is the learning rate, $\mathcal{L}$ is the loss function.

Ensure: The trained model W.

ServerUpdate:
Initialize the weight parameter w_0 and send it to all clients;
for each round $t = 1, 2, \ldots, T$ **do**
  The server randomly selects $\gamma \cdot N$ local clients $\mathcal{P}_t$;
  for each client $P_{i \in [N]}$ in parallel **do**
    $w_{i,t} \leftarrow$ ClientUpdate $(P_i, \overline{w}_{t-1})$ and sends the result to the server;
  end for
  $\overline{w}_t \leftarrow \frac{1}{|\mathcal{P}_t|} \sum_i w_{i,t}$;
  return $\overline{w}_t$.
end for
ClientUpdate$(P_i, \overline{w}_{t-1})$**:**
$t \leftarrow t + 1$;
Receive the latest updates from the server: $w_{i,t} \leftarrow \overline{w}_{t-1}$;
for each local epoch $j = 1, 2, \ldots, E$ **do**
  for each batch $b \in B$ **do**
    $w_{i,t} \leftarrow w_{i,t} - \eta \cdot \nabla \mathcal{L}(w_{i,t}; b)$;
  end for
end for
$\widetilde{w}_{i,t} \leftarrow$Personalized$(w_{i,t})$.

5.1.1. Server Update

First, the server initializes the global model parameter w_0 and distributes it to all clients, thus providing them with the same model structure. During each round of interaction, the server randomly selects $\gamma \cdot N$ local clients to upload model parameter, instead of selecting all clients. The large number of local clients results in a significant communication overhead when sending parameters to the server, and this overhead increases with the number of clients. Therefore, randomly selecting a certain proportion of clients for parameter sharing can decrease the communication overhead. For the aggregation operation, we use the gradient average method

$$\overline{w}_t \leftarrow \frac{1}{|\mathcal{P}_t|} \sum_i w_{i,t} \,, \tag{7}$$

where $|\mathcal{P}_t|$ indicates the number of randomly chosen participants and $w_{i,t}$ denotes the model parameters of participants P_i in round t. Owing to the local personalized differential privacy perturbation algorithm being unbiased, the server does not need to carry out any further operations after the aggregation operation. After the aggregation is completed, the server will check if the model has converged or if the maximum number of communication rounds has been reached. If either of these conditions is fulfilled, the server will notify the participants to stop training and send the final model to all participants. Otherwise, it will transmit the aggregation results to the participants to continue training.

5.1.2. Client Update

Upon receiving the latest model parameters from the server, each participant replaces them with the model parameters of the current round. Then, they perform forward and back propagation to update the parameters. A total of E iterations are required to be executed locally. During each iteration, B training samples are randomly chosen. In forward propagation, the training data are extracted and compressed sequentially via the stacked convolution, ReLU, and pooling layers. The training sample's prediction label value is finally obtained through the full connection layer. Then, calculate the cross entropy loss function value $\mathcal{L}_{i,t}$ by the predicted label values and their corresponding true label values,

$$\mathcal{L}_{i,t} = \frac{1}{B} \sum_i -[y_i ln(p_i) + (1 - y_i) ln(1 - p_i)]. \tag{8}$$

where y_i represents the true label value of the sample, while p_i denotes the predicted label value of the sample. In backward propagation, the gradient value of the weight parameter is obtained by calculating the partial derivative of the weight parameter in relation to the loss value. Then the weight parameter is updated through the small batch gradient descent algorithm,

$$\omega_{i,t} = \omega_{i,t} - \eta \cdot \bigtriangledown \mathcal{L}(\omega_{i,t}; b), \tag{9}$$

where $\bigtriangledown \mathcal{L}(\omega_{i,t}; b)$ denotes the gradient value.

Since the server is not completely honest, the updated weight parameters of each client cannot be uploaded directly. Therefore, we employ the personalized local differential privacy algorithm to perturb them before uploading. Hereafter, we will discuss the detailed process of the perturbation algorithm.

5.2. Personalized Perturbation Algorithm

The existing local differential privacy protection mechanism has a significant limitation that assumes all clients have the same privacy protection level. That is, the same privacy budget is set. Since each client has different data sensitivity requirements, data privacy protection can either be inadequate or excessive. Inspired by the ideas of Chen et al. [37] and Xue et al. [31], we propose a novel perturbation algorithm PDPM that satisfies personalized local differential privacy. Its pseudocode is shown in Algorithm 2. The PDPM algorithm involves two stages.

Algorithm 2 PDPM

Require: The personalized privacy parameters (τ_i, ϵ_i), the range length of τ_i is $L_i = |\tau_i|$, the midpoint of τ_i is c_i, and the client's input value $w_i \in \tau_i$.

Ensure: The value after perturbation $\widetilde{w}_i$.

if $c_i \neq 0$ **then**
  Translating c_i to point 0 so that τ_i is symmetric about point 0;
  $v_i = w_i - c_i$;
else
  $v_i = w_i$.
end if
Sampling a random variable a such that
  $Pr[a = L_i/2] = \frac{v_i(e^{\epsilon_i}-1)}{L_i(e^{\epsilon_i}+2)} + \frac{e^{\epsilon_i}+1}{2(e^{\epsilon_i}+2)}$;
if $a = L_i/2$ **then**
  $\widetilde{v}_i = \frac{L_i(e^{\epsilon_i}+3)}{2(e^{\epsilon_i}-1)}$;
else if $a = -L_i/2$ **then**
  $\widetilde{v}_i = -\frac{L_i(e^{\epsilon_i}+1)}{e^{\epsilon_i}-1}$;
else
  $\widetilde{v}_i = 0$;
end if
$\widetilde{w}_i = \widetilde{v}_i + c_i$;
return $\widetilde{w}_i$.

5.2.1. Set Privacy Parameter

Each participant sets their privacy parameter pair (τ_i, ϵ_i), where τ_i denotes the security interval, the client's true value $w_i \in \tau_i$, τ_i is the minimum interval in which the clients' expected true value is indistinguishable, and ϵ_i is privacy budget that indicates the privacy protection level. Let L_i denote the length of τ_i and c_i represents the center of τ_i. As each participant can define their privacy parameters, their input value range varies, increasing the complexity of implementing the perturbation algorithm. To achieve a unified perturbation algorithm, we request that each participant first determine if the center c_i of the security interval is at zero. If so, no operation is necessary. Otherwise, the center c_i of the security interval will be shifted to zero, producing a symmetrical security interval. The original data are correspondingly translated to $v_i = w_i - c_i$, so that each client's input value $v_i \in [-L_i/2, L_i/2]$.

5.2.2. Compute Perturbation Value

The perturbation value $\widetilde{v}_i$ of v_i is obtained for each client according to the following probabilities.

$$\Pr[\mathcal{M}(v_i) = \widetilde{v}_i] = \begin{cases} \frac{v_i(e^{\epsilon_i}-1)}{L_i(e^{\epsilon_i}+2)} + \frac{(e^{\epsilon_i}+1)}{2(e^{\epsilon_i}+2)}, \text{ if } \widetilde{v}_i = \frac{L_i(e^{\epsilon_i}+3)}{2(e^{\epsilon_i}-1)}, \\ \frac{e^{\epsilon_i}+3}{4(e^{\epsilon_i}+2)} - \frac{v_i(e^{\epsilon_i}-1)}{2L_i(e^{\epsilon_i}+2)}, \text{ if } \widetilde{v}_i = \frac{-L_i(e^{\epsilon_i}+1)}{e^{\epsilon_i}-1}, \\ \frac{e^{\epsilon_i}+3}{4(e^{\epsilon_i}+2)} - \frac{v_i(e^{\epsilon_i}-1)}{2L_i(e^{\epsilon_i}+2)}, \text{ if } \widetilde{v}_i = 0, \end{cases} \tag{10}$$

where $\mathcal{M}$ denotes the perturbation algorithm. The algorithm produces the corresponding disturbance value under this probability. Since the perturbation is applied to the translated value $\widetilde{v}_i$, the original value $\widetilde{w}_i = \widetilde{v}_i + c_i$ is obtained.

Theorem 1. *Algorithm 2 provides (τ_i, ϵ_i)-PLDP for each client. In addition, the perturbation of algorithm satisfy unbiasedness, i.e., $\mathbb{E}(\widetilde{w}_i) = w_i$, and $Var[\widetilde{w}_i] \leq \frac{(e^{\epsilon_i}+1)(e^{\epsilon_i}+3)}{L_i(e^{\epsilon_i}-1)^2}$.*

Proof. For any $\widetilde{w}_i \in \{\frac{L_i(e^{\epsilon_i}+3)}{2(e^{\epsilon_i}-1)} + c_i, -\frac{L_i(e^{\epsilon_i}+1)}{e^{\epsilon_i}-1} + c_i, c_i\}$, and any two values $w_i, w_i' \in \tau_i$, we have $v_i = w_i - c_i$, $v_i' = w_i' - c_i$, $\widetilde{v}_i = \widetilde{w}_i - c_i$. Then,

$$\begin{aligned}\frac{\Pr[\widetilde{w}_i \mid w_i]}{\Pr[\widetilde{w}_i \mid w_i']} &= \frac{\Pr[\widetilde{v}_i \mid v_i]}{\Pr[\widetilde{v}_i \mid v_i']} \\ &= \frac{2v_i(e^{\epsilon_i}-1) + L_i(e^{\epsilon_i}+1)}{2v_i'(e^{\epsilon_i}-1) + L_i(e^{\epsilon_i}+1)} \\ &\leq \frac{L_i(e^{\epsilon_i}-1) + L_i(e^{\epsilon_i}+1)}{-L_i(e^{\epsilon_i}-1) + L_i(e^{\epsilon_i}+1)} \\ &= e^{\epsilon_i}.\end{aligned} \tag{11}$$

Thus Algorithm 2 satisfies (τ_i, ϵ_i)-PLDP. Next,

$$\begin{aligned}\mathbb{E}(\widetilde{w}_i) &= \mathbb{E}(\widetilde{v}_i + c_i) \\ &= \mathbb{E}(\widetilde{v}_i) + c_i \\ &= \frac{L_i(e^{\epsilon_i}+3)}{2(e^{\epsilon_i}-1)} \cdot \left[\frac{v_i(e^{\epsilon_i}-1)}{L_i(e^{\epsilon_i}+2)} + \frac{(e^{\epsilon_i}+1)}{2(e^{\epsilon_i}+2)}\right] - \\ &\frac{L_i(e^{\epsilon_i}+1)}{e^{\epsilon_i}-1} \cdot \left[\frac{e^{\epsilon_i}+3}{4(e^{\epsilon_i}+2)} - \frac{v_i(e^{\epsilon_i}-1)}{2L_i(e^{\epsilon_i}+2)}\right] + c_i \\ &= v_i + c_i \\ &= w_i.\end{aligned} \tag{12}$$

Moreover, for the mean value of the perturbed data,

$$\begin{aligned}\mathbb{E}(\overline{\mathcal{M}(w)}) &= \mathbb{E}\left(\frac{1}{\gamma N}\sum_{i=1}^{\gamma N}\widetilde{w}_i\right) \\ &= \frac{1}{\gamma N}\sum_{i=1}^{\gamma N}\mathbb{E}(\widetilde{w}_i) = \overline{w}.\end{aligned} \tag{13}$$

where $\overline{\mathcal{M}(w)} = \widetilde{w}$, and then,

$$\begin{aligned}\mathrm{Var}[\widetilde{w}_i] &= \mathrm{Var}[\widetilde{v}_i + c_i] \\ &= \mathrm{Var}[\widetilde{v}_i] \\ &= \mathbb{E}\left(\widetilde{v}_i^2\right) - (\mathbb{E}(\widetilde{v}_i))^2 \\ &\leq \frac{(e^{\epsilon_i}+1)(e^{\epsilon_i}+3)}{L_i(e^{\epsilon_i}-1)^2}.\end{aligned} \tag{14}$$

□

6. Performance Evaluation

In this section, we evaluate the practicality and performance of the scheme PLDP-FL through experiments. We will focus on the impact of privacy parameters on the performance of the scheme and also compare the functionality of the proposed scheme.

6.1. Experimental Setup

6.1.1. Datasets

To evaluate the scheme's practicality more reasonably and comprehensively, we will assess it on synthetic and real datasets. For the synthetic dataset, we randomly generate 100 data records within the safe region selected by the participants, and conduct performance tests on their private data set. Real data sets for experiments are provided by the MNIST and Fashion-MNIST datasets. The MNIST dataset is a handwritten digital image classification data set, containing ten target categories, with 60,000 training samples and 10,000 test samples. The scale, format, and division of the training and test set in Fashion-MNIST are the same as those in MNIST. In contrast to the MNIST dataset, the Fashion-MNIST dataset comprises images depicting ten distinct types of clothing. A convolutional neural network (CNN) is employed for model training, which consists of one input layer, two convolution layers, two max pooling layers and two fully connected layers. The local iteration period is set to 5, the batch size of each iteration is 20, the learning rate is 0.01, and the proportion of participants in each round of training is 0.7.

6.1.2. Metrics

For the synthetic dataset, the relative error (RE) is employed as the evaluation standard of the scheme, and the relative error between the estimated value and the true value is calculated using the following formula.

$$RE = \frac{|T_v - E_v|}{|T_v|}, \tag{15}$$

where RE denotes relative error, T_v is true value, E_v is estimated value. In the context of the MNIST dataset, the most commonly used measure of success is accuracy rate. Our goal is to attain a higher model accuracy while ensuring personalized privacy.

6.1.3. Privacy Parameters

The privacy parameter τ offers personalized privacy protection to local clients, with its safe region representing the range that attackers cannot distinguish from the original data. Thus, the larger the safe region, the more difficult it is to distinguish attackers from the raw data, although it can significantly affect performance. To assess the effect of privacy parameter τ on the scheme, we construct a different safe region size set $L_\tau = \{0.2, 0.4, \ldots, 1.2, \ldots, 2.0\}$, each client can choose a size for their safe region, and then adjust their safe region in accordance with the size of the safe region τ_i. For example, if a client chooses a safe region size of 1.2, they can set the safe region as $[-0.6, 0.6]$, $[-0.2, 1.0]$, etc. To accurately evaluate the effect of the safe region on the scheme's performance, three modes were established: $\tau_1 = \{0.2, 0.4, \ldots, 0.2, 0.4\}$, $\tau_2 = \{0.2, 0.4, \ldots, 1.8, 2.0, 0.2, \ldots\}$, $\tau_3 = \{1.8, 2.0, \ldots, 1.8, 2.0\}$, where τ_1 being the smallest, τ_2 being the next smallest, and τ_3 being the largest.

The privacy parameter ϵ also provides personalized privacy protection to the clients, where its size indicates the level of indistinguishability from the adversary. The smaller the ϵ, the more robust the privacy protection, leading to a greater impact on performance. Based on our experience, participants are given the option to choose their privacy budget ϵ_i from the selection set $E_\epsilon = \{0.1, 0.2, \ldots, 0.9, 1.0\}$. To accurately assess its effect on the scheme's performance, also set three modes with $\epsilon_1 = \{0.1, 0.2, \ldots, 0.1, 0.2\}$, $\epsilon_2 = \{0.1, 0.2, \ldots, 0.9, 1.0, 0.1, \ldots\}$, $\epsilon_3 = \{0.9, 1.0, \ldots, 0.9, 1.0\}$, ϵ_1 opts for the least privacy budget for the participants, ϵ_2 chooses the privacy budget in ascending order, and ϵ_3 selects the highest privacy budget for the participants. We first evaluate the effect of each mode on the model performance, and then assess the effect of their various combinations, such as $\epsilon_1\_\tau_1, \ldots, \epsilon_3\_\tau_3$.

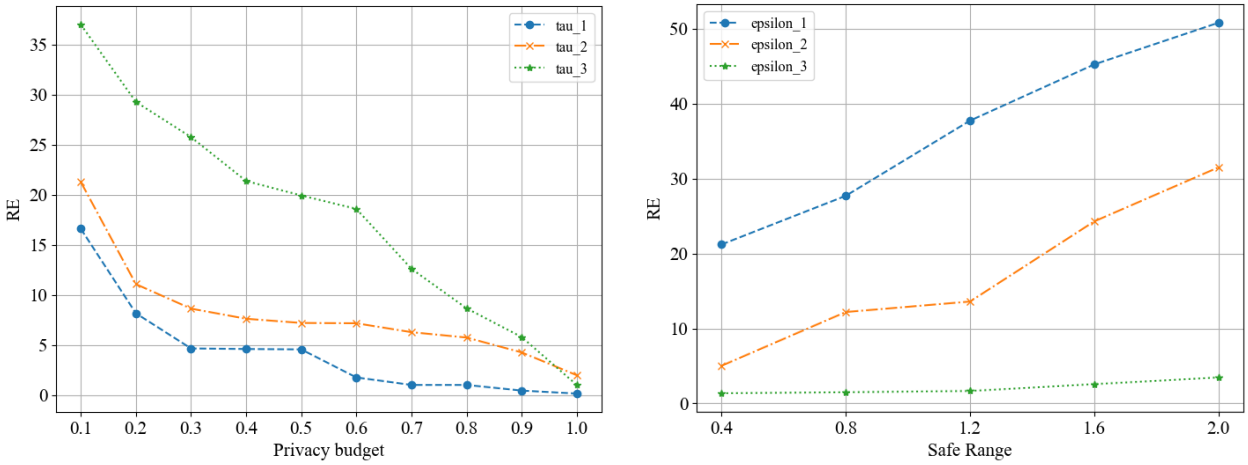
6.1.4. Environment

All of our experiments were performed using Python programming language and on a local server (32 Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz, 156GB RAM, 2TB, Ubuntu 20.04.3, cuda 11.2, GPU). All test results are obtained by taking the average value after running several times.

6.2. Experimental Results

6.2.1. Impact of Privacy Parameters on Synthetic Dataset

For this experiment, 70 participants were used to measure the relative error of the scheme under varying privacy budgets and security intervals. The results of this experiment are illustrated in Figure 2. Figure 2a illustrates the alteration of relative error according to the variation of security interval modes when the privacy budget is changed, while Figure 2b displays the change of relative error in response to the alteration of privacy budget modes when the safe region size is changing. As depicted in Figure 2a, the relative error of the scheme decreases with the increase of the privacy budget. Because a larger privacy budget leads to smaller noise introduced by the local differential privacy algorithm, thus providing more accurate results, which corroborates our prior theoretical analysis. Simultaneously, it is evident from the three security interval mode curves that the larger the security interval, the more significant the effect on the scheme performance, leading to a greater error in the result. As depicted in Figure 2b, the relative error increases with the safe region. However, ϵ_3 with a larger privacy budget produces a smaller relative error than ϵ_1 with a smaller privacy budget.



(**a**) Privacy budget vs. Relative error

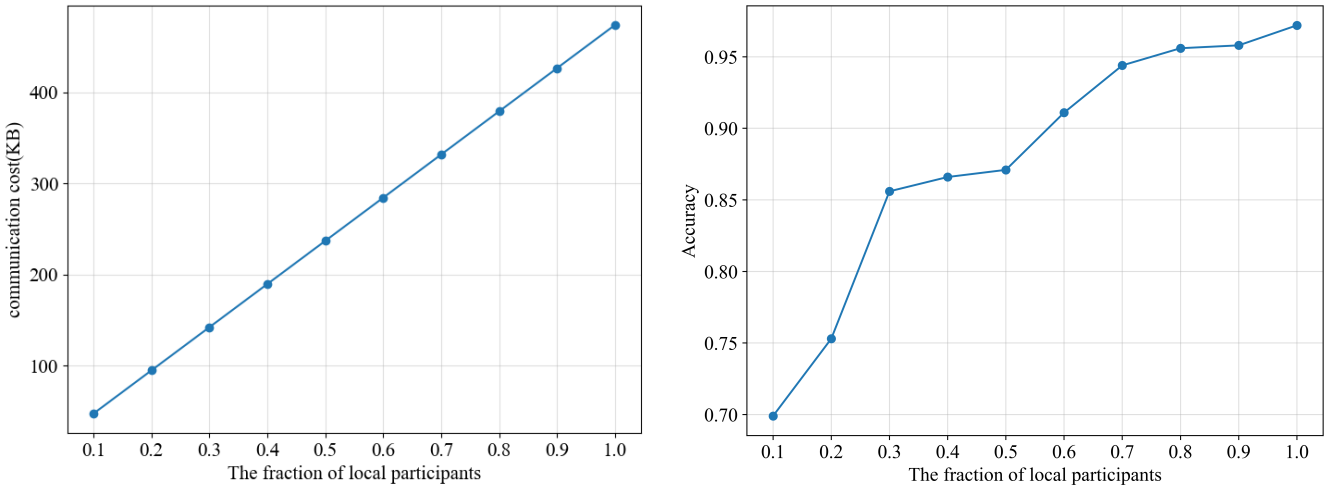
(**b**) Safe region vs. Relative error

Figure 2. Relative error of the scheme under different privacy parameters.

6.2.2. Impact of privacy parameters on MNIST and Fashion-MNIST

For the image classification model training experiment of federated learning, we have set 100 local participants and the participation rate for each round of the training process is 0.7. The convolution core is $10 * 5 * 5 * 32$, and the first fully connected layer boasts an input channel of 320 and an output channel of 50, while the second fully connected layer has an input channel of 50 and an output channel of 10. It can be observed that the communication cost increases linearly with the increase of the local participant's proportion from Figure 3a, and Figure 3b illustrates the effect of varying the number of participants on the model's accuracy. It is evident that the accuracy of the model increases with the number of local participants, due to the fact that a greater amount of intermediate parameters can be shared, thus resulting in a more precise trained model. Figure 3b demonstrates that selecting participants with a ratio of 0.7 or more has a relatively minor impact on the

model's accuracy. To reduce communication costs and maintain accuracy, only 70% of the participants will be selected for training in each round, instead of all participants.



(**a**) Local client ratio vs. Communication cost

(**b**) Local client ratio vs. Accuracy

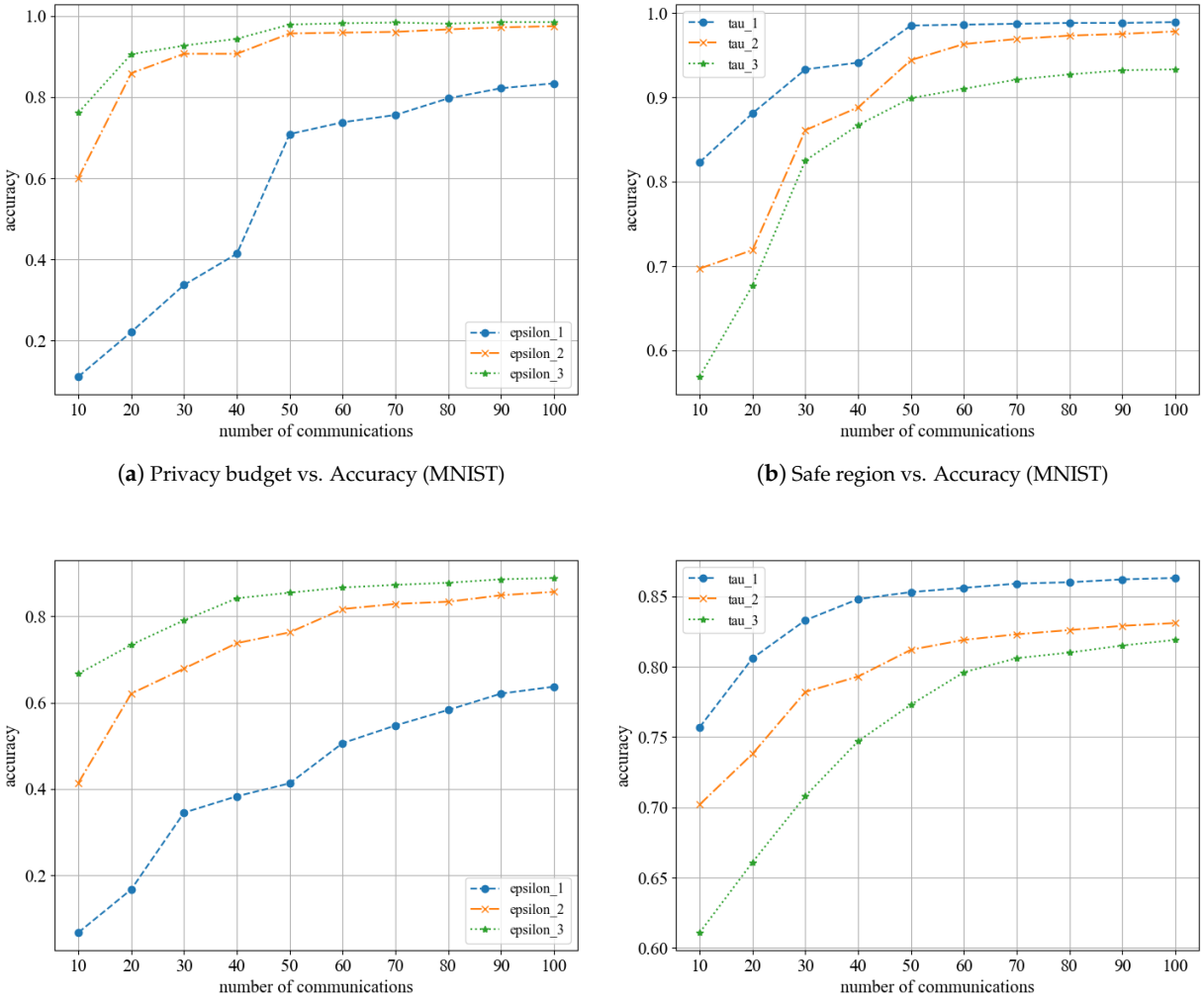
Figure 3. Performance of the scheme under different local client ratio.

We conducted experiments to observe the effect of privacy parameters on the model's accuracy in various settings. As illustrated in Figure 4, the accuracy of the model increases with the number of communication rounds. As is evident from Figure 4a,c, when a small privacy budget is chosen, the local differential perturbation algorithm introduces a significant amount of noise to the parameters, resulting in a bad aggregation outcome and a comparatively low accuracy of the final model. In contrast, the model accuracy is notably improved when a larger privacy budget is chosen. These two selection methods are excessively rigorous and do not accurately reflect the personalized privacy requirements of each participant. The ϵ_2 mode allows for selecting a privacy budget that can well reflect the individual's privacy needs, and its final model accuracy is similar to that of the ϵ_3 mode. From Figure 4b,d, it is clear that a smaller security interval is set, the accuracy of the resulting model is greater, while a larger safe region yields a lower accuracy. Participants can still obtain a reasonably high model accuracy even with different safe regions. In conclusion, the experiments conducted with different privacy parameters demonstrate that a satisfactory model can be obtained by using personalized privacy parameters with minimal loss.

6.2.3. Impact of Different Privacy Parameters Combinations on MNIST and Fashion-MNIST

We further tested the accuracy of the model when combining different privacy parameters. Each participant can set two privacy parameters according to personalized local differential privacy. The results of three privacy budget selection modes and three safe region selection modes are presented in Figure 5a–f, respectively. Figure 5a,b demonstrate that the combination under ϵ_1 mode offers robust privacy protection, though at the cost of low accuracy. Pursuing high-intensity privacy protection, the parameter combination form of ϵ_1_τ_1 can still produce a model with relatively high accuracy after a certain amount of training. As depicted in Figure 5c,e following 50 rounds of training on the MNIST dataset, the accuracy under the combination of modes ϵ_2 and ϵ_3 can reach 95.7% and 97.9%, respectively. In contrast, it can be seen from Figure 5d,f the highest accuracy achieved by the same combination on the Fashion-MNIST dataset was 79.2% and 85.6%, respectively. Furthermore, the accuracy of the model decreases as the safe region increases, which agrees with the outcomes of the earlier separate examination of privacy parameters. If the data

sensitivity of each participant is not high, the $\epsilon_3\_\tau_1$ parameter combination form can be utilized in order to obtain a model with high accuracy. In case of varying data sensitivities among participants, the parameter combination in ϵ_2 mode should be applied, as it offers improved accuracy compared to ϵ_1 mode and a reduced accuracy loss compared to ϵ_3 mode. Our scheme can produce a model with a high degree of accuracy while still maintaining personalized privacy parameters.



(**a**) Privacy budget vs. Accuracy (MNIST)

(**b**) Safe region vs. Accuracy (MNIST)

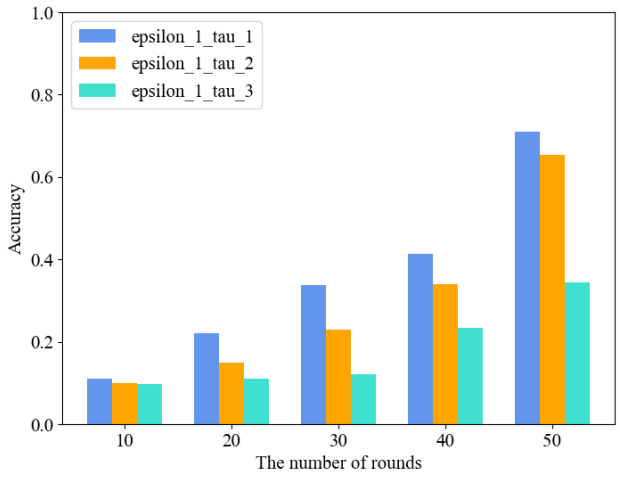
(**c**) Privacy budget vs. Accuracy (Fashion-MNIST)

(**d**) Safe region vs. Accuracy on (Fashion-MNIST)

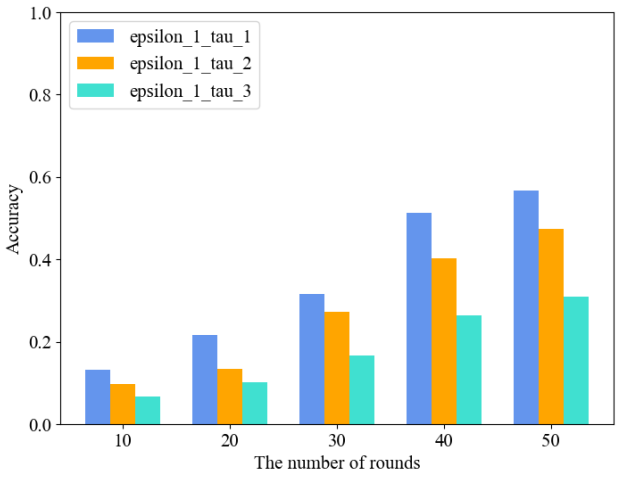
Figure 4. Performance of the scheme under different parameters.

6.2.4. Comparison of Scheme PLDP-FL against the Existing Approaches

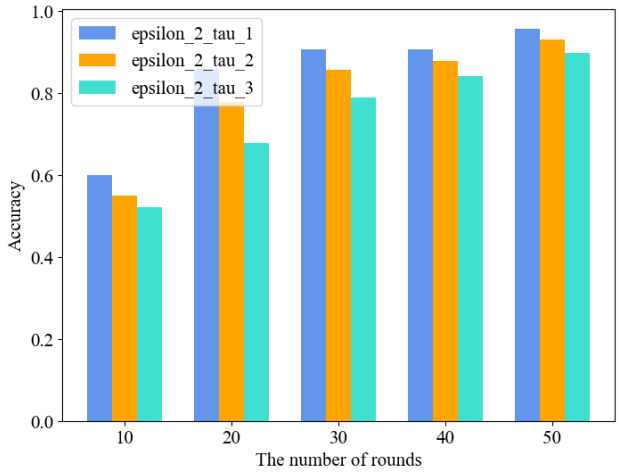
The results of our comparison are shown in Figure 6. We compare our scheme with the first, and only federal learning scheme PLU-FedOA [34], which is based on personalized local differential privacy. However, their personalized local differential privacy implementation algorithm differs from ours, as it only has one privacy parameter ϵ. To ensure a fair comparison, the experiment is conducted on MNIST dataseet, and fixed the safe region of τ for each client to $[-1, 1]$ and then compared the performance of its case2 scenario with our ϵ_2 mode. The scheme FedAvg [3] can be used as an upper bound for our performance comparison. From Figure 6, we can see that after 100 rounds of training, our scheme has essentially converged, with the accuracy rate of the model reaching 97.5%, which is equivalent to the accuracy rate under non-private conditions. In contrast, the PLU-FedOA scheme only achieved 90.3% accuracy rate. It can be concluded that our scheme is more competitive in terms of model performance.



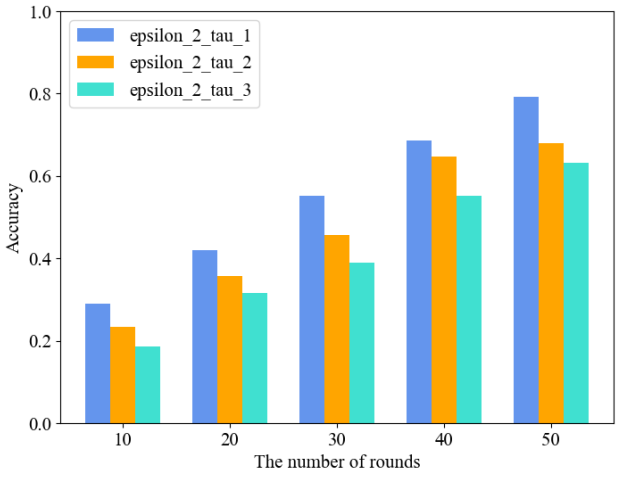
(**a**) ϵ_1 vs. Accuracy (MNIST)



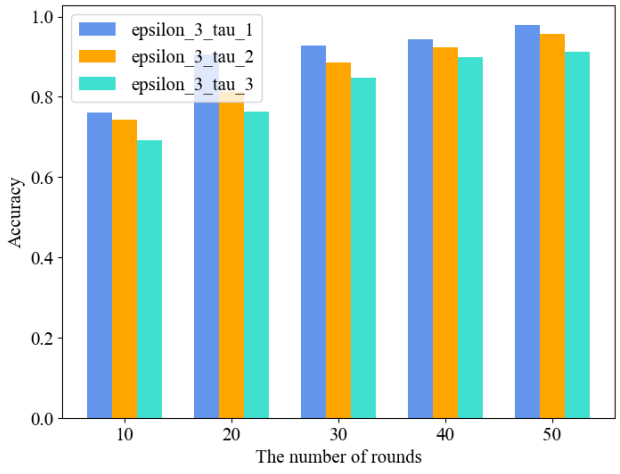
(**b**) ϵ_1 vs. Accuracy (Fashion-MNIST)



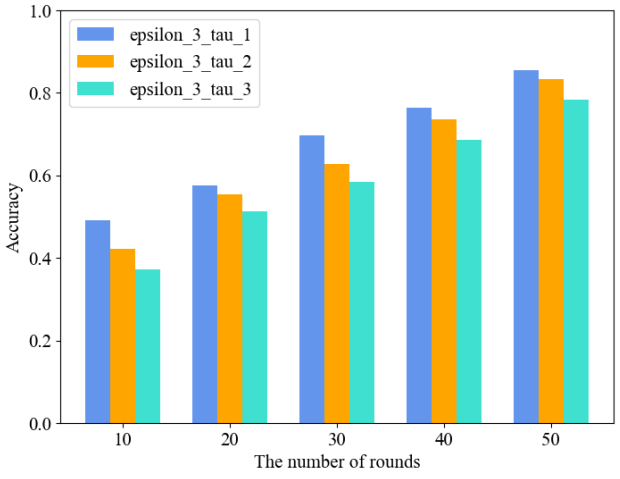
(**c**) ϵ_2 vs. Accuracy (MNIST)



(**d**) ϵ_2 vs. Accuracy (Fashion-MNIST)



(**e**) ϵ_3 vs. Accuracy (MNIST)



(**f**) ϵ_3 vs. Accuracy (Fashion-MNIST)

Figure 5. Classification accuracy of scheme PLDP-FL on image dataset.

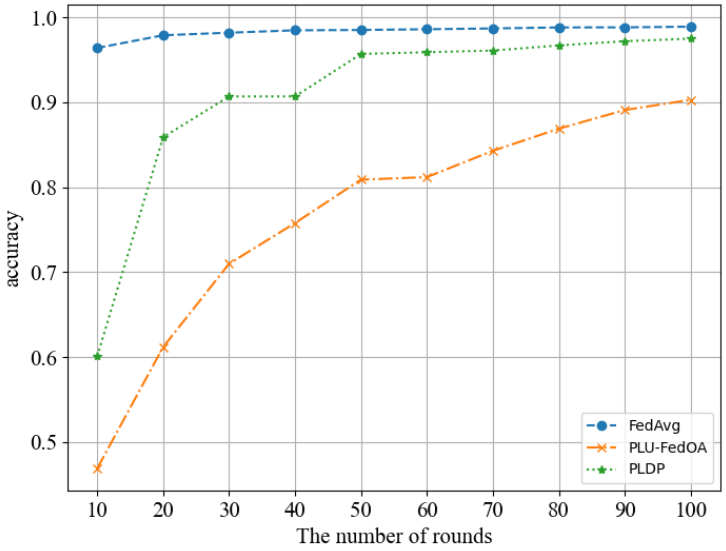


Figure 6. Comparison of scheme PLDP-FL against the existing approaches.

6.3. Functionality Comparison

This section compares the functions implemented by the PLDP-FL scheme with the relevant representative work, and the results are displayed in Table 2. This comparison work focuses on the scheme's security, such as parameter privacy, model privacy and whether it can prevent inference attacks. It also looks into the practicability of the scheme, such as the size of the actual privacy budget parameter when differential privacy is applied. Furthermore, it assesses whether the scheme supports personalized privacy protection of local clients. The reference [3] introduced the federated learning method of model aggregation, yet this method fails to provide privacy protection for the data during the training process, thus making it insecure. The reference [19] produced a new algorithm to enhance privacy loss by utilizing global differential privacy technology with a random gradient descent algorithm. Unfortunately, this technique is not compatible with distributed learning scenarios. The scheme [12] introduced a new federated learning system using CLDP, a variant technology of local differential privacy, which addresses the problem that the existing local differential privacy protocol is unsuitable for high-dimensional data. However, the actual corresponding privacy budget in the scheme is large, resulting in a weak privacy protection effect, making it unfeasible. Additionally, the local participants in the scheme cannot gain personalized privacy protection. A noise-free differential privacy mechanism for the federated model distillation framework was proposed in [26], which can solve the training data leakage caused by model prediction. However, it fails to provide personalized privacy protection to the local participants. Our proposed approach commences by addressing participants' individualized privacy protection requirements, constructing a personalized local differential privacy algorithm that satisfies these requirements, and training superior models while providing personalized privacy protection for participants.

Table 2. Functionality Comparison.

Scheme	Techniques	Parameter Privacy	Model Privacy	Practical	Protection against Inference Attacks	Personalized Privacy Protection
FedAvg [3]	FL	×	×	-	×	-
DPSGD [19]	DP	✓	✓	✓	✓	-
LDP-Fed [12]	LDP	✓	✓	×	✓	×
FedMD-NFDP [26]	NFDP	✓	✓	✓	✓	×
PLU-FedOA [34]	PLDP	✓	✓	×	✓	✓
PLDP-FL	PLDP	✓	✓	✓	✓	✓

7. Conclusions

In this paper, we present a personalized local differential privacy federated learning scheme, to overcome the limitations of the existing local differential privacy-based federated learning scheme. This scheme has developed a perturbation algorithm PDPM, to satisfy the personalized local differential privacy needs of local participants. The PDMP algorithm enables each participant to decide their own privacy parameter pair (τ_i, ϵ_i), instead of the fixed privacy parameter ϵ, thus satisfying the personalized privacy protection requirements in the federated learning process. Numerous experiments have demonstrated that clients can adjust their privacy parameters while still obtaining a high-accuracy model. This paper only focuses on the personalized privacy protection needs of the client. In fact, every participant's data may contain multiple attributes, and each attribute's privacy sensitivity might differ. and the value range of each attribute may contain multiple values, and the privacy sensitivity of each value may also differ. In the future, we intend to design solutions that address personalized privacy protection needs from multiple angles, such as the attributes and values of the client.

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

Funding: This research was funded by the National Natural Science Foundation of China (Grant Nos. 62272362, U19B2021), the Open Research Fund of Key Laboratory of Cryptography of Zhejiang Province (ZCL21016), the Fundamental Research Funds for the Central Universities (XJS220122).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Abbreviations

The following abbreviations are used in this manuscript:

FL	federated learning
LDP	local differential privacy
PLDP	personalized local differential privacy
PDPM	perturbation algorithm that satisfies personalized local differential privacy
AI	artificial intelligence
MI	machine learning
SMC	secure multi-party computation
HE	homomorphic encryption
DP	Differential privacy
GDP	global differential privacy
FedAvg	Federated Average Algorithm
SGD	stochastic gradient descent algorithm
PLDP-FL	Federated Learning with Personalized Local Differential Privacy

References

1. Yang, Q.; Liu, Y.; Cheng, Y.; Kang, Y.; Chen, T.; Yu, H. Federated learning. *Synth. Lect. Artif. Intell. Mach. Learn.* **2019**, *13*, 1–207.
2. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
3. McMahan, H.B.; Moore, E.; Ramage, D.; y Arcas, B.A. Federated Learning of Deep Networks using Model Averaging. *arXiv* **2016**, arXiv:1602.05629.

4. Phong, L.T.; Aono, Y.; Hayashi, T.; Wang, L.; Moriai, S. Privacy-preserving deep learning: Revisited and enhanced. In Proceedings of the International Conference on Applications and Techniques in Information Security, Auckland, New Zealand, 6–7 July 2017; pp. 100–110.
5. 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 Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 739–753. [CrossRef]
6. Fredrikson, M.; Jha, S.; Ristenpart, T. Model Inversion Attacks That Exploit Confidence Information and Basic Countermeasures. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1322–1333.
7. Tramèr, F.; Zhang, F.; Juels, A.; Reiter, M.K.; Ristenpart, T. Stealing Machine Learning Models via Prediction APIs. In Proceedings of the 25th USENIX Security Symposium (USENIX Security 16), Austin, TX, USA, 10–12 August 2016; pp. 601–618.
8. Truex, S.; Baracaldo, N.; Anwar, A.; Steinke, T.; Ludwig, H.; Zhang, R.; Zhou, Y. A hybrid approach to privacy-preserving federated learning. In Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, London, UK, 15 November 2019; pp. 1–11.
9. Xu, G.; Li, H.; Zhang, Y.; Xu, S.; Ning, J.; Deng, R. Privacy-preserving federated deep learning with irregular users. *IEEE Trans. Dependable Secur. Comput.* 2022, *19*, 1364–1381. [CrossRef]
10. Dwork, C.; Roth, A. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* **2014**, *9*, 211–407. [CrossRef]
11. Geyer, R.C.; Klein, T.; Nabi, M. Differentially Private Federated Learning: A Client Level Perspective. *arXiv* **2017**, arXiv:1712.07557.
12. Truex, S.; Liu, L.; Chow, K.H.; Gursoy, M.E.; Wei, W. LDP-Fed: Federated learning with local differential privacy. In Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking, Heraklion, Greece, 27 April 2020; pp. 61–66.
13. Zhao, Y.; Zhao, J.; Yang, M.; Wang, T.; Wang, N.; Lyu, L.; Niyato, D.; Lam, K.Y. Local differential privacy-based federated learning for internet of things. *IEEE Internet Things J.* **2020**, *8*, 8836–8853. [CrossRef]
14. 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 (IJCAI), Montreal, QC, Canada, 19–27 August 2021; pp. 1571–1578. [CrossRef]
15. Phong, L.T.; Aono, Y.; Hayashi, T.; Wang, L.; Moriai, S. Privacy-Preserving Deep Learning via Additively Homomorphic Encryption. *IEEE Trans. Inf. Forensics Secur.* 2018, *13*, 1333–1345. [CrossRef]
16. Xu, R.; Baracaldo, N.; Zhou, Y.; Anwar, A.; Ludwig, H. Hybridalpha: An efficient approach for privacy-preserving federated learning. In Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, London, UK, 15 November 2019; pp. 13–23.
17. Chen, Y.; Wang, B.; Zhang, Z. PDLHR: Privacy-Preserving Deep Learning Model with Homomorphic Re-Encryption in Robot System. *IEEE Syst. J.* 2022, *16*, 2032–2043. [CrossRef]
18. Shokri, R.; Shmatikov, V. Privacy-preserving deep learning. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1310–1321.
19. 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, Vienna, Austria, 24–28 October 2016; pp. 308–318.
20. Hu, R.; Guo, Y.; Li, H.; Pei, Q.; Gong, Y. Personalized federated learning with differential privacy. *IEEE Internet Things J.* **2020**, *7*, 9530–9539. [CrossRef]
21. Wei, K.; Li, J.; Ding, M.; Ma, C.; Yang, H.H.; Farokhi, F.; Jin, S.; Quek, T.Q.; Poor, H.V. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3454–3469. [CrossRef]
22. Zhao, L.; Wang, Q.; Zou, Q.; Zhang, Y.; Chen, Y. Privacy-preserving collaborative deep learning with unreliable participants. *IEEE Trans. Inf. Forensics Secur.* **2019**, *15*, 1486–1500. [CrossRef]
23. Phan, N.; Wang, Y.; Wu, X.; Dou, D. Differential privacy preservation for deep auto-encoders: An application of human behavior prediction. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
24. Duchi, J.C.; Jordan, M.I.; Wainwright, M.J. Minimax optimal procedures for locally private estimation. *J. Am. Stat. Assoc.* **2018**, *113*, 182–201. [CrossRef]
25. Liu, R.; Cao, Y.; Yoshikawa, M.; Chen, H. Fedsel: Federated sgd under local differential privacy with top-k dimension selection. In Proceedings of the International Conference on Database Systems for Advanced Applications, Jeju, Republic of Korea, 24–27 September 2020; pp. 485–501.
26. Sun, L.; Lyu, L. Federated Model Distillation with Noise-Free Differential Privacy. In Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI), Montreal, QC, Canada, 19–27 August 2021; pp. 1563–1570. [CrossRef]
27. Arachchige, P.C.M.; Bertok, P.; Khalil, I.; Liu, D.; Camtepe, S.; Atiquzzaman, M. Local differential privacy for deep learning. *IEEE Internet Things J.* **2019**, *7*, 5827–5842. [CrossRef]
28. Damgård, I.; Jurik, M. A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In Proceedings of the Public Key Cryptography: 4th International Workshop on Practice and Theory in Public Key Cryptosystems, Cheju Island, Republic of Korea, 13–15 February 2001; pp. 119–136.

29. Nie, Y.; Yang, W.; Huang, L.; Xie, X.; Zhao, Z.; Wang, S. A Utility-Optimized Framework for Personalized Private Histogram Estimation. *IEEE Trans. Knowl. Data Eng.* **2019**, *31*, 655–669. [CrossRef]
30. Shen, Z.; Xia, Z.; Yu, P. PLDP: Personalized Local Differential Privacy for Multidimensional Data Aggregation. *Secur. Commun. Netw.* **2021**, *2021*, 6684179:1–6684179:13. [CrossRef]
31. Xue, Q.; Zhu, Y.; Wang, J. Mean estimation over numeric data with personalized local differential privacy. *Front. Comput. Sci.* **2022**, *16*, 163806. [CrossRef]
32. Akter, M.; Hashem, T. Computing Aggregates Over Numeric Data with Personalized Local Differential Privacy. In Proceedings of the Information Security and Privacy—22nd Australasian Conference (ACISP 2017), Auckland, New Zealand, 3–5 July 2017; Volume 10343, pp. 249–260. ._14. [CrossRef]
33. Li, X.; Yan, H.; Cheng, Z.; Sun, W.; Li, H. Protecting Regression Models with Personalized Local Differential Privacy. *IEEE Trans. Dependable Secur. Comput.* 2022. [CrossRef]
34. Yang, G.; Wang, S.; Wang, H. Federated Learning with Personalized Local Differential Privacy. In Proceedings of the 6th IEEE International Conference on Computer and Communication Systems (ICCCS 2021), Chengdu, China, 23–26 April 2021; pp. 484–489. [CrossRef]
35. Dwork, C.; McSherry, F.; Nissim, K.; Smith, A. Calibrating noise to sensitivity in private data analysis. In Proceedings of the Theory of Cryptography Conference, New York, NY, USA, 4–7 March 2006; pp. 265–284.
36. Kasiviswanathan, S.P.; Lee, H.K.; Nissim, K.; Raskhodnikova, S.; Smith, A.D. What Can We Learn Privately? *SIAM J. Comput.* **2011**, *40*, 793–826. [CrossRef]
37. Chen, R.; Li, H.; Qin, A.K.; Kasiviswanathan, S.P.; Jin, H. Private spatial data aggregation in the local setting. In Proceedings of the 32nd IEEE International Conference on Data Engineering (ICDE 2016), Helsinki, Finland, 16–20 May 2016; . [CrossRef]
38. Warner, S.L. Randomized response: A survey technique for eliminating evasive answer bias. *J. Am. Stat. Assoc.* **1965**, *60*, 63–69. [CrossRef] [PubMed]
39. Kairouz, P.; Bonawitz, K.; Ramage, D. Discrete distribution estimation under local privacy. In Proceedings of the International Conference on Machine Learning, New York City, NY, USA, 19–24 June 2016; pp. 2436–2444.