Fed-CAD: Federated Learning with Correlation-aware Adaptive Local Differential Privacy

Bingzhu Zhu*, Shan Chang*, Guanghao Liang*, Hongzi Zhu†, Jie Xu*
*Donghua University, China
†Shanghai Jiao Tong University, China
2212500@mail.dhu.edu.cn, changshan@dhu.edu.cn, 2232843@mail.dhu.edu.cn,
hongzi@cs.sjtu.edu.cn, 2232820@mail.dhu.edu.cn

2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS) | 979-8-3503-5012-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/IWQoS61813.2024.10682944

***Abstract*—Federated Learning (FL) enables multiple participants to collaboratively train a globally shared model without the need of explicit data sharing. However, prior research indicates that local model updates released during the federated training may also jeopardize privacy of participants. To address this issue, local differential privacy (LDP) mechanism has been applied to FL systems. LDP provides privacy protection with rigorous mathematical proof by introducing random perturbations, e.g., Gaussian noise, to the released updates, however excessive noise compromises the utility of the updates. In this paper, we propose a novel Correlation-aware Adaptive LDP mechanism, Fed-CAD, for FL, which reduces the required scale of noise by leveraging the temporal correlation between consecutive local model updates belonging to the same participant, without increasing the privacy budgets (risks). We theoretically prove that Fed-CAD satisfies (ε, δ)-LDP as long as the difference between local models is smaller than the differential bound, and analyze the noise variance, a metric of utility. We implement Fed-CAD on image classification FL tasks. Experimental results demonstrate that Fed-CAD significantly outperforms the one-shot LDP baseline.**

***Index Terms*—federated learning, adaptive local differential privacy, Gaussian mechanism, correlation-aware**

I. Introduction

Federated Learning (FL), a framework that allows a large number of participants to collaborate in training a universal model without exposing their local data. During each federated training iteration, participants use their private data to train Machine Learning (ML) models locally and only submit the results of training, i.e., local model updates (model parameters or gradients), to a remote coordinator, i.e., a server. After receiving enough local updates, the server runs an aggregation protocol to calculate the current global model, accordingly. Although the private data of participants have never been shared, literature reports that the exposure of local updates may also cause the sensitive information of participants [1]–[3] being inferred by attacks, e.g., other participants in the same FL task or the central server [4] [5]. Recent studies further demonstrate that an attacker continuously collecting global models and updates from a victim participant is able to successfully reveal its hidden properties [6] or recover private training samples belonging to it [7] [8].

To providing participants with stronger data privacy, it is necessary to introduce data privacy protection techniques in FL, which can be divided into three categories: Homomorphic Encryption (HE), Secure Multi-party Computation (SMC) and Differential Privacy (DP). Compared to HE and SMC, DP requires no expensive encryption and decryption operations, and thus is more friendly to low-end devices which are fairly common in cross-device FL applications. DP protects the data privacy by introducing randomized perturbations. Roughly speaking, it is hardly, if not impossible, to accurately infer the original values of data after being perturbed with DP. Meanwhile, DP allows statistical analysis on randomized data, ensuring the utility of the data. More importantly, DP provides mathematical guarantee on the risk of privacy leakage, thus has become one of the most popular privacy protection technology in recent years. Local updates in plain-text, even being randomized, allows the server to conduct online validation on them, providing chances to identify those poisoned local updates from internal attackers. Consequently, applying DP to FL has received widespread attention in both the industry and academia.

To employ DP in FL, a practical solution should meet the following three requirements: 1) *Strong privacy*: according to the definition of DP, privacy budget (or privacy risk) is inversely proportional to the scale of perturbations. It is necessary to add sufficient perturbations, i.e., introducing enough randomness, to the data being protected so as to minimize privacy risks. 2) *High utility*: it is desired that the federated model is not affected by the perturbations introduced, in terms of convergence and accuracy, as much as possible. 3) *Low cost*: to encourage the participation of low-end devices in FL, it is essential for such a scheme to be light-weight on computing and storage.

In the literature, R. C. Geyer *et al.,* approximate the federated model, i.e., the average of local models, with a Gaussian-based DP mechanism to hide the contribution from a single participant within the entire FL procedure [9]. In other words, a federated model does not reveal whether a participant takes part in FL or not. Such centralized DP (CDP) mechanisms assume that the server is trustworthy. The local models submitted by participants are directly acquired by the server without being processed by privacy protection mechanisms. Unfortunately, it is difficult to ensure the trustworthiness of a server in practice. An honest-but-curious server may infer privacy of participants from their local updates.

Contrastively, a local DP (LDP) mechanism directly applies perturbations to local updates, which provides participant-level privacy in the case of the server is untrusted. The LDP mechanism ensures that the federated model aggregated from perturbed local models is an unbiased estimate of its original version, ensuring the accuracy of it [10]. However,

Shan Chang is corresponding author.

achieving the best tradeoff between utility and privacy under LDP mechanisms is very challenging. Large perturbations provide strong protection on local updates, however inducing large deviations between the randomized federated models and the original ones. The larger the perturbations, the greater the deviations, damaging the convergence as well as performance of the federated model. On the contrary, if the perturbations are too small, although ensuring utility, insufficient protection places participants at risk of privacy disclosure. Some research work attempts to improve utility by only disturbing relatively important values. For example, R. Liu *et al.,* propose FedSel, which selects top-*k* dimensions of gradients to apply LDP [11]. However, a small portion of gradients can leak a considerable amount of sensitive information about local data [12], even leading to the leakage of original data [7]. R. Shokri *et al.,* [13] introduce a distributed selective SGD (DSSGD) algorithm, where a fraction of parameters are selected for adding noises and uploading in each iteration. DSSGD offers an attractive trade-off between the utility and privacy through parameter selection, however slows down convergence due to that unselected parameters are treated as *zero* and their updates are delayed. J. Liu *et al.,* [14] present PFA, where local models are projected to a low-rank space before adding noise, however inducing expensive computation overhead on both the server and participants, especially when the model architecture is complex. As a result, to the best of our knowledge, none of existing work satisfies all requirements of a successful DP solution in FL.

In this work, we follow the architecture of Fed-LDP. For each iteration, the perturbation is added to each local model update, i.e., the difference between models before and after local training. However, we make remarkable modifications to the original Fed-LDP. We capture the inherent temporal autocorrelations between the local model updates, which have never been utilized in existing LDP-based FL solutions, so as to enhance utilities of perturbed updates. We conduct empirical experiments to prove that the L_2 Norms of differences between two consecutive local updates are significantly smaller than that of the updates themselves, referred as *Strong Autocorrelation*. Moreover, as the federated model converges, the difference will further decrease. We propose a Correlation-aware Adaptive LDP mechanism in FL, named Fed-CAD, which is composed of two critical components. First, we utilize a bound to clip the excessive change between model updates, and employ Correlated Gaussian Mechanism (CGM) [15] instead of general Gaussian Mechanism to generate temporally correlated perturbations for them. For each participant, the noise injected in a local model update is the linear combination of a fresh Gaussian noise and a portion of the noise injected to the last local model update. In this way, due to the fact that consecutive local model updates of the same participant are not independent, the randomness accumulated in the updates will be partially cancelled out, leading to a smaller variance of noise and better utility, i.e., smaller mean square error of the perturbed updates. Second, we propose an algorithm to adjust the clipping bound as well as the noise scale of model updates for each iteration adaptively, according to the privacy budget and the difference between model updates. It helps to fine-grained determine the noise scale, and thus further improves utility. We mathematically prove that Fed-CAD satisfies (ε, δ)-LDP, and analyze its expected utility gains compared to the baseline scheme, i.e., repeatedly applying a one-shot Gaussian noise in each iteration. Extensive experiments confirm the significant advantage of Fed-CAD in terms of the model accuracy and convergence.

II. Related Work

A. Federated learning with Central Differential Privacy

CDP is initially designed for centralized scenarios. In such scenarios, a trusted database server can clearly see all training samples of participants and answer queries through randomized results. R. C. Geyer *et al.,* [16] apply CDP on FL and make two improvements on it: 1) In each iteration, the server selects a subset of participants submitting local updates in plain-text. 2) A central server is responsible for estimating and perturbing the global model according to the aggregation protocol and DP mechanism, respectively. The proposed CDP-based scheme can provide participant-level DP. In other words, the contribution of individual participant can be hidden, and an internal attacker cannot determine whether a particular participant joins in the distributed training in an iteration or not. McMahan *et al.,* [4] introduce moment accountant to accurately compute and control the privacy loss, and provide flat and layer-wise clipping strategies for deep network structures. They also design two estimators based on different sensitivities to ensure the accuracy of the model.

Since the noise is added directly to the global update, the CDP-based global model works best under the same privacy conditions. However, it requires a large number of participants to ensure the utility of the model, and global model convergence is problematic when the number of participants is small, making it unsuitable for horizontal FL systems with a relatively small number of participants. At the same time, the assumption of a trusted server in CDP is overly idealized in many scenarios, as it constitutes a single point of failure for the central server, and the entire privacy-preserving mechanism may be threatened if the server fails or if the participant-server communication process suffers an attack. In scenarios of untrusted centralized server, which is much more realistic, LDP is used to protect personal privacy.

B. Federated Learning with Local Differential Privacy

Compared with CDP, LDP provides stronger privacy guarantee. The participants in this scenario do not trust the central server, so they choose to inject noise into their data locally, and then send the noisy data to the central server or other collaborators, and the server only owns the noisy version data, and all the subsequent operations are carried out based on the noisy data, which effectively reduces the burden of protecting private data.

Shokri *et al.,* [13] first apply LDP to distributed machine learning, where each participant uploads a small subset of locally updated parameters whose changes exceed a predetermined threshold and add Gaussian or Laplace noise before sending them to a central server, thus ensuring LDP. Bhowmick *et al.,* [1] consider a more realistic attack scenario by limiting the prior knowledge of the adversary and propose a local differential privacy training method for a large-scale model, which effectively guards against joint learning reconfiguration attacks at the cost of a higher number of iterations. Truex *et al.,* [17] extend the exponential mechanisms EM and α-CLDP to LDP-based FL, which helps to handle high-dimensional, continuous model parameter updating.

Methods such as randomized response and some other typical LDP mechanisms (e.g., OUE [18] and PM [19]) can also improve the privacy-preserving performance of FL. Chamikara *et al.,* [20] split the neural network into two parts, where each participant locally trains a convolution neural network whose last layer is a fully connected layer, perturbs the output of the last layer using the RAPPOR [21] stochastic response algorithm, and uploads the perturbed output to a central server to complete the subsequent training process.

To further reduce the variance of the perturbed global model, existing studies utilize various privacy amplification techniques (e.g., sub-sampling and shuffling) to reduce model expectation variance and bypass the curse of high-dimension parameters in deep learning models. The LDP-FL [22] approach imparts anonymity during local model updating through shuffling technique, which breaks the direct link between the central server and a particular participant, greatly enhancing the privacy of the entire framework.

More work investigates other relevant metrics for LDP-based FL, such as communication overhead, convergence efficiency, etc LDP. In addition, Naseri *et al.,* [10] also evaluate the impact of LDP on the relationship between privacy and Byzantine robustness in FL.

Since the perturbations are executed independently by each participant without the knowledge of training data of other participants, the added noise is independent of each other, which limits the scope of application of LDP. In particular, for cases involving a large number of participants and a large size model, without fine-grained calibration, the diminution of utility introduced by LDP is unacceptable.

III. Preliminaries

A. FL System

An FL system consists of a server and N participants. D_i denotes the private dataset of the ith participant P_i, $i \in \{1, 2, \cdots, N\}$, D represents the sum of all participants' private datasets, i.e., $D = \sum_{i=1}^{N} D_i$. The goal of FL is to enable individual participants to collaboratively train a neural network model that is nearly equivalent to a centralized machine learning model.

In FL, the server is responsible for saving, aggregating and distributing the global model w_{global}, by minimizing the global objective function $Func(w)$, which results in the optimal global model w^*.

$$w^* = \arg\max_{w} \mathrm{Func}(w)$$

FL progressively improves the global model through continuous iterative training, and all participants optimally update the global model based on the local privacy dataset D_i.

$$\mathrm{Func}(w) = \sum_{i=1}^{N} \frac{|D_i|}{|D|} \mathrm{Func}_i(w)$$

Taking the t-th iteration as an example, the central server sends the global model w^t to the participants involved in this iteration, the i-th participant P_i trains locally based on its local private dataset D_i, and calculates the loss and obtains the gradient ∇G_i^t, and updates the local model w_i^t based on the gradient, and then submits the update of local model, i.e., the model difference before and after local training, denoted by w_i^t, to the central server.

$$w_i^t = w^t - \eta * \nabla G\left(w^t, D_i\right)$$

After collecting enough number of local model uploads from those selected participants, the central server performs an aggregation protocol to obtain the global update and further calculates the new global model accordingly. This global model is validated on the central test dataset to evaluate the performance of the model after updating.

$$w^{t+1} = \sum_{i=1}^{N} p_i w_i^t, \text{ s.t } \sum_{i=1}^{N} p_i = 1$$

After a number of iterations, the loss of the federated model gradually stabilizes, and eventually both the participants and the central server obtain a federated (global) model with good generalization performance, i.e., high model accuracy.

Definition 1 Strong Autocorrelation. *The difference between two consecutive local model updates $\|\Delta_i\theta - \Delta_{i-1}\theta\|$ is significantly smaller than the corresponding local model updates themselves $\Delta_i\theta$ and $\Delta_{i-1}\theta$. Generally, we can compare the L_2 norm of them:*

$$\|\Delta_i\theta - \Delta_{i-1}\theta\| < \|\Delta_i\theta\|, \|\Delta_{i-1}\theta\|$$

B. Local Differential Privacy

Local Differential Privacy. For LDP, each participant adds random noise needed during local execution. Each participant runs a randomized perturbation algorithm G on its local model update, and sends the results to the server. Based on the output of G, the private model update is protected from the perturbation result.

Definition 2 (ε, δ)-LDP. *Let X be a set of possible values and O the output of values. G is (ε, δ)-local differential private if for all $x, x' \in X$ and for all $o \in O$:*

$$\Pr[G(x) = o] \leq e^{\varepsilon} \Pr\left[G\left(x'\right) = o\right] + \delta,$$

where ε is the privacy budget, δ represents the probability of catastrophic failure, and $\Pr(\cdot)$ represents the probability of an event occurring. If $\delta > 0$, it is called relaxed differential privacy, and generally δ takes the value 10^{-5}.

Definition 3 Global Sensitivity. *For a function $G : D \to \mathbb{R}^d$, The dataset D is the input and R^d identifies the output of a d-dimensional vector of real numbers. Then for any two datasets D and D', the sensitivity of the function G is defined as follows:*

$$S(G) = \max_{D, D'} \left\|G(D) - G\left(D'\right)\right\|,$$

where $\|G(D) - G(D')\|$ denotes the paradigm distance between $G(D)$ and $G(D')$, typically the L_2 norm. The smaller *the distance, the smaller the sensitivity and the smaller the gap between the two.*

Definition 4 Gaussian Mechanism. *Given an algorithm G with global sensitivity $S(G)$, privacy budget ε, for $\delta \in (0,1)$, $\sigma > \frac{\sqrt{2\ln(1.25/\delta)}S(G)}{\varepsilon}$, and noise distribution $Y \sim N\left(0, \sigma^2\right)$, algorithm G' is said to be a randomized algorithm if it satisfies $G'(D) = G(D) + Y$, which satisfies $(\varepsilon, \delta) - DP$.*

Proposition 1 Post-processing. *For a randomized algorithm G_1, if it satisfies $(\varepsilon, \delta) - LDP$, then for any algorithm G_2 whose input is the output of G_1, $G_2\left(G_1\right)$ still satisfies $(\varepsilon, \delta) - LDP$.*

Definition 5 Rényi Divergence. *Given any two random distributions G_1 and G_2, the Rényi Divergence between G_1 and G_2 when order $\alpha > 1$ is defined as:*

$$D_\alpha\left(G_1 \| G_2\right) = \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim G_2}\left(\frac{G_1(x)}{G_2(x)}\right)^\alpha .$$

Definition 6 Rényi Differential Privacy. *For any randomization algorithm $G : D \rightarrow \mathbb{R}^d$, and any two neighboring datasets D and D', it satisfies $(\alpha, \varepsilon) - LDP$ if the following condition holds,*

$$D_\alpha\left(G(D) \| G\left(D'\right)\right) \leq \varepsilon .$$

C. Fed-LDP

In Fed-LDP, the participant iterates multiple rounds during the local training process, updates the local model using the noise-added model in each round of iteration, uploads the updated local model to the central server, and the central server directly weights and averages the local models of all participants to obtain a new round of the global model. Specifically, the training process of Fed-LDP is as follows:
(1) Initializing the global model: Before federated training starts, the server is responsible for initializing the global model and configuring the optimizer as well as the relevant hyperparameters, including the number of training rounds T, the number of local training rounds E, the clipping threshold $\mathcal{C}$, the noise scale σ, and the optimizer learning rate η.
(2) Participant selection and local model training: At the beginning of each training cycle, the server randomly selects K participants from the participant set to participate in the current training round. The selected local participants download the latest global model from the central server, perform random sampling (Poisson sampling) in the local dataset, and use these sample data for local training of the initial global model. The participants perform model updating with stochastic gradient descent to get the updated local model.
(3) Adding differential privacy perturbations: A participant computes the difference between its local models before and after local training in the i-th iteration, denoted by $\Delta_i\theta$, and clips it. This step aims to limit the sensitivity and ensure that the noise addition process satisfies the sensitivity of differential privacy. After clipping $\Delta_i\theta$ to a suitable size $\overline{\Delta_i\theta}$, an appropriate amount of perturbation is added based on the noise scale σ and the clipping threshold $\mathcal{C}$. The local model is then uploaded to the central server. At the same time, the privacy budget consumed by this training round is calculated using the Rényi Moment Accountant.
(4) Global model aggregation: The server collects the local model updates ($\overline{\Delta_i\theta}$ after perturbation) from all participants in the i-th iteration, and adjusts the contribution of each participant to the global model based on its weight to form an updated global model by means of Federal Average (Fed-Avg) [23] or other aggregation protocols.
(5) Iterative training: Iterative training: Repeat the above steps (1)-(4) until the privacy budget is consumed or the global model performance reaches a predetermined target.

Since the participant adds Gaussian noise with a mean of 0 sampled from the normal distribution $N(0, \sigma^2 \mathrm{I})$, then $\widetilde{\Delta_i\theta}$ is an unbiased estimator of $\overline{\Delta_i\theta}$. Therefore, by adding Gaussian noise to the local model update through the mechanism described above, the expected mean-square error for the first T rounds for the i-th user is the sum of the errors on all dimensions of the model, i.e., $\sigma^2\mathcal{C}^2 d$, which can be expressed as follows by Lemma III.1.

Lemma III.1. *Fed-LDP satisfies $(\varepsilon, \delta) - LDP$, which has an expected mean square error of $\sigma^2\mathcal{C}^2 d$:*

$$\frac{1}{t} \sum_{i=1}^{t} E\left[\left\|\widetilde{\Delta_i\theta} - \overline{\Delta_i\theta}\right\|^2\right] = \sigma^2\mathcal{C}^2 d,$$

where d denotes the dimension of the neural network, $\mathcal{C}$ denotes the sensitivity $S(G)$ of the model clipping, t is the total number of iterations, and σ denotes the noise scale.

D. Problem Definition

The goal of this paper is to implement a randomization mechanism A,which allows a federated learning participant to publish its training results to an untrustworthy central server while satisfying $(\varepsilon, \delta) - LDP$. Under the federated learning framework, the privacy publishing problem can be formalized as follows.

Problem Definition: For a participant who takes part in t successive rounds, and releases the corresponding local model updates, i.e., $\Delta_1\theta, \Delta_2\theta, \cdots, \Delta_t\theta$. Each update satisfies $\|\Delta_i\theta\| \leq \mathcal{C}(i = 1, \cdots, t)$. Design a randomized mechanism A, which takes the model update $\Delta_1\theta, \Delta_2\theta, \cdots, \Delta_t\theta$ as the input of G, and for the output noisy models, i.e., $\widetilde{\Delta_1\theta}, \widetilde{\Delta_2\theta}, \cdots, \widetilde{\Delta_t\theta}$, the utility can be guaranteed by minimizing the mean square error of the perturbed updates, i.e.,

$$\min \frac{1}{t} \sum_{i=1}^{t} E\left[\left\|\widetilde{\Delta_i\theta} - \Delta_i\theta\right\|^2\right]$$

Meanwhile, G should satisfy (ε, δ)-LDP, i.e.,

$$\begin{aligned} &\operatorname{Pr}\left[G\left(\Delta_1\theta, \Delta_2\theta, \cdots, \Delta_t\theta\right) \in O\right] \\ &\leq \exp(\varepsilon) \cdot \operatorname{Pr}\left[G\left(\Delta_1'\theta, \Delta_2'\theta, \cdots, \Delta_t'\theta\right) \in O\right] + \delta, \end{aligned}$$

where the set of outputs $O \subseteq Range(G)$.

By reducing the expected mean square error between the noised model and the original model during the training process, it aims to make the noised model update data of the local model reflect the changes of the model more realistically under the premise of satisfying the privacy protection, provide more accurate model training information, and thus accelerate the convergence of the global model.

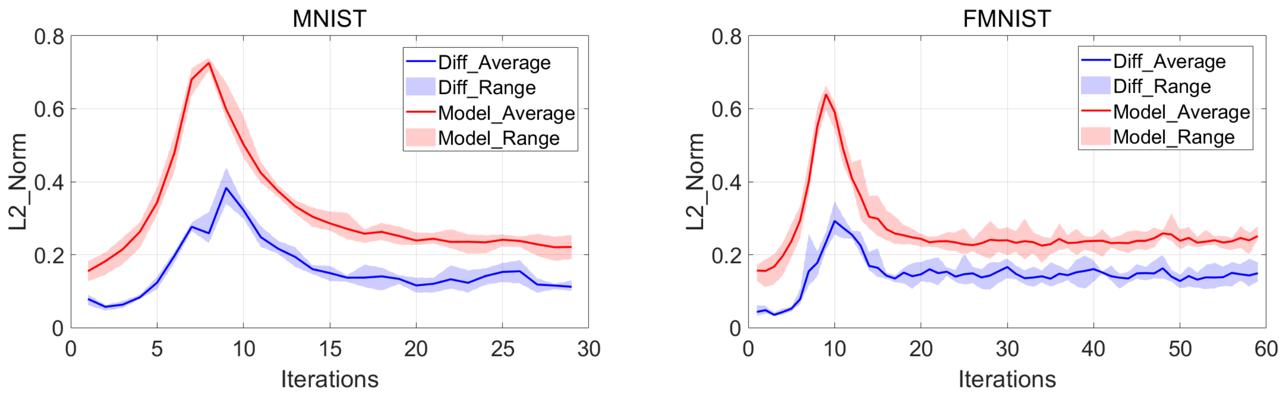

Fig. 1: The average and range of L_2 norm vs. iteration

E. Attack Model

We assume that the server is honest-but-curious, i.e. , the central server will collaborate with all participants to train the model well while being curious about each participant's local data. In addition, the participants involved in training can also be considered as honest-but-curious, i.e., while actively participating in the local training and uploading the local model updates, they also use the downloaded model to infer the local data of a particular participant.

IV. Observation on Temporal Correlation between Local Model Updates

A. Datasets and FL Tasks

We implement an FL system, and perform two image classification FL tasks on MNIST, FMNIST datasets, which are standard datasets for handwritten digit recognition and clothing classification. Each contains 60,000 training examples and 10,000 testing examples, respectively. The neural network used for training consists of two convolution layers and two fully connected layers, with ReLU activation between each layer. We set 100 participants for training, each participant possesses 600 training samples, and the samples belongs to different participants are of IID.

B. Analysis on the Autocorrelation between Local Model Updates

We empirically examine the autocorrelation between local model updates belonging to different iterations of the same participant. In this experiment, 100 participants are involved in each iteration. We calculate the average range of L_2 norm of the differences between two consecutive local model updates of the same participant in the both FL tasks. Figure 1 demonstrates the experimental results. We have two observations on the figure. First, it can be seen that, in all iterations, the L_2 norm is smaller than **1**, which implies that it is possible to find a small $Diff$ to bound the L_2 Norm with small value. Second, as the number of iterations increases, the value of the L_2 Norm first rapidly increases, then slowly decreases and tends to stabilize at a stage. It is because in the early stage of training, the accuracy of the model improves rapidly, and thus the local models undergo significant changes across iterations. In the later stage of training, as the federated model gradually converges, the changes of the local model are also become smaller. Thus, we can make use of the variation trend of the the L_2 norm of model difference to determine $Diff$ dynamically.

V. Design of Fed-CAD

Fed-CAD is based on the procedure of Fed-LDP. According to Lemma III.1, its trivial version (see in III-C) satisfies (ε, δ)-LDP. However, the noise added to each local model update depends on the scale of difference between updates and the noise scale σ. Large difference and σ imply significant noise. It is desired to reduce the mean squared error of expected value σ, so as to increase utility, while maintaining the privacy protection strength.

A. Perturbing Local Model Updates with Correlated Gaussian Mechanism

In our Fed-CAD, the server first initiates the total privacy budget, and start the iteration. In each iteration, given a succession of local model updates from the same participant, indicated by $\Delta\theta(d) = (\Delta_1\theta(d), \Delta_2\theta(d), \cdots, \Delta_t\theta(d))$. Norm clipping is conducted on each local model update to satisfy $\|\Delta_i\theta(d)\| \leq \mathcal{C}$. Suppose that the difference between adjacent model updates satisfies $\|\Delta_{i+1}\theta(d) - \Delta_i\theta(d)\| \leq Diff < \mathcal{C}$, where $Diff$ is a constant pre-determined by the server. We add Gaussian noise to $\Delta_i\theta(d)$ to satisfy (ε, δ)-LDP as follows:

First, Sample noise μ_1, which will be applied to $\Delta_1\theta(d)$, from a Gaussian distribution $N(0, \sigma^2 \mathrm{I})$ using Gaussian mechanism. Calculate random perturbed version of $\Delta_1\theta(d)$, i.e., $\widetilde{\Delta_1\theta(d)} = \Delta_1\theta(d) + \mu_1$. Notice that directly adding $\Delta_2\theta(d)$ with noise μ_2 sampled in the same way of generating μ_1 refers to the trivial version of Fed-LDP. Instead, Fed-CAD utilizes the Correlated Gaussian Mechanism (CGM) [15], which is able to exploit the correlation between consecutive values to reduce noise. To this end, we introduce an auxiliary function φ_2, i.e.,

$$\varphi_2 = \alpha * \Delta_2\theta(d) - \beta * \Delta_1\theta(d),$$

where $\alpha = 1 + (1 - Diff)^2$ and $\beta = (1 - Diff)$ are coefficients determined by $Diff$, forming the linear combination between $\Delta_2\theta(d)$ and $\Delta_1\theta(d)$. Since $Diff$ is publicly known, no additional privacy budget should be consumed on it.

Second, Sample Gaussian noise μ_2 from $N(0,\sigma^2\mathrm{I})$, and calculate $\widetilde{\varphi_2} = \varphi_2 + \mu_2$. Then, derive $\widetilde{\Delta_2\theta(d)}$ based on the following equation, i.e.,

$$\widetilde{\Delta_2\theta(d)} = \frac{1}{\alpha}\widetilde{\varphi_2} + \frac{\beta}{\alpha}\widetilde{\Delta_1\theta(d)}.$$

Then, we can obtain that,

$$\begin{aligned}\widetilde{\Delta_2\theta(d)} &= \frac{1}{\alpha}\varphi_2 + \frac{1}{\alpha}\mu_2 + \frac{\beta}{\alpha}\widetilde{\Delta_1\theta(d)} + \frac{\beta}{\alpha}\mu_1\\ &= \Delta_2\theta(d) - \frac{\beta}{\alpha}\Delta_1\theta(d) + \frac{1}{\alpha}\mu_2 + \frac{\beta}{\alpha}\Delta_1\theta(d) + \frac{\beta}{\alpha}\mu_1\\ &= \Delta_2\theta(d) + \frac{1}{\alpha}\mu_2 + \frac{\beta}{\alpha}\mu_1.\end{aligned}$$

Accordingly, it can be seen that the noise added in $\widetilde{\Delta_2\theta(d)}$ is a linear combination of μ_1 and μ_2 added in $\widetilde{\varphi_2}$ and $\widetilde{\varphi_1}$, respectively.

Adding noise through the auxiliary function φ is able to effectively reduce the amount of noise required to be injected into $\widetilde{\Delta_2\theta(d)}$. Since $\widetilde{\Delta_2\theta(d)}$ is the linear combination of noisy $\widetilde{\Delta_1\theta(d)}$ and $\widetilde{\varphi_2}$, which are unbiased noises that follow $N(0,\sigma^2\mathrm{I})$, the average value of noises in $\widetilde{\Delta_2\theta(d)}$ is $\mathbf{0}$. Consequently, $\widetilde{\Delta_2\theta(d)}$ is the unbiased estimation of $\Delta_2\theta(d)$. The variance of $\widetilde{\Delta_2\theta(d)}$ can be computed by,

$$Var(\widetilde{\Delta_2\theta(d)}) = \frac{1}{\alpha^2}\sigma^2 + \frac{\beta^2}{\alpha^2}\sigma^2 = \frac{\sigma^2}{1+(1-Diff)^2}.$$

It should be noticed that the trivial version directly injects noise into $\widetilde{\Delta_2\theta(d)}$, which is the same as the way of obtaining noisy $\widetilde{\Delta_1\theta(d)}$. Thus, the variance of noise in $\widetilde{\Delta_2\theta(d)}$ is σ^2, i.e.,

$$Var(\widetilde{\Delta_2\theta(d)}) = \sigma^2.$$

It is obvious that σ^2 is larger than $\frac{\sigma^2}{1+(1-Diff)^2}$.

We prove that such correlated noise satisfies (ε,δ)-LDP as follows: Given a pair of adjacent inputs, i.e., local model updates, d and d', the following inequality holds:

$$\begin{aligned}&\|\varphi_2(d) - \varphi_2(d')\|\\ &= \|\alpha\cdot(\Delta_2\theta(d) - \Delta_2\theta(d')) - \beta\cdot(\Delta_1\theta(d) - \Delta_1\theta(d'))\|\\ &\le (\alpha-\beta)\cdot\|\Delta_2\theta(d) - \Delta_2\theta(d')\| + \beta\cdot\|\Delta_2\theta(d) - \Delta_1\theta(d))\|\\ &+ \beta\cdot\|\Delta_1\theta(d') - \Delta_2\theta(d')\|\\ &\le 2\cdot(\alpha-\beta) + \beta\cdot Diff + \beta\cdot Diff\\ &\le 2 = \|\Delta_2\theta(d) - \Delta_2\theta(d')\|\end{aligned}$$

From the above we can conclude that the sensitivity of the auxiliary function φ is the same as that of the original function $\Delta\theta(d)$, and thus φ also satisfies (ε,δ)-LDP. According to the Post-processing property of LDP, reusing noise in previous iterations (updates) will not induce extra privacy risks. Then, injecting Gaussian noise μ_1 and μ_2 into $\Delta_1\theta(d)$ and φ_2, respectively, the privacy risk quantified by Rényi-DP, still satisfies (ε,δ)-LDP.

As a result, if the local model updates $\Delta\theta(d)$ belonging to a participant satisfies that the difference between two adjacent updates is smaller than $Diff$, i.e., $\|\Delta_{i+1}\theta(d) - \Delta_i\theta(d)\| \le Diff < \mathcal{C}$, it is feasible to make use of a linear combination of noises in the two updates. In other words, reusing μ_i in $\widetilde{\Delta_i\theta(d)}$ can reduce the amount of noise μ_{i+1} actually injected into $\widetilde{\Delta_{i+1}\theta(d)}$, and thus increase model update utility.

B. Adjusting $Diff$ Adaptively

One essential parameter in Fed-CAD is $Diff$. At the beginning of an FL training, the server decides and publishes it. $Diff$ is relevant to the strength of autocorrelation among local updates. According to the CGM, introducing $Diff$ helps to reduce the amount of noise, so as to enhance the contribution of participants, and weaken the impact of LDP mechanism to the performance of federated training.

It can be found that the variance of noise relates to not only the factor σ but also $Diff$. The smaller the value of $Diff$, the smaller the variance of noise. During FL, the L_2 of model updates changes with iterations. More specifically, $Diff$ will first increase and then decrease until it converges. It implies that it is reasonable to change $Diff$ dynamically to adapt to the variation of model update differences. To this end, we propose a strategy to adjust $Diff$, which draws inspiration from the concept of momentum in optimizer. In each iteration, when calculating the current $Diff$, we consider both the average historical values of $Diff$ and the difference between updates on the current and last iterations. Formally,

$$\mathbb{E}[Diff]_i = (1-\gamma)\cdot\mathbb{E}[Diff]_{i-1} + \gamma\cdot\|\Delta_i\theta(d) - \Delta_{i-1}\theta(d)\|,$$

where $\mathbb{E}[Diff]_{i-1}$ indicates the historical expectation of $Diff$ from the first to $(i-1)th$ iterations, γ is a hyper-parameter, named as *obsolete factor*, the value of which is between $\mathbf{0}$ and $\mathbf{1}$. The value of γ determines the importance of historical $Diff$ to the current one. A large value of γ means more consideration of past model updates, making it more stable during its updates.

Meanwhile, to avoid introducing new privacy risks caused by using $\mathbb{E}[Diff]_{i-1}$, it is necessary to perturb it with Gaussian noise. Likewise, since $Diff_i$ is composed of $\mathbb{E}[Diff]_{i-1}$ and $\|\Delta_i\theta(d) - \Delta_{i-1}\theta(d)\|$, it implies that we only need to inject noise into $\|\Delta_i\theta(d) - \Delta_{i-1}\theta(d)\|$ as $\mathbb{E}[Diff]_{i-1}$ is already perturbed in the previous iterations, i.e.,

$$\mathbb{E}[\widetilde{Diff}]_i = \mathbb{E}[Diff]_{i-1} + \gamma\cdot N(0,\sigma^2_{Diff}),$$

where σ_{Diff} is small constant and relevant to the variation range of $Diff$.

The procedures of Fed-CAD on the server and participant side are described in the Algorithm 1 and 2 with pseudo-code, respectively.

VI. Theoretical Analysis

A. Privacy Guarantee

Theorem VI.1. *The Fed-CAD algorithm satisfies* $(\varepsilon,\delta)-LDP$

Proof. 1) If $i=1$, i.e., for the first iteration, $\overline{\Delta_i\theta(d)} = \Delta_i\theta(d)/\max\left(1, \frac{\|\Delta_i\theta(d)\|}{\mathcal{C}}\right)$, and the added noise is normally

Algorithm 1 Fed-CAD: on the Participant Side

```
Input: Local training rounds $E$, sampling rate $q$, clipping threshold $\mathcal{C}$, noise scale $\sigma$, learning rate $\eta$
Output: Model update $\widetilde{\Delta\theta_i^t}$
1: $\theta \leftarrow \theta^{t-1}$ Initializing the Local Model
2: for $i = 1, 2, \ldots, E$ do
3:     $(x_i, y_i) \leftarrow batch\_size$ samples are randomly sampled by Poisson at sampling rate $q$ in the local dataset $D_i$
4:     $g_i = \nabla L(\theta, (x_i, y_i))$
5:     $\theta_i = \theta - \eta \cdot g_i$
6: end for
7: if $i = 0$ then
8:     $\Delta\theta_i^t = \theta_i - \theta$
9:     $\overline{\Delta\theta_i^t} = \Delta\theta_i^t / \max\left(1, \left\|\Delta\theta_i^t\right\| / \mathcal{C}\right)$
10:    $\widetilde{\Delta\theta_i^t} = \overline{\Delta\theta_i^t} + N\left(0, \sigma^2\mathcal{C}^2\mathrm{I}\right)$
11:    $v_i = 1$
12: else
13:    if $\left\|\overline{\Delta\theta_i^t} - \overline{\Delta\theta_{i-1}^t}\right\| \leq \mathbb{E}[Diff]$ then
14:        $r_i = \frac{1-\mathbb{E}[Diff]}{(1-\mathbb{E}[Diff])^2 + v_{i-1}}$
15:        $\sigma_i = ((1-r_i) + r_i \cdot \mathbb{E}[Diff]) \cdot \sigma$
16:        $\mu_i = N\left(0, \sigma_i^2\mathcal{C}^2\mathrm{I}\right) + r_i \cdot \mu_{i-1}$
17:        $\widetilde{\Delta\theta_i^t} = \overline{\Delta\theta_i^t} + \mu_i$
18:        $v_i = \frac{v_{i-1}}{(1-\mathbb{E}[Diff])^2 + v_{i-1}}$
19:    else
20:        $S = \max\left(1, \frac{\left\|\overline{\Delta\theta_i^t} - \overline{\Delta\theta_{i-1}^t}\right\|}{\mathbb{E}[Diff]}\right)$
21:        $\overline{\Delta\theta_i^t} = \overline{\Delta\theta_{i-1}^t} + \left(\overline{\Delta\theta_i^t} - \overline{\Delta\theta_{i-1}^t}\right) / S$
22:        goto 14
23:    end if
24: end if
25: $\mathbb{E}[Diff]_i = (1-\gamma) \cdot \mathbb{E}[Diff]_{i-1} + \gamma \cdot \left\|\overline{\Delta\theta_i^t} - \overline{\Delta\theta_{i-1}^t}\right\|$
26: return $\widetilde{\Delta\theta_i^t}$
```

Algorithm 2 Fed-CAD: on the Server Side

```
Input: training round $T$, local training round $E$, clipping threshold $\mathcal{C}$, model update difference threshold $Diff$, noise scale $\sigma$, learning rate $\eta$
Output: global model $\theta^T$, privacy budget $\{\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n\}$
1: $\theta_0 \leftarrow$ Initializing the Local Model
2: for iteration $t$ in range $T$ do
3:     $K \leftarrow$ Randomly select $K$ participants
4:     for each participant $P_i \in K$ do
5:         $\Delta\theta_i^t \leftarrow ParticipantUpdate(\theta^{t-1}, D_i, \sigma, \cdots)$
6:         $\varepsilon_i^t = Rényi\_Moment\_Accountant(\sigma)$
7:     end for
8:     $\theta_t = \theta_{t-1} + \sum_{i=1}^{K} \Delta\theta_i^t / K$
9: end for
10: return $\theta^T$, $\{\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n\}$
```

distributed Gaussian noise that satisfies the differential privacy sensitivity $\mathcal{C}$ (which generally takes the value of 1.0) , and the addition process satisfies $(\varepsilon, \delta) - LDP$.

2) If $i > 1$ and $\left\|\overline{\Delta_i\theta(d)} - \overline{\Delta_{i-1}\theta(d)}\right\| < \mathbb{E}[Diff] < \mathcal{C} = 1.0$

$$\begin{aligned}\overline{\Delta_i\theta(d)} &= \overline{\Delta_i\theta(d)} - r_i \cdot \overline{\Delta_{i-1}\theta(d)} + r_i \cdot \overline{\Delta_{i-1}\theta(d)} \\ &= (1-r_i) \cdot \overline{\Delta_i\theta(d)} + r_i \cdot \overline{\Delta_i\theta(d)} - r_i \cdot \overline{\Delta_{i-1}\theta(d)} \\ &+ r_i \cdot \overline{\Delta_{i-1}\theta(d)} \\ &= (1-r_i) \cdot \overline{\Delta_i\theta(d)} + r_i \cdot \left(\overline{\Delta_i\theta(d)} - \overline{\Delta_{i-1}\theta(d)}\right) \\ &+ r_i \cdot \overline{\Delta_{i-1}\theta(d)}\end{aligned}$$

Then the noisy version $\widetilde{\Delta_i\theta(d)}$ of $\overline{\Delta_i\theta(d)}$ can be represented as follows,

$$\begin{aligned}\widetilde{\Delta_i\theta(d)} &= (1-r_i) \cdot \widetilde{\Delta_i\theta(d)} + r_i \cdot \left(\widetilde{\Delta_i\theta(d)} - \widetilde{\Delta_{i-1}\theta(d)}\right) \\ &+ r_i \cdot \widetilde{\Delta_{i-1}\theta(d)}\end{aligned}$$

Since $\widetilde{\Delta_{i-1}\theta(d)}$ is already processed by the noise addition in the previous round of model training, according to the post-processing principle of differential privacy, the reuse for the $\Delta_{i-1}\theta(d)$ part does not cause additional privacy overhead. Therefore, we only need to focus on the noise added to the $(1-r_i) \cdot \widetilde{\Delta_i\theta(d)} + r_i \cdot \left(\widetilde{\Delta_i\theta(d)} - \widetilde{\Delta_{i-1}\theta(d)}\right)$. And the sensitivity $S(G)$ of this part can be expressed as

$$S(G) = (1-r_i) + r_i \cdot \mathbb{E}[Diff] < (1-r_i) + r_i = 1 = \mathcal{C}$$

So, adding a normally distributed Gaussian noise that satisfies the differential privacy sensitivity of $(1-r_i) + r_i \cdot \mathbb{E}[Diff]$ to this section also satisfies $(\varepsilon, \delta) - LDP$.

3) If $\left\|\overline{\Delta_i\theta(d)} - \overline{\Delta_{i-1}\theta(d)}\right\| > \mathbb{E}[Diff]$, we clip $\overline{\Delta_i\theta(d)}$ to satisfy that the L_2 norm difference is less than $Diff$, i.e.,

$$\overline{\Delta_i\theta(d)} = \overline{\Delta_{i-1}\theta(d)} + \frac{\overline{\Delta_i\theta(d)} - \overline{\Delta_{i-1}\theta(d)}}{\max\left(1, \frac{\left\|\overline{\Delta_i\theta(d)} - \overline{\Delta_{i-1}\theta(d)}\right\|}{\mathbb{E}[Diff]}\right)}$$

The processed update satisfies case 2), and adding noise in the way 2) satisfies $(\varepsilon, \delta) - LDP$. In summary, Fed-CAD satisfies $(\varepsilon, \delta) - LDP$, and the proof is complete. □

B. Variance Analysis

Theorem VI.2. *The noise variance added by Fed-CAD is* $\mathrm{Var}\left(\widetilde{\Delta_i\theta(d)}\right) = \frac{2\mathbb{E}[Diff] - \mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^{2i}} \cdot \sigma^2$.

Proof. According to Fed-CAD, noise μ_i added by $\widetilde{\Delta_i\theta(d)}$ is represented as

$$\mu_i = N(0, \sigma_i^2\mathrm{I}) + r_i \cdot \mu_{i-1},$$

where $\sigma_i = ((1-r_i) + r_i \cdot \mathbb{E}[Diff]) \cdot \sigma$.

Then there is

$$\begin{aligned}&Var(\widetilde{\Delta_i\theta(d)}) = \sigma_i^2 + r_i^2 \cdot Var(\widetilde{\Delta_{i-1}\theta(d)}) \\ &= ((1-r_i) + r_i \cdot \mathbb{E}[Diff]) \cdot \sigma^2 + r_i^2 \cdot Var(\widetilde{\Delta_{i-1}\theta(d)})\end{aligned}$$

When the current round is the first training round $(i = 1)$,

$$Var(\widetilde{\Delta_1\theta(d)}) = \sigma^2, v_1 = 1.$$

When $(i = 2)$,

$$\begin{cases} r_2 = \dfrac{1-\mathbb{E}[Diff]}{(1-\mathbb{E}[Diff])^2+1} \\ v_2 = \dfrac{1}{(1-\mathbb{E}[Diff])^2+1} \\ \mathrm{Var}\left(\widetilde{\Delta_2\theta(d)}\right) = \dfrac{2\mathbb{E}[Diff]-\mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^4}\cdot\sigma^2 \end{cases}$$

When$(i = 3)$,

$$\begin{cases} r_3 = \dfrac{1-\mathbb{E}[Diff]}{(1-\mathbb{E}[Diff])^2+\frac{2\mathbb{E}[Diff]-\mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^4}} \\ v_3 = \dfrac{v_2}{(1-\mathbb{E}[Diff])^2+v_2} \\ \mathrm{Var}\left(\widetilde{\Delta_3\theta(d)}\right) = \dfrac{2\mathbb{E}[Diff]-\mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^6}\cdot\sigma^2 \end{cases}$$

By mathematical induction, it can be inferred that, when the round is the i-th round,

$$\begin{cases} r_i = \dfrac{1-\mathbb{E}[Diff]}{(1-\mathbb{E}[Diff])^2+\frac{2\mathbb{E}[Diff]-\mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^{2i-2}}} \\ v_i = \dfrac{v_{i-1}}{(1-\mathbb{E}[Diff])^2+v_{i-1}} \\ \mathrm{Var}\left(\widetilde{\Delta_i\theta(d)}\right) = \dfrac{2\mathbb{E}[Diff]-\mathbb{E}[Diff]^2}{1-(1-\mathbb{E}[Diff])^{2i}}\cdot\sigma^2 \end{cases}$$

□

VII. Experiments

A. Methodology

We conduct a series of experiments on the FL tasks mentioned before. We use Fed-LDP as our comparison method, which adopts fixed-norm clipping and applies one-shot Gaussian noise-based LDP on local model updates. The experimental results show that our Fed-CAD (both constant and adaptive $Diff$) outperforms Fed-LDP. We implement all experiments using PyTorch with RTX 4080Ti.

We set 100 participants for training, each participant possesses 600 training samples, including IID and non-IID settings with different partition of data. In each iteration, a portion of participants will be selected to join the federated training, e.g., 10%. We use the optimizer of SGD with a learning rate η of 0.01. Each participant conducts one epoch of local training for updating local model, and the batch size is 60 (i.e., the sampling rates q is 0.1.).

B. Impact of Obsolete Factor γ

Figure 2 gives the impact of different γ on the global model accuracy on the MNIST dataset. A larger γ indicates that more attention is given to change of L_2 Norm difference between the past and current model updates. When γ = 0, $Diff$ is a constant value. We set the clipping threshold $\mathcal{C}$ = 1.0, the initial $Diff$ = 0.5 and the noise scale σ = 0.3. It can be seen that in the early stage of model training, especially in the rapid convergence stage, the model with γ = 0 performs better and converges faster, simultaneously L_2 Norm difference between the past and current model updates rises rapidly. That's because the $Diff$ initially published by

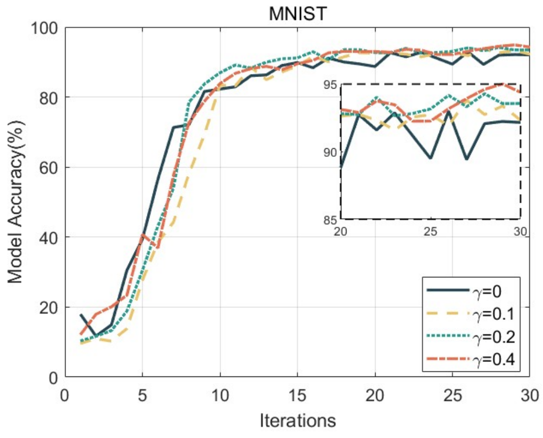

Fig. 2: The impact of different γ

the server is relatively large in order to meet the threshold requirement of the participants most of the time, and thus fewer extra operations are needed to bound the correlation difference between the past and current model updates under the fixed $Diff$. The model with $\gamma > 0$ triggers the model correlation operation repeatedly which minimizes the $Diff$ value. It makes the current updating of the models not truly reflecting the changing trend of them, which slows down then convergence. When the model is close to convergence, the range of threshold variance is smaller at this time, and a larger γ implies the participants could adjust their difference threshold $Diff$ rapidly and supply suitable scale of noise to the local model updates in each iteration, so the model eventually converges better. In order to improve the final model accuracy, the factor γ is set to 0.4 in the following experiments.

C. Impact of Noise Scale on Model Accuracy

Figures 3 and 4 demonstrate the performance of the global models in Fed-LDP and Fed-CAD under different σ. It can be seen, for the same σ. comparing with Fed-LDP, both constant and adaptive Fed-CAD achieve better accuracy with the same privacy, especially when σ increases. On MNIST, when the number of global iterations is 30 and the noise scale σ is set as 0.3 (ε = 2.72) and 0.5 (ε = 1.5), Fed-CAD achieves (2%, 4%) and (10%, 16%) accuracy improvement compared to Fed-LDP, respectively. On FMNIST, when the number of global iterations is 60 and the noise scale σ is set as 0.3 (ε = 3.97), Fed-CAD achieves (4%, 8%) accuracy improvement compared to Fed-LDP, respectively. While the noise scale σ is as small as 0.1, the noise variance is small, Fed-CAD needs to perform additional model correlation operations, resulting in a slower convergence than Fed-LDP, while the final accuracy is nearly the same.

D. Impact of the Number of Participants

In Figure 5, we analyze the impact of the numbers of participants on MNIST and FMNIST. For MNIST, we set the noise scale σ = 0.3 and $Diff$ = 0.5. It can be seen that the Fed-CAD outperforms Fed-LDP under all settings.

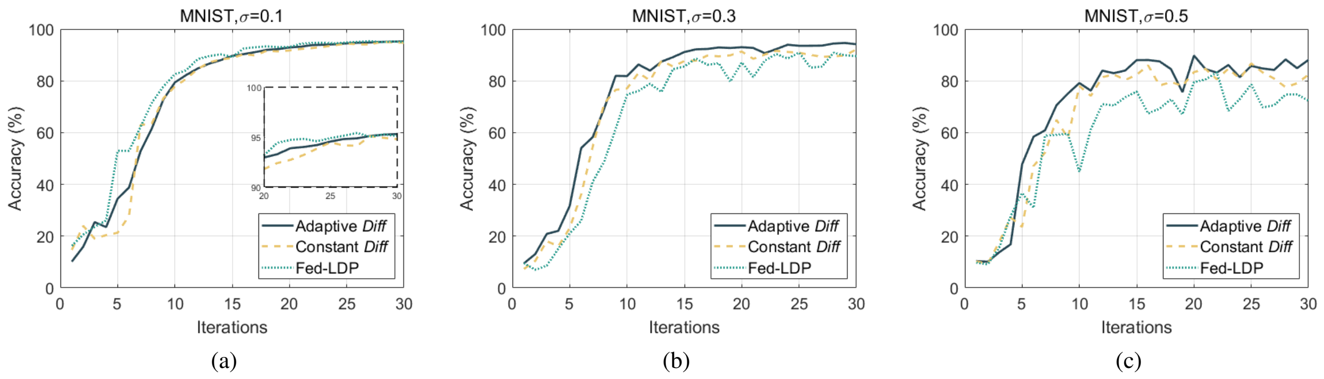

Fig. 3: The impact of different σ on the MNIST Dataset

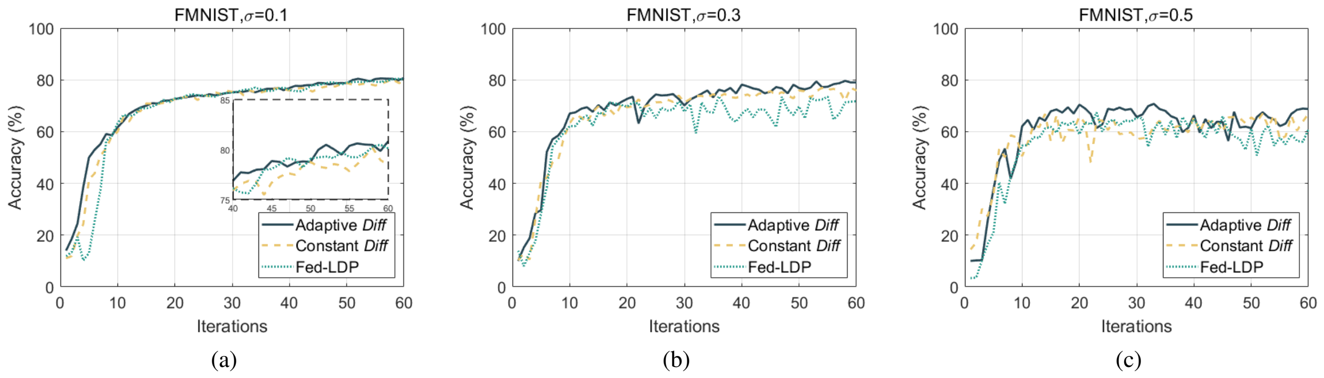

Fig. 4: The impact of different σ on the FMNIST Dataset

On FMNIST, we set the noise scale the same as before and $Diff = 0.7$. We can see that two types of Fed-CAD always outperform Fed-LDP. Furthermore, when the number of participants is small, the constant $Diff$ performs better than adaptive $Diff$. When more participants are involved in the global training, Fed-CAD with adaptive $Diff$ achieves better accuracy than that of constant $Diff$.

E. Impact of Data Heterogeneity

In Figure 6, we analyze the impact of data heterogeneity on Fed-CAD and Fed-LDP. We utilize the setting used by McMahan et al., [23] to achieve the pathological non-IID partition of data. We can observe that the Fed-CAD exhibits high robustness on data heterogeneity. We set the noise scale $\sigma = 0.3$ and $Diff = 0.5$ on MNIST and $Diff = 0.7$ on FMNIST. Under different data distributions, Fed-CAD always outperforms Fed-LDP. Particularly when the number of classes held by each participant is 2, Fed-LDP achieves 84.7% accuracy on MNIST. This is mainly because that the data similarity provides more clear direction for the local model to move during federated updating, therefore the noise variance could be smaller. As the number of classes held by each participant increasing, Fed-CAD achieves (1%, 2%) and (1%, 1%) accuracy improvements on MNIST, (3%, 5%) and (3%, 1%) on FMNIST resepctively, compared with Fed-LDP.

VIII. Conclusion

In this paper, we take advantage of the strong auto-correlation between local model updates so as to alleviate the paradox between privacy risk and data utility, in LDP-based FL systems, with negligible computational overhead. We introduce a Correlation-aware Adaptive Differential Privacy mechanism, named Fed-CAD. In our Fed-CAD, a clipping bound is adaptively selected and applied to guarantee the maximum difference between local model updates. The temporally correlated Gaussian noise, i.e., a combination of the fresh Gaussian noise and a portion of noise contained in the previous noisy updates is injected to model updates, so as to reduce the noise scale while maintaining the privacy protection strength. compared to the one-shot Gaussian noise. We demonstrate the correctness and efficacy of Fed-CAD with both formal proof and extensive experiments.

Acknowledgment

This work was supported in part by Natural Science Foundation of Shanghai (Grant No.22ZR1400200), Fundamental Research Funds for the Central Universities (No. 2232023Y-01), HK RGC under Grants R6021-20F, R1012-21, RFS2122-1S04, C2004-21G, C1029-22G, and N CityU139/21.

References

[1] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, “Protection against reconstruction and its applications in private federated learning,” *arXiv preprint arXiv:1812.00984*, 2018.

[2] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, “Exploiting unintended feature leakage in collaborative learning,” in *2019 IEEE symposium on security and privacy (SP)*, pp. 691–706, 2019.

[3] K. Du, S. Chang, H. Wen, and H. Zhang, “Fighting adversarial images with interpretable gradients,” in *Proceedings of the ACM Turing Award Celebration Conference-China*, pp. 44–48, 2021.

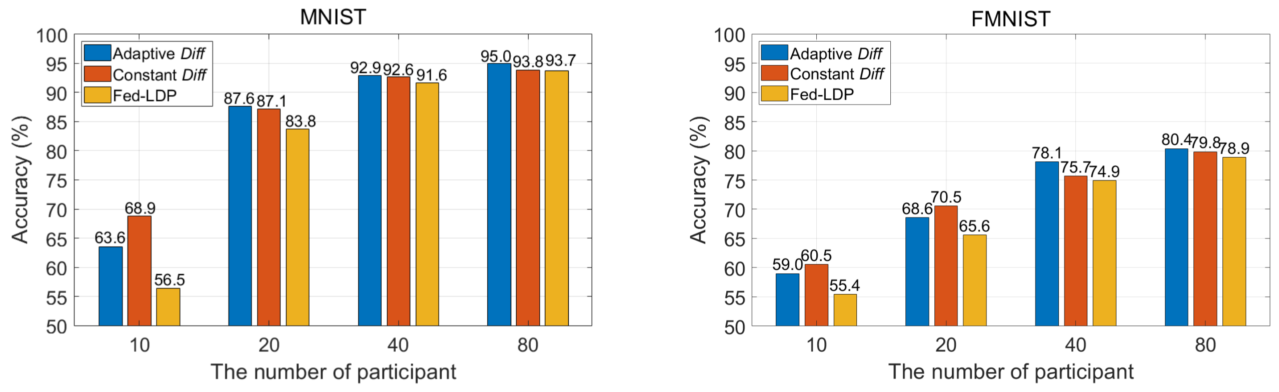

Fig. 5: The impact of the number of participant

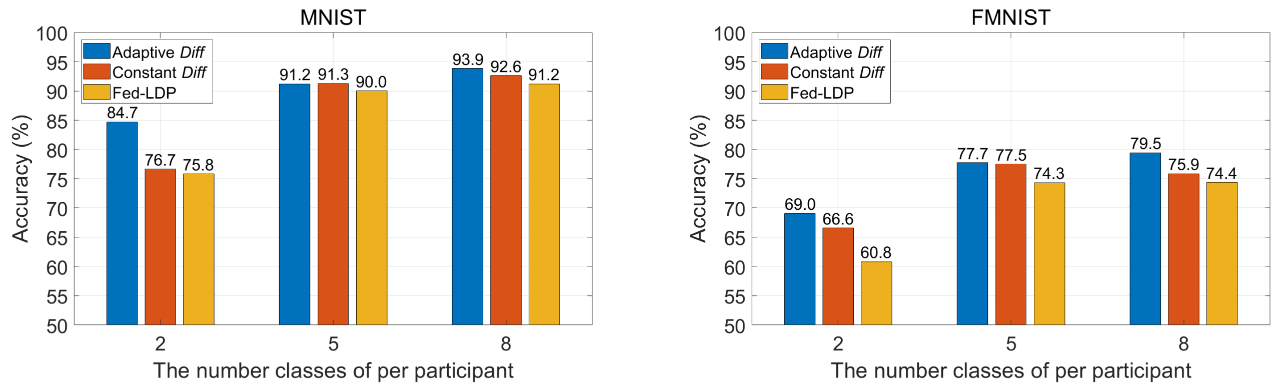

Fig. 6: The impact of Data Heterogeneity

[4] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," *arXiv preprint arXiv:1710.06963*, 2017.

[5] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpsgd: Communication-efficient and differentially-private distributed sgd," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[6] S. Chang, Y. Tao, H. Zhu, and B. Li, "Friendseeker: Inferring hidden friendship in mobile social networks with sparse check-in data," in *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, pp. 440–450, IEEE, 2023.

[7] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[8] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[9] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[10] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," *arXiv preprint arXiv:2009.03561*, 2020.

[11] R. Liu, Y. Cao, M. Yoshikawa, and H. Chen, "Fedsel: Federated sgd under local differential privacy with top-k dimension selection," in *Database Systems for Advanced Applications: 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24–27, 2020, Proceedings, Part I 25*, pp. 485–501, 2020.

[12] Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE transactions on information forensics and security*, vol. 13, no. 5, pp. 1333–1345, 2017.

[13] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1310–1321, 2015.

[14] J. Liu, J. Lou, L. Xiong, J. Liu, and X. Meng, "Projected federated averaging with heterogeneous differential privacy," *Proceedings of the VLDB Endowment*, vol. 15, no. 4, pp. 828–840, 2021.

[15] E. Bao, Y. Yang, X. Xiao, and B. Ding, "Cgm: an enhanced mechanism for streaming data collection with local differential privacy," *Proceedings of the VLDB Endowment*, vol. 14, no. 11, pp. 2258–2270, 2021.

[16] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," *arXiv preprint arXiv:1712.07557*, 2017.

[17] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pp. 61–66, 2020.

[18] B. Bebensee, "Local differential privacy: a tutorial," *arXiv preprint arXiv:1907.11908*, 2019.

[19] L. Sun, X. Ye, J. Zhao, C. Lu, and M. Yang, "Bisample: Bidirectional sampling for handling missing data with local differential privacy," in *Database Systems for Advanced Applications: 25th International Conference, DASFAA 2020, Jeju, South Korea, September 24–27, 2020, Proceedings, Part I 25*, pp. 88–104, Springer, 2020.

[20] P. C. Mahawaga Arachchige, D. Liu, S. Camtepe, S. Nepal, M. Grobler, P. Bertok, and I. Khalil, "Local differential privacy for federated learning," pp. 195–216, 2022.

[21] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pp. 1054–1067, 2014.

[22] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pp. 61–66, 2020.

[23] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, 2017.