Faster Convergence on Differential Privacy based Federated Learning

Shangyin Weng, *Student Member, IEEE,* Lei Zhang, Xiaoshuai Zhang, *Senior Member, IEEE* and Muhammad Ali Imran, *Fellow, IEEE*

***Abstract*—As a novel distributed machine learning approach, federated learning (FL) is proposed to train a global model while preserving data privacy. However, some studies manifest that adversaries can still recover private information from the gradients. Differential privacy (DP) is a rigorous mathematical tool to protect records in a database against leakage. It has been widely applied in FL by perturbing the gradients. Nevertheless, while using DP in FL, the convergence performance of the global model is inevitably degraded. In this paper, we implement a DP-based FL scheme, which achieves local DP (LDP) by adding well-designed Gaussian noise on the gradients before clients upload them to the server. After that, we propose two strategies to improve the convergence performance of the DP-based FL. Both methods are realized by modifying the local objective function to limit the effect of LDP noise on convergence without degrading the privacy protection level. We then provide the detailed framework which adopts the LDP scheme and two strategies. The framework on different machine learning models is tested by simulation results, which show that our framework can improve the convergence performance up to 40% faster under different noise compared with other DP-based FL. Finally, we show the theoretical convergence guarantee of our proposed framework by first presenting the expected decrease in the global loss function for one round of training and then providing the upper convergence bound after multiple communication rounds.**

***Index Terms*—Privacy-preserving federated learning, differential privacy, convergence performance.**

I. Introduction

With the enormous amount of data generated from the Internet of Things, artificial intelligence (AI) has been broadly developed and deployed in recent years in many sectors, including finance, industries, network service applications, etc. For such an unprecedented blooming with the great benefits brought by AI that relies on a large amount of data to achieve acceptable performance, there is a privacy problem causing great attention in public during data collection. In addition, with the new General Data Protection Regulation (GDPR) law [1], it becomes more difficult to collect raw data to train a good ML model. To solve this issue, Google first proposes federated learning (FL) in the smart keyboard application for typing recommendation [2]. The key idea of FL is to let local users train models with their data and upload the gradients instead of their raw data, which will then be aggregated to obtain a global model. Since the private data never leaves local users' devices, it is claimed that FL can preserve privacy during training models.

A baseline FL model, Fed-Avg, is proposed to use stochastic gradient descent (SGD) to train local models for multiple epochs to reduce the communication rounds between the server and clients [2]. However, when training with Non-independent and identically distributed (Non-IID) data, Fed-Avg has unsatisfactory convergence performance [3]. Several local optimizers for FL are proposed to improve the FL convergence performance with Non-IID data and proved to converge much faster than Fed-Avg [4], [5]. In addition, many studies have proposed different structures for FL to speed up convergence time. For example, authors in [6] propose a hierarchical clustered FL and authors in [7] use Blockchain as coordination to achieve decentralized FL.

Even though FL is proposed for privacy protection, several studies have shown that useful information can be recovered even from the gradients of the trained model to violate user privacy [8], [9]. In many FL settings, the servers are honest to finish the FL tasks, but sometimes they are curious about the users' private data and try to recover them from the gradients. Therefore, further studies are essential to protect user privacy when FL servers are not fully trusted. For example, homomorphic encryption (HE) can be used by each client to encrypt the gradients before they are uploaded to the server [10]. By using HE, the server can aggregate all the encrypted gradients without acknowledging the content shared by each client. Then, the clients decrypt the aggregated encrypted gradients to obtain the new global model for training. However, this requires that all the clients are trustworthy to protect the secret key. Furthermore, performing HE on the gradients needs strong computational capability, which is normally unavailable for resource-constrained smart devices. On the other hand, as a mathematical tool, differential privacy (DP) is widely adopted in FL by adding noise or perturbing original gradients before uploading to the server to provide a strong privacy guarantee [11], [12], also known as Local-DP (LDP). However, the work in [12] has a strict bound for privacy loss, which might be difficult for the real world. Besides, the work in [11] still requires the server to compute the privacy loss for users so that the server may fool the users for more training rounds, causing

All work in this paper are done in University of Glasgow.

Shangyin Weng, Lei Zhang and Muhammad Ali Imran are with the James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ, UK (e-mail: s.weng.2@research.gla.ac.uk; lei.zhang@glasgow.ac.uk; muhammad.imran@glasgow.ac.uk). Corresponding authors: Xiaoshuai Zhang and Lei Zhang.

Xiaoshuai Zhang is with the Ocean University of China (e-mail: x.zhang@ouc.edu.cn).

privacy leakage. Therefore, a new LDP-based scheme for FL is considered in this paper. Although DP is a strong privacy protection tool, perturbing gradients inevitably decrease the convergence performance to incur longer training time and lower accuracy. Thus, how to improve the convergence performance under DP noise still needs to be addressed.

In this paper, we propose an LDP-based framework and two novel strategies for modifying the local objective function to reduce the convergence time cost and improve the accuracy performance while maintaining the same privacy protection level, namely Federated Noise Reduction 1&2 (Fed-Nore-1&2). We implement an LDP-based FL framework by adding Gaussian noise before uploading their gradients and use RDP to keep track of the privacy loss. Our contributions are listed as follows.

- Our first proposed strategy is to compute the difference between the gradients with and without DP noise and add the difference values to the loss function to limit the effect of the noise. On the other hand, by adding noise to the gradients, it is considered that noisy gradients will generate an additional term to the final loss. Our second proposed strategy is to calculate the increment of the loss from the noise on the gradients and then subtract that increment from the local objective function. Detailed formulas on how to make those modifications to the local objective function for different learning models are illustrated. To the best of our knowledge, this is the first of its kind to try to improve the FL's convergence performance under the DP noise by limiting its possible effect on the loss without degrading the privacy protection level.
- A theoretical convergence bound on the first modified local objective function is developed, which presents the expected increment in the loss function of one round and then the upper convergence bound after multiple rounds.
- We perform simulations on our proposed framework, and the results present that our proposed framework can spare up to 40% training rounds to reach the same performance as normal DP-based FL under certain settings. Besides, our proposed work also achieves higher accuracy performance compared to other DP-FL.

The remainder of this paper is structured as follows. A summary of the notations is given in Table I. Section II introduces the related works, and Section III presents the background of the theories of DP and the threat model. In Section IV, the privacy-preserving FL framework based on DP and the models to improve the performance under DP noise are proposed. We give the theoretical convergence bound analysis in Section V. Then, the simulation and the numerical results are shown in Section VI. Finally, this article is concluded in Section VII.

II. Related works

With the emerging development of ML and the rising attention to privacy, FL (Fed-Avg) is proposed. However, it still suffers from privacy leakage.

Table I: Summary of Main notations

D, D'	Two adjacent datasets
R	A DP mechanism
ϵ, δ, α	The privacy budget for DP
M	The number of the clients in total
m	The number of the selected clients in each round
W^t	The global model of the tth communication round
t	The tth communication round
$F()$	The local loss function
W_i	The model of the ith client
∇W_j^t	The gradients of the jth client in tth round
C	The sensitivity in the DP mechanism
n	The number of data of each client
σ	The base noise variance of the DP mechanism
J	The distance between noisy gradients and original ones in Fed-nore1
$N(\mu, \sigma^2)$	A Gaussian distribution with a mean of μ and a variance σ^2
$\lambda_{nore1}, \lambda_{nore2}$	A factor used to control the Fed-nore
w_k, b_k	The parameters of the kth hidden and bias layer
z_k, a_k	The kth layer's intermediate optimization parameters
dz^{conv}, dz^{pool}	The dz of the convolution layer and pooling layer
K	The final layer of the model
$g(), g()'$	The activation function and its derivative
n'	The number of the input data of training
$d\overline{z}, d\overline{w}, d\overline{b}$	The expected change on the gradients in Fed-nore-2
l	The index of the true label in the one-hot output

Dwork et al. define pure ϵ-DP by considering the information loss between two datasets [13]. Then, (ϵ, δ)-DP is introduced to provide better flexibility. Besides, in applications requiring multiple uses of the DP mechanism, (ϵ, δ)-DP can adopt the advanced composition DP theorem to obtain a tighter bound of privacy loss. As regards privacy protection for learning algorithms, Martin et al. [14] first adopt DP into single-end ML, namely as DP-SGD. The algorithm achieves DP by adding Gaussian noise in every trained batch. To record the accumulative privacy loss during the multiple DP procedures, they come up with moments accountant (MA), which has a tighter bound of privacy loss for DP-based learning, compared to the original DP composition theorem [13]. Moreover, authors in [15] have proposed Rényi differential privacy (RDP) or (α, ϵ)-DP by using Rényi divergence to calculate the distance between datasets, which is a natural relaxation for (ϵ, δ)-DP and it is proved to be more flexible and effective than previous DP composition theories. Then, Geyer et al. [16] first propose a client-level DP-based FL framework, also known as Central-DP (CDP), for the purpose of hiding every client's track in training. It is achieved by adding Gaussian noise to the global model before the central server broadcasts it. However, malicious servers can still recover the original data. After that, LDP is proposed to randomize local gradients on the local side to protect data privacy [17], [18]. Authors in [17] implement LDP by perturbing the local gradients, while authors in [18] propose to add noise to the local gradients before uploading them. However, their frameworks satisfy the Laplace mechanism, which may have too tight bounds to be achieved in real-world algorithms [13].

Even though DP can protect private data from leaking to adversaries, the DP-based learning algorithms have a degraded convergence performance in terms of longer convergence time and lower accuracy performance due to the randomization of the gradients [19], [20] and the studies in [19] also present the convergence bound for different client select rates in each communication round. Furthermore, to enhance privacy protection, the authors in [21] have adopted secure multi-party computation combined with DP into FL, which can also improve convergence performance. Besides, the work in [22] proposes a contract-based incentive mechanism by modeling local users' contribution and computation, privacy and communication costs to improve FL's utility performance. The study in [23] utilizes HE along with DP to improve privacy protection. In addition, the authors in [24] have proved that applying CDP and LDP in FL can prevent the training from backdoor attacks. Nevertheless, the accuracy performance of DP-based FL is still greatly degraded due to the noise. However, only a few works are trying to improve the accuracy performance by using adaptive gradient clipping [25], [26], while these works affect the DP settings by using relatively smaller noise. This paper proposes to limit the effect of noise on the accuracy performance during local training by adding a noise-related term to the local objective function so that, during gradient descent, the noise effect on the accuracy performance can also be decreased while no changes are made to the DP mechanism.

III. Preliminaries

In this section, we introduce the definitions and principles of DP and the threat model in this paper.

A. Differential Privacy

The DP mechanism is utilized for privacy protection for sharing data. One relaxed definition of the DP mechanism, the (ϵ, δ)-DP, is that there are two neighboring databases D, D', and after adding the DP mechanisms, the probability distributions (Pr) of their outputs are bounded by e^{ϵ}, and δ is the probability of that the difference between the probabilities is not bounded for the given ϵ. This mechanism is known as the Gaussian mechanism, which is formalized as:

***Definition** 1:* A randomized mechanism R achieves (ϵ, δ)-DP if it satisfies the following, $\forall D, D'$, and $S \subset \mathbb{R}$ [13] :

$$Pr\left[R(D) \in S\right] \leq e^{\epsilon} Pr\left[R(D') \in S\right] + \delta, \tag{1}$$

where $\epsilon > 0$ and $\delta \geq 0$.
To obtain a relaxed privacy calculation, RDP is adopted in this paper, which is defined as follows [15]:

***Definition** 2:* A randomized mechanism M achieves (α, ϵ)-DP if it satisfies [15]:

$$Pr\left[M(D) \in S\right] \leq \left(e^{\epsilon} Pr\left[M(D') \in S\right]\right)^{(\alpha-1)/\alpha}, \tag{2}$$

where $\alpha \in (1, +\infty)$.

B. Threat model

Even though in FL, only gradients are uploaded, and the data is stored locally, useful information can still be recovered. In this paper, we consider a reconstruction attack as the threat model, where we assume that the server is honest but curious. To be specific, they execute the FL honestly, but they try to infer the private data from the transmitted data. Therefore, we aim to prevent the model from obtaining the real data.

IV. Proposed optimization of Federated Learning with noise

In this section, we propose an LDP-based FL scheme by adding Gaussian noise. Then, two modifications on the local cost function $F(W)$ are introduced to improve the convergence under noise. The first can work on universal models, while the second slightly varies for different ML models. In this paper, the CNN and the DNN are used as the training model along with the two modifications, and we provide the corresponding derivations of the second modification on the local objective function for them.

A. Differential privacy Federated Learning

We apply LDP to protect sensitive data from revealing. Before applying DP mechanisms, each layer of the gradients needs to be clipped element-wise as:

$$\nabla W_i^t = \nabla W_i^t / max(1, \frac{||\nabla W_i^t||_2}{nC}), \tag{3}$$

so that it can eliminate their effect on average value to make them close to the global gradients, where C is the sensitivity of LDP, n is the size of the involved data samples and nC is the clipping threshold. The DP noise is generated as $N(0, C^2\sigma^2)$. The σ^2 is a preset base noise variance, and C is used to control the noise scale. We adopt the calculation of C in [11]:

$$C = \frac{median||\nabla W_i^t||_2}{n} = \frac{median||W^{t-1} - W_i^t||_2}{n}, \tag{4}$$

where C is calculated for each layer separately and by taking the median value of all unclipped gradients in each client. Then, the noise is added to the local gradients, and the noisy gradients are sent to the server for aggregation. To track the privacy loss, we use RDP [15] to calculate the final privacy loss (ϵ, α) with a fixed δ and the total training rounds. In our LDP scheme, each client records their privacy loss locally and individually. Once the client has reached the preset privacy budget (ϵ, δ), it drops out. The server can abort the training process when there are not enough clients for training. Since the clients are chosen every round randomly, the client drop-out pattern satisfies a uniform distribution, which will not lead to an unbalanced FL model.

B. Federated Learning with Noise Resistance

In this part, two different models are proposed to improve the convergence performance under DP noise while maintaining the same protection level, namely as Fed-nore-1&2 , which are two different local training optimizers for FL by modifying the local objective function in two different ways. To the best of our knowledge, this is the first work to improve the convergence performance in terms of time cost and accuracy of the DP-based FL framework by modifying the local objective function with a noise-related term while maintaining the same privacy protection level.

In our proposed framework, we add a modification term related to the noise to the local objective function. Then, local models can offset the training loss caused by the noise through optimization. Meanwhile, the privacy protection level is not degraded since no changes are made to the DP mechanism settings. The proposed Fed-nore-1 and Fed-nore-2 share the same general FL protocol and the proposed LDP mechanism. However, they are different at the local training optimizer, where Fed-nore-1 is proposed to minimize the distance between the noisy gradients and the original ones, while Fed-nore-2 is proposed to minimize the expected loss created by the noise.

For Fed-nore-1, we consider the difference J between the noisy and the original gradients, which is computed as:

$$J = ||w_i^t - R(w_i^t)||_2 \tag{5}$$

$$= ||w_i^t - (w_i^t + N(0, C^2\sigma^2))||_2 \tag{6}$$

$$= ||N(0, C^2\sigma^2)||_2, \tag{7}$$

where J can then be simplified to $C\sigma$. By adding the difference between the original gradients and noisy ones into the local objective function, the local optimizer can reduce the distance between them in order to improve the accuracy performance. Meanwhile, as no changes are made to the DP mechanisms, the privacy protection level remains the same. We then give the formal implementation of Fed-nore-1, where the difference term is added to the local objective function as follows:

$$\arg\min_{W_i^t} h(W_i^t; W^t) = F(W_i^t) + \lambda_{nore1} \cdot C\sigma, \tag{8}$$

where λ_{nore1} is used to control the size of its effect.

Before introducing Fed-nore-2, the change in the loss is considered. When the noise is added to the gradients, the loss is added with a value, and the noisy gradients can be directly derived through the gradient descent from the new loss. In order to calculate the change, the backpropagation process of training is reversed. We first consider a normal DNN with ReLU as the activation function for hidden layers and the Sigmoid as the activation function for the output layer. During the backpropagation, the gradients are computed by taking the partial derivatives of the loss with respect to each parameter in the forward propagation as follows [27]:

$$dz_K = a_K - Y, \tag{9}$$

$$dz_k = da_k \times g_k'(z_k), \tag{10}$$

$$dw_k = \frac{1}{n'} dz_k \cdot (a_{k-1})^T, \tag{11}$$

$$db_k = \frac{1}{n'} \sum^{j=n'} dz_k^j, \tag{12}$$

$$da_{k-1} = (w_k)^T \cdot dz_k, \tag{13}$$

where K is the index of the final layer, db_k is the sum of the dz_k of every input, n' is the number of the input data, $\times$ means matrix-wise multiplication and $()^T$ is the transpose operation of matrix. The gradients are then used to update the model. Based on the formula (9), to calculate the expected change on the final loss, we need to calculate the expected change on each layer's $d\overline{z}$.

According to formulas (10)-(13) during the back-propagation, dz_i are used to obtain dz_{k-1} (only when $k > 1$), dw_k and db_k. Therefore, if the back-propagation is reversed, $d\overline{z}$ is computed with the expected change on dz_{k-1} (only when $k > 1$), and the noise on dw_k and db_k, which are computed in Lemma 1:

Lemma 1:

$$dz_k = dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g_k'(z_{k-1}), \tag{14}$$

$$d\overline{w_k} = N(w_k), \tag{15}$$

$$d\overline{b_k} = N(b_k), \tag{16}$$

$$d\overline{z_k} = N(w_k) \cdot a_{k-1} + N(b_k) + w_k \cdot d\overline{z_{k-1}} \times g_k'(z_{k-1}), \tag{17}$$

where the formula (14) describes the procedure of reversing the original gradients (without noise), and the formula (17) is the expression of the expected change on every layer. The proof of the Lemma 1 is presented in Appendix A.

Then, we propose the Theorem 1 of calculating the expected change on the final loss.

Theorem 1: If we apply DP through the Gaussian mechanism on FL, the noise added to the gradients can be regarded as an expected change added to the loss, which can directly derive the noisy gradients during the backpropagation. For a DNN with ReLU as the hidden layer's activation function and Sigmoid as the output layer's activation function, the expected change to the loss is scaled to itself, which is obtained by dividing the formula (14) by the formula (17) and the noise generation method discussed in Section III.A. Finally, we simplify the expected change on the loss to:

$$d\overline{a_K} = \frac{(a_K - Y) \cdot \sigma}{\sqrt{n}}. \tag{18}$$

The proof of the Theorem 1 is presented in Appendix B. With Theorem 1, Fed-nore-2 is proposed. To improve the accuracy performance, we assume that by subtracting the original loss from the expected change term, the new gradients with noise addition can reach the same performance as the original ones. Since we focus on the modification term as one term related to the parameters, we discard the term of Y. Therefore, as categorical-cross-entropy is used as the local loss function, the formal definition of the local objective function in Fed-nore-2 is modified as:

$$\arg\min_{w_i^t} h(w_i^t; w^t) = -ln(a_K^l * (1 - \lambda_{nore2}(\frac{\sigma}{\sqrt{n}})), \tag{19}$$

where l is the index of the correct label and λ_{nore2} is used to scale the proposed modification terms.

Then, we consider a CNN model with several convolution layers followed by max pooling (the hyper-parameters of these layers do not affect the results), a fully connected layer and ReLU activation and a final softmax output layer. Similar to DNN, we need to reverse the backpropagation. For the CNN model, the gradients of the fully connected layer are the same as the hidden layer in the DNN. With regards to the convolution layer and pooling layer, the gradients are obtained as follows:

$$dz_{k-1} = dz_k^{conv} * rot180(a_k) \times g_k'(z_{k-1}), \tag{20}$$

$$dw_k = dz_k^{conv} * a_{k-1}, \tag{21}$$

$$dz_{k-1} = upsample(dz_k^{pool}), \tag{22}$$

where the upsampling process means that the gradients dz_{k-1} of the largest parameter in every sub-region created in down-sampling is the same with dz_k, while the others are zeros. Then, we propose Corollary 1 to compute the expected change in the loss due to the noise on the convolution layers.

Corollary 1: If we apply DP through the Gaussian mechanism on a regular CNN model, there is an expected change in the final loss. With Theorem 1, the expected change on the convolution layer is computed as follows:

$$\frac{d\overline{z_k}}{dz_k} = \frac{\sigma}{\sqrt{n}}. \tag{23}$$

The proof of the Corollary 1 is presented in Appendix C.

The expected change of the max pooling layer is the same with the added noise, and the one of the fully connected layers is similar to DNN. Therefore, with Theorem 1 and Corollary 1, the expected change in the loss of CNN can be formalized as the same with formula (19).

C. Proposed framework

In this part, the proposed framework with DP through Gaussian Mechanism and Fed-nore-1&2 is introduced in Algorithm 1. At first, the server initializes the FL training and creates an initial model W^0. In this part, every client can choose their privacy budget and base noise variance σ on the purpose of personalized privacy protection level. Then, in each round, all the clients check for their remaining privacy budget and drop out of training if it runs out. After that, the server randomly selects m clients from the remaining clients, and broadcasts the model to the selected clients. Next, the selected clients use their local data to train the global model with Fed-nore. To be specific, the local clients train the global model with the local optimizer following formula (8) in Fed-nore-1 or following formula (19) in Fed-nore-2. The clients then calculate the gradients ∇W_i^t, clip the gradients, add noise and upload the noisy gradients $\nabla\overline{W_i^t}$ to the server. After receiving all the noisy gradients, the server aggregates and averages the gradients to obtain a new model. The server and clients repeat the above procedures until the global model reaches an acceptable accuracy or the server cannot find enough clients with remaining privacy budgets for the training process.

Algorithm 1 LDP-FL with Fed-nore-1&2

```
procedure SERVER
    Generate a global model $W^0$, the number of remaining clients $\overline{M}^0$ and privacy budget for clients $(\epsilon, \delta)$
    for round $t = 0, 1, 2...$ do
        $\overline{M}^t \leftarrow$ DP-client
        if remaining clients are not enough then
            return $W^{t-1}$
        Select a list of $M$ clients as $M^t$
        for all Client $i$ in $M^t$ do
            $\nabla\ W_i^{t+1} \leftarrow$ Fed-nore-client($i$,$W_t$)
        $\nabla W^{t+1} = \frac{1}{M}\sum_{i=1}^{M^t} \nabla\overline{W_i^{t+1}}$
procedure DP-CLIENT
    for every client $k$ in $M$ do
        Calculate its privacy loss based on the number of its participated communication rounds
        if the privacy loss $\leq \epsilon$ then
            Client $k$ drops out the training
    return remaining clients $\overline{M}^t$
procedure FED-NORE-CLIENT($t$,$i$,$W^t$)
    if Fed-nore-1 then E epochs of
        $W_i^t = argmin_{W_i^t} F(W_i^t) + \lambda_{nore1} * C\sigma$
    if Fed-nore-2 then E epochs of
        $W_i^t = argmin_{W_i^t}[-ln(a_K^l * (1 - \lambda_{nore2}(\frac{\sigma}{\sqrt{n}})))]$
    $\nabla W_i^{t+1} = W^t - W_i^t$
    Gradients clipping
    $\nabla W_i^{t+1} += N(0, C_{client_i}^2 \sigma_{client_i}^2)$
    return $\nabla W_i^{t+1}$
```

D. Complexity analysis

In this subsection, we discuss the difference in complexity among our proposed frameworks, traditional FL (Fed-Avg) and traditional DP-FL (without Fed-nore), for one local client. First, compared with Fed-Avg, the major changes in traditional DP-FL are gradient clipping and noise computing. As for gradients clipping, the weights need to be clipped element-wise as $\nabla W_i^t = \nabla W_i^t / max(1, \frac{||\nabla W_i^t||_2}{C})$ so that the complexity would be $O(size(\nabla W_i^t))$. Meanwhile, the C is computed as $C = \frac{||\nabla W_i^t||_2}{n} = \frac{||W^{t-1} - W_i^t||_2}{n}$, which brings $O(size(\nabla W_i^t))$ time. Second, the noise generates as $N(0, C^2\sigma^2)$, which takes $O(1)$ time. Third, adding the noise to the gradients as $\nabla W_i^{t+1} += N$ will take $O(size(\nabla W_i^t))$ time. In addition, the difference between our proposed work and traditional DP-FL is only the modification of the local objective function, which only takes $O(1)$ time.

In conclusion, the complexity of our proposed algorithm is dominated by the size of the model parameters, represented as $O(size(\nabla W_i^t))$.

V. Convergence analysis for the proposed models

In this section, we present the theoretical convergence guarantee for our proposed models. We analyze the expectation of the decrease in the loss function and then the convergence bound for the models. For the Fed-nore-2, as we compute the expectation of the effect of the noise on the loss and eliminate the effect during optimizing, the convergence bound of Fed-nore-2 is expected to be the same with FL without noise. Therefore, we only focus on the convergence bound for the Fed-nore-1 in this part.

For the derivation, we first make Assumption 1 for our proposed model:

Assumption 1:

(a) $F_i(W)$ is $\beta - Lipschitz$, implying that $||\nabla F_i(W)|| \le \beta$;

(b) $F(W)$ satisfies Polyak-Lojasiewicz condition with the positive parameter mu, implying that $F(W) - F(W^*) \le \frac{1}{2\mu}||\nabla F(W)||^2$, where W^* is the optimal solution;

(c) $F_i(W)$ is $\rho - Lipschitz$ smooth, implying that $||\nabla F_i(W) - \nabla F_i(W')|| \le \rho||w - w'||$.

in which $F()$ is the global loss function and computed as $F() = \sum \frac{F_i()}{m}$.

Based on Assumption 1, we can first obtain the expected decrease in the global loss function for one round of training.

Lemma 2: The expected decrease of the loss for the global loss function in one round is given as follows:

$$E\{F(\overline{W}^{t+1}) - F(\overline{W}^{t})\} \le (\frac{\rho k^2}{2} - k)||\nabla F||^2 + (1 - k\rho)||\nabla F||E\{||N||\} + \frac{\rho}{2}E\{||N||^2\}, \tag{24}$$

where:

$$k = \frac{1}{1 + \frac{\lambda\sigma\sqrt{n}}{C}}. \tag{25}$$

The proof of Lemma 2 is presented in Appendix D.

Then, we assume the noise generated in all the rounds shares the same bound value since they are generated in the same and independent way, and the $F()$ is convex. By using Lemma 2 and Assumption 1, the convergence of Fed-nore-1 is upper bounded after T communication by:

Theorem 2: After Tth communication of FL training with noise and Fed-nore-1 as local loss function, the convergence upper bound of the proposed model is presented as:

$$E\{F(\overline{W}^{t+1}) - F(W^*)\} \le l^T E\{F(\overline{W}^{0}) - F(W^*)\} + (\beta q(1 - k\rho) + \frac{\rho}{2}q^2) * \frac{(1 - l^T)}{1 - l}, \tag{26}$$

where:

$$l = (\mu\rho k^2 - 2k\mu + 1), \tag{27}$$

$$q = \frac{\sigma}{\sqrt{n}}(\lambda * \sigma n^{\frac{3}{2}} - \beta). \tag{28}$$

The proof of Theorem 2 is presented in Appendix E.

In addition, we consider that $F()$ is non-convex, we have the following convergence analysis:

Theorem 3: If $F()$ is non-convex and ρ-Lipschitz smooth, we have the following bound after T communication round of FL:

$$E||\nabla F||^2 \le \frac{F(W^0) - F(W^*)}{(k - \frac{\rho k^2}{2}) * T} + \frac{\beta q(1 - k\rho) + \frac{\rho q^2}{2}}{(k - \frac{\rho k^2}{2})}. \tag{29}$$

The proof of Theorem 3 is shown in Appendix F.

VI. Results and discussions

In this section, to validate the convergence performance of our proposed models, multiple simulations of both Fed-nore-1&2 are performed with the MNIST (a dataset hand-written number image with 60000 training data and 10000 testing data) [28]. In this paper, we perform our FL with 100 simulated clients, and the training data is categorized by class and divided through a Non-IID way into 200 shards, while each shard contains the data with the same label. Then, each client is assigned two shards with different classes. To evaluate the performance, we deploy Fed-nore-1&2 with different initial learning rates and different λ values, where the learning rate will decay by 0.96 for the first 20 round and be fixed after that. For the DP mechanism, we use $\delta = 1\text{e}{-6}$ for all the simulations. We use an RDP-based privacy analysis framework [29] to keep track of privacy loss and calculate the final privacy parameters.

We study the Fed-nore on two models in our paper. The first is a Multi-layer perceptron (MLP) with two hidden layers (each layer has 200 hidden units) with ReLU activation and an output layer with Softmax activation, which is optimized by SGD. The MLP shares the same feed-forwarding and backpropagation rules with basic DNN, which makes the proposed Fed-nore-2 of DNN work on MLP. The second one is a CNN model with two 5×5 convolution layers (the first one with 32 channels and the second with 64 channels, both followed by a 2×2 max-pooling layer) and a fully connected layer with Softmax activation. In each communication round, 50% of the clients are selected, and each client optimizes the global model with the corresponding local loss function for 10 epochs. Besides, due to the randomness of the noise generation and training, all the figures are processed in the same way (smoothed averages of multiple simulations of the same hyper-parameters). To show the effectiveness of our results, we provide the results of the plain DP-FL as a baseline model, which is the original DP-FL model without our improvements. We truncate the training curve after 52 rounds, where the increase in accuracy is marginal.

A. The performance of Fed-nore on DNN

In this subsection, the performance of Fed-nore-1 on the MLP is first presented, where we evaluate Fed-nore-1 with different λ_{nore1} values $(0.1, 1, 25)$.

The results of the Fed-nore-1 are compared with the plain DP-FL under the same hyper-parameters settings, where the plain DP-FL has an accuracy performance of 95.8% around 50th round. The results in Fig. 1 show that when the λ_{nore1} value is larger than 0, our proposed framework can improve the accuracy performance under DP-noise. In addition, when the λ_{nore1} is 1, its accuracy performance reaches the highest, 96.2%. However, when λ_{nore1} is getting larger, the performance is the same with the plain DP-FL and even worse.

Then, the performance of the Fed-nore-2 on the DNN is evaluated where we test the model with λ_{nore2} in

$(0.1, 25, 200)$. As shown in Fig. 2, the accuracy performance of Fed-nore-2 outperforms the plain DP-FL when the selected λ_{nore2} is larger than 0. Meanwhile, as λ_{nore2} is increased, the improvement of the Fed-nore-2 on the accuracy performance becomes better, and the overall training accuracy is more stable, which means that the noise has a smaller effect on the accuracy performance. It is shown that our Fed-nore-2 has the best accuracy performance of 96.3% when λ_{nore2} is 200. Meanwhile, it can reach 95.8% in the 38rd round, which means that our proposed Fed-nore-2 can save up to 30% of the communication and computation cost compared with the plain DP-FL and 5% compared to Fed-nore-1 even though the best accuracy performance of it is lower than the one of the Fed-nore-1. The simulations show that when the λ_{nore2} is larger than 200, the accuracy performance decreases. Since the FL is deployed on many Internet of Things devices having limited bandwidth, computational capability and power [6], [30], our Fed-nore-2 can converge faster and reach an acceptable accuracy performance while saving a huge amount of communication cost.

Meanwhile, we have evaluated Fed-nore-1&2 with different base noise variances to show their robustness by choosing σ in the range of $(4, 6, 10, 12, 16, 24, 40)$. After calculating the communication round with RDP for corresponding σ, we present only the first 52 rounds (where the base noise variance is 8) of the training results for those that have more training rounds for comparison with the previous results. When the base noise variance is smaller than 8, the communication round for FL is much smaller, leading to the FL not converging so that the accuracy performance is very bad. It is shown in Fig. 3 that under a small noise, our Fed-nore-1 can not improve the accuracy performance. Besides, it is shown that with a larger base noise variance (compared to the previous results) and an optimal scaling factor, Fed-nore-1 can perform much better than the plain DP-FL. For Fed-nore-2, it is shown in Fig. 4 that our proposed Fed-nore-2 can greatly improve the accuracy performance compared to the plain DP-FL when the base noise variance is small. However, the Fed-nore-2 performs worse with an increasing base noise variance than the Fed-nore-1, while it is still better than the plain DP-FL.

B. The performance of Fed-nore on CNN

In this subsection, the performance of the Fed-nore-1&2 on the mentioned CNN model is demonstrated. Since the noise is positively related to the learning rate (a larger learning rate brings a larger l_2-norm value), we present the Fed-nore-1&2 with different learning rates to show their performance. In this simulation, we choose the learning rate from $0.01, 0.1$ and 1. We first present the results for our Fed-nore-1 on CNN with the learning rates of 0.01 and 1. As shown in Fig. 5, Fed-nore-1 can slightly improve the convergence performance and accuracy performance for CNN with all the initial learning rates when the scaling factor is set as ten. Then, we conduct the simulation with an initial learning rate of 0.1 and different scaling factors to further test its effectiveness. It is shown in Fig. 6 that our Fed-nore-1 has a limited effect on the CNN model by only improving the accuracy performance by 0.07% with the optimal settings. Next, we evaluate Fed-nore-2 with an initial learning rate of 0.1 and different scaling factors. As shown in Fig. 7, our Fed-nore-2 can improve the accuracy performance than the plain one when the scaling factor is larger than 0.1. Meanwhile, it is found that when the scaling factor is set to 10, the improvement is the best, and it can save up to 40% of communication rounds to achieve the same results with the plain DP-FL. We also test our Fed-nore-2 with different initial learning rates. The results in Fig. 8 show that our Fed-nore-2 can also improve the accuracy performance with an initial learning rate of 0.01 compared with the plain one. However, when the learning rate increases, our Fed-nore-2 performs worse than the plain DP-FL.

In addition, we have performed simulations with CIFAR-10 on the same CNN model as previously. The data are assigned

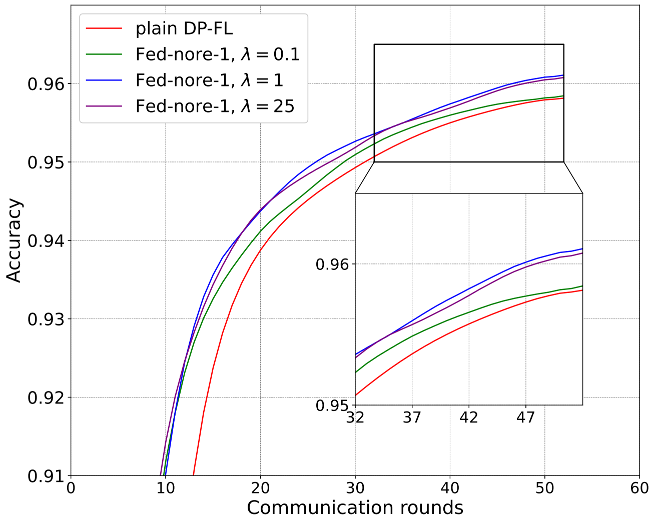


Fig. 1. Accuracy performance of the Fed-nore-1 on DNN of different scaling factor (λ) compared with the plain DP-FL.

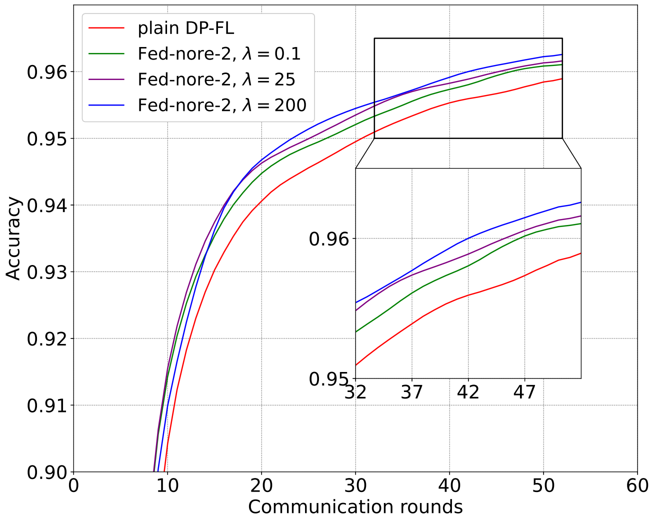


Fig. 2. Accuracy performance of the Fed-nore-2 on DNN of different scaling factor (λ) compared with the plain DP-FL.

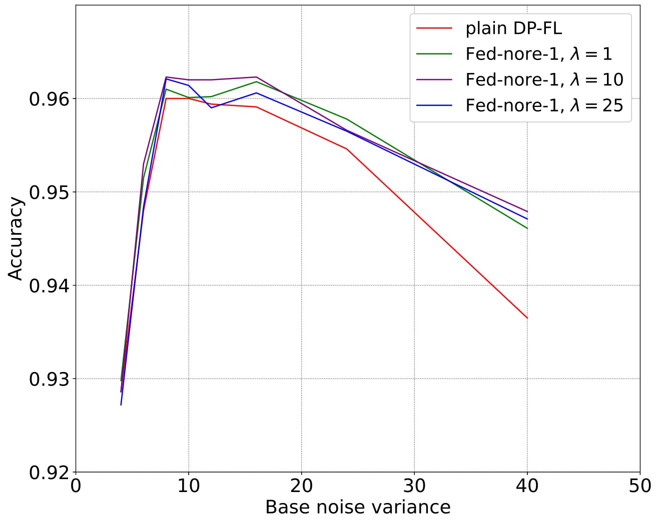


Fig. 3. Accuracy performance of the Fed-nore-1 on DNN of different base noise variance and scaling factor (λ) compared with the plain DP-FL.

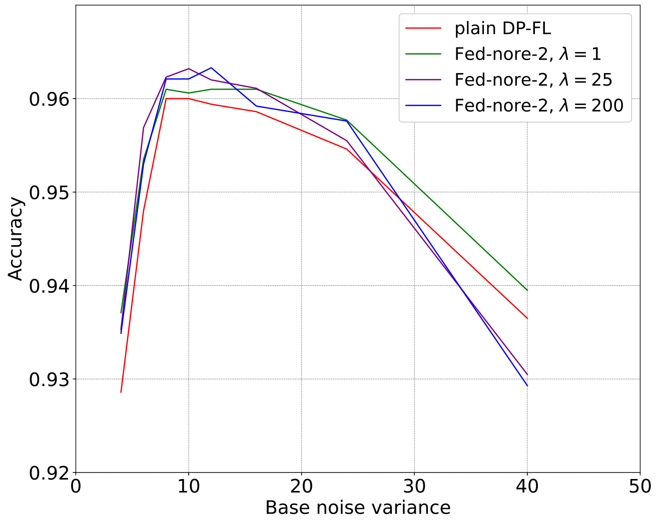


Fig. 4. Accuracy performance of the Fed-nore-2 on DNN of different base noise variance and different scaling factor (λ) compared with the plain DP-FL.

to 100 clients in the same way as MNIST. We present the best accuracy of our proposed frameworks and plain DP-FL within 50 communication rounds with different settings in Table II, which shows that our proposed framework can increase the test accuracy from 48.7% for plain DP-FL to 52.0% for Fed-nore-1 and 53.7% for Fed-nore-2. Meanwhile, both Fed-nore-1&2 reach the highest accuracy with 10% fewer communication rounds than plain DP-FL. Therefore, our proposed Fed-nore-1&2 show a greater improvement on more complicated datasets.

Moreover, we implement two strategies on CNN with CIFAR-10 at the same time with an initial learning rate of 0.1, base sigma of 8 and several scaling factors. However, the accuracy performance is worse than using only one, as

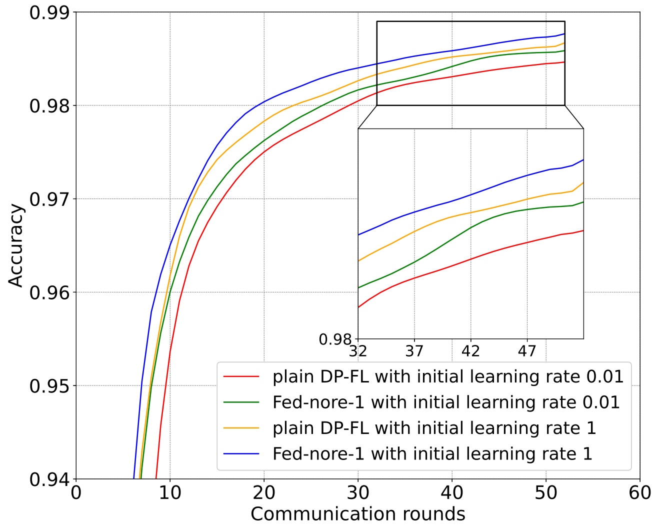


Fig. 5. Accuracy performance of the Fed-nore-1 with a scaling factor of 10 and of different learning rates compared with the plain DP-FL.

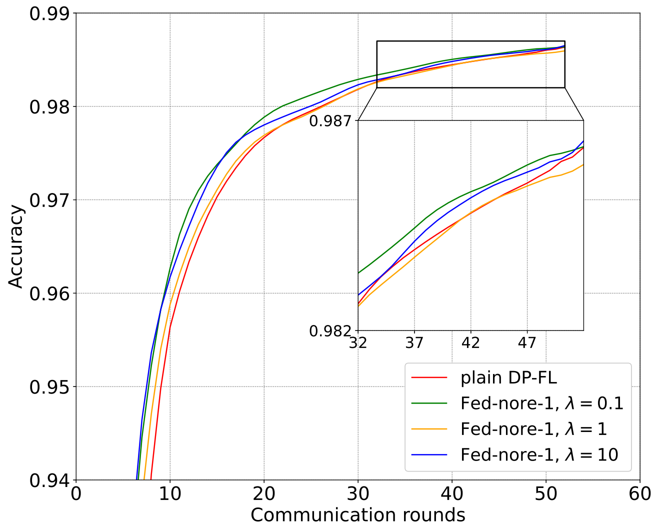


Fig. 6. Accuracy performance of the Fed-nore-1 of different scaling factors (λ) compared with the plain DP-FL.

shown in Table II, and it also takes a longer time to reach the same accuracy with the original DP-FL than using only one strategy. A possible reason for this phenomenon is that they may interfere with each other during local training.

C. Performance comparison and discussion

The performance of the proposed framework is compared with the plain DP-FL (the DP-FL framework uses the original local loss function all the time), Fed-Avg and some well-known DP-based learning in Table III, where R means the communication rounds in FL and epochs in ML, SR means the rate of clients selected in each round and ACC means the accuracy performance. We provide our proposed framework with the best accuracy performance results, where the results in the blanket show the communication round to reach the

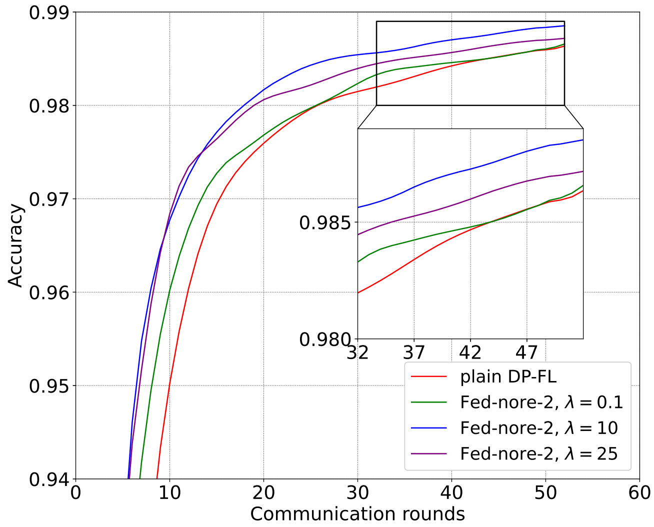


Fig. 7. Accuracy performance of the Fed-nore-2 of different scaling factors (λ) compared with the plain DP-FL.

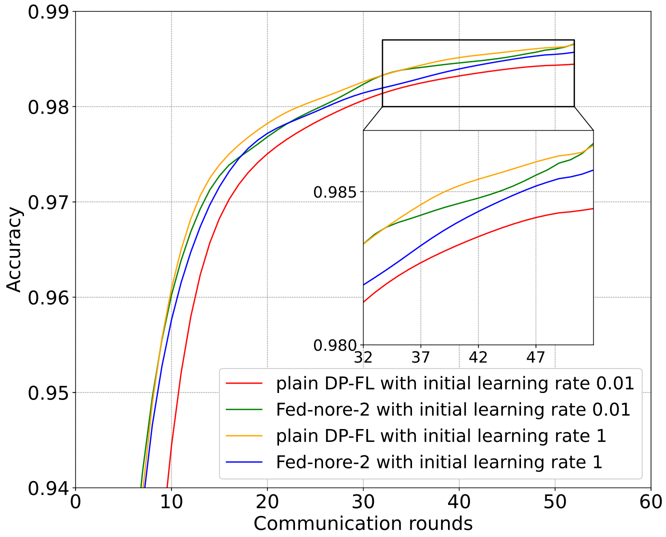


Fig. 8. Accuracy performance of the Fed-nore-2 with a scaling factor of 10 and different learning rates compared with the plain DP-FL.

same accuracy performance with the plain DP-FL. Meanwhile, unlike the average results in all Figures, Table III contains the best results under the optimal settings, where the base noise variance is eight. It is shown in Table III that our proposed Fed-nore-1 can slightly improve the accuracy performance for the DNN and CNN while the Fed-nore-2 can save up to 40% of the training time to reach the same results for both models and also provides a much better accuracy performance compared with Fed-nore-1.

By comparing the presented figures, we can also find that the Fed-nore-1 works better when the noise variance is large (large base noise variance and large learning rate), while Fed-nore-2 shows it has much better performance when the value of the noise variance is moderate.

Table II: This table shows the best accuracy comparisons among the plain DP-FL and our proposed frameworks with CIFAR-10.

Algorithm	Scaling factor	Accuracy
Fed-nore-1	1	49.6%
	10	49.9%
	20	**52.0%**
	200	50.9%
Fed-nore-2	0.1	51.1%
	1	52.4%
	10	**53.7%**
	200	50.9%
Fed-nore-1&2	10	49.0%
	20	50.5%
Plain DP-FL	0	48.7%

Table III: This table shows convergence performance comparisons among other well-known privacy-preserving ML, the plain DP-FL and our proposed framework with MNIST.

Framework	R	SR	ACC	DP budget
Fed-Avg	380	1	97%	
DP-SGD(MLP) [14]	700		97%	(8,1e-5)-DP
CDP-FL(MLP) [16]	11	0.5	78%	(0.5,1e-3)-DP
LDP-FL(CNN) [17]	10	1	95.36%	(0.5)-DP
Plain DP-FL(MLP)	52	0.5	95.9%	(0.27,63)-RDP
Plain DP-FL(CNN)	52	0.5	98.7%	(0.27,63)-RDP
Fed-nore-1(DNN)	52(42)	0.5	96.4%(96%)	(0.27,63)-RDP
Fed-nore-2(DNN)	52(33)	0.5	**96.7**%(96%)	(0.27,63)-RDP
Fed-nore-1(CNN)	52(49)	0.5	98.77%(98.7%)	(0.27,63)-RDP
Fed-nore-2(CNN)	52(32)	0.5	**99**%(98.7%)	(0.27,63)-RDP

VII. Conclusion

In this paper, we propose a novel FL framework to enhance privacy protection in FL and its convergence performance in terms of accuracy and time consumption. We first propose an LDP-FL scheme by adding Gaussian noise on the local gradients before uploading to satisfy RDP. Second, to improve its performance, we propose two modifications on the local objective function and detailed derivations to improve the accuracy performance of the noisy training, namely Fed-nore-1&2. The first one is to calculate the difference between the noisy gradients and the original ones and add the difference value to the local objective function, hence minimizing the difference during the training under the same DP protection level. The other one is to calculate the expected change in the final loss due to the noise by reversing the back-propagation process in the ML training. Then, by modifying with the expected change in the local objective function, the FL can also minimize the loss created by noise and converge faster. Besides, both modification terms are controlled by a scale value. Finally, we conduct multiple simulations on DNN and CNN with the corresponding modification to show the effectiveness of our proposed frameworks. For both CNN and DNN, the results show that, compared to original DP-FL, our Fed-nore-1&-2 can both increase the accuracy performance and greatly increase the convergence time under different magnitudes of the noise with an appropriate scale value. To be specific, Fed-nore-2 can save up to 40% communication rounds to reach the same accuracy results with the plain DP-FL under optimal settings. On the other hand, when using CNN as the training model, Fed-nore-2 also has a higher accuracy performance than Fed-nore-1 for most scenarios. Besides, the improvement of CIFAR-10 on accuracy is greater than the one of MNIST. Furthermore, we find that Fed-nore-1 works better with a relatively larger noise, while Fed-nore-2 works better when the noise is small.

For future work, the scalability of our proposed framework needs to be further studied. In addition, in this paper, we only propose Fed-nore-2 for two types of ML models, so our derivations of the two modifications on other widely used models, including RNN, GNN and GCN, could be explored to provide a more general implementation of the proposed framework, which is expected to be similar to the currently proposed method.

Appendix

A. Proof of Lemma 1

To obtain the expected change on dz_k, we first obtain dz_k by reversing the backpropagation. Then we can obtain the expected change dz_k generated bt dw_k and db_k correspondingly, which is noted as $d_{dw_k}z_k$ and $d_{db_k}z_k$. Based on formula (11), we can get:

$$dw_k = \frac{1}{n'} dz_k \cdot (a_{k-1}^T), \tag{30}$$

$$dw_k \cdot a_{k-1} = \frac{1}{n'} dz_k \cdot (a_{k-1}^T) \cdot a_{k-1}, \tag{31}$$

$$d_{dw_k}z_k = dw_k \cdot a_{k-1}. \tag{32}$$

Then, as formula (12) is a vectorization implementation for the DNN training and we can not directly obtain $d_{db_k}z_k$, we obtain an approximate value as $d_{db_k}z_k = db_k$. Therefore, for the first layer, we can obtain its dz_k through reverse backpropagation as:

$$\begin{aligned} dz_k &= d_{dw_k}z_k + d_{db_k}z_k & (33)\\ &= dw_k \cdot a_{k-1} + db_k. & (34) \end{aligned}$$

After that, for all the following layers, to compute the reversed dz_k, we also need to consider the derivative part of the dz_{k-1}. Based on the formulas (10)-(13), we can get that:

$$\begin{aligned} dz_{k-1} &= da_{k-1} \times g'_{k-1}(z_{k-1}) & (35)\\ &= (w_k)^T \cdot dz_k \times g'_{k-1}(z_{k-1}) & (36)\\ w_k \cdot dz_{k-1} &= dz_k \times g'_{k-1}(z_{k-1}). & (37) \end{aligned}$$

Next, we need to simplify the multiplication term $g'_{k-1}(z_{k-1})$, where g() is the activation function. For all the hidden layers (except the first one), we use ReLU activation in this paper. Therefore $g'(z)$ is a function of following,

$$g'(z) = \begin{cases} 0 & z < 0, \\ 1 & others. \end{cases} \tag{38}$$

In this case, for all the elements in z_{k-1}, whose values are not larger than zero, the dz_k has no effect on these elements so that the reversed dz_k from dz_{k-1} is zero. Then, for all the elements in z_{k-1}, whose values are larger than zero, the reversed dz_k of these elements are the same as the values of the corresponding elements of dz_{k-1}. To formalize, the expected changes dz_k from dz_{k-1}, noted as $d_{dz_{k-1}}z_k$ is computed equivalent as followed:

$$d_{dz_{k-1}}z_k = w_k \cdot dz_{k-1} \times g'_{k-1}(z_{k-1}). \tag{39}$$

Then, for all the hidden layers (except the first one), the dz_k is computed:

$$\begin{aligned} dz_k &= d_{dz_{k-1}}z_k + d_{w_k}z_k + d_{b_k}z_k & (40)\\ &= dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g'_{k-1}(z_{k-1}). & (41) \end{aligned}$$

Finally, to obtain the expected change in the noisy gradients on the loss, we need to substitute the expected changes of dw_k, db_k and dz_{k-1} into formula (41), where the expected changes of dw_k and db_k are the noise added on the gradients and the expected changes of dz_{k-1} is obtained by iterative calculation.

B. Proof of Theorem 1

To find the expected change in the final loss function, we first consider the relation between formulas (14) and (17) as follows:

$$\frac{dz_k}{d\overline{z_k}} = \frac{dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g'_{k-1}(z_{k-1})}{N(dw_k) \cdot a_{k-1} + N(db_k) + w_k \cdot d\overline{z_{k-1}} \times g'_k(z_{k-1})}, \tag{42}$$

Then we generate the noise as follows:

$$N(\nabla W) = N(0, C^2\sigma^2), \tag{43}$$

where C is calculated in formula (3) as $\frac{||\nabla W||}{n}$ for every layer. Therefore, we can suffice to obtain the following:

$$\frac{N(dw_k)}{dw_k} = \frac{||N(dw_k)||}{||dw_k||} = \frac{\frac{||dw_k||*\sigma}{\sqrt{n}}}{||dw_k||} = \frac{\sigma}{\sqrt{n}}. \tag{44}$$

With the formula (9) and the constant label value Y, we now can finally obtain that the expected change of $da_K = \frac{\sigma}{\sqrt{n}} * (a_K - Y)$, which completes the proof.

C. Proof of Corollary 1

Similar to Corollary 1, the expected change, $d\overline{z_k}$, is also generated with the expected changes on dz_{k-1}, db_k and dw_k. We use the properties of the convolution process, $a * b \times c = a * (b \times c) = a * (b \times c)$, to obtain the corresponding expected change $d_{\overline{z_{k-1}}}z_k$ from the expected changes on $d\overline{z_{k-1}}$ as followed:

$$\begin{aligned} d\overline{z_{k-1}} &= dz_{k-1} \cdot \frac{d\overline{z_{k-1}}}{dz_{k-1}} & (45)\\ &= \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dz_k^{conv} * rot180(a_k) \times g'_k(z_{k-1}) & (46) \end{aligned}$$

$$d_{\overline{z_{k-1}}}z_k^{conv} = \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dz_k^{conv}. \tag{47}$$

Similarly, the corresponding expected change $d_{\overline{w_k}} z_k$ from the expected changes on $d\overline{w_k}$ can be computed as followed:

$$d^{conv}_{\overline{w_k}} z_k = \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dw_k. \tag{48}$$

Finally, by combining Corollary 1 and formulas (42) and (44), we can obtain the following:

$$d\overline{z_k} = \frac{\sigma}{\sqrt{n}} \cdot dz_k \tag{49}$$

D. Proof of Lemma 2

As shown in Algorithm 1, in one round of the training, we have the following:

$$h(w_i^t; w^t) = F_i(W_i) + \lambda * C\sigma, \tag{50}$$

$$\nabla h(w_i^t; w^t) = \nabla F_i(W_i) + \lambda * \sigma \nabla C, \tag{51}$$

$$W_i^{t+1} = W^t - \nabla h(w_i^t; w^t), \tag{52}$$

$$\nabla W_i^{t+1} = W_i^{t+1} - W^t, \tag{53}$$

$$\overline{W}^{t+1} = W^t + \frac{1}{m} \sum (\nabla W_i^{t+1} + N_i), \tag{54}$$

where η is the learning rate. By substituting formula (4) into (52), we have:

$$\nabla h(w_i^t; w^t) = \nabla F_i(W_i) + \lambda * \sigma \frac{W_i^{t+1} - W^t}{C}, \tag{55}$$

Since $F_i()$ is ρ-Lipschitz smooth as in Assumption 1, we can obtain the following:

$$\begin{aligned} F_i(\overline{W}^{t+1}) \leq F_i(\overline{W}^t) + \bigtriangledown(\overline{W}^t)^T(\overline{W}^{t+1} - \overline{W}^t) \\ + \frac{\rho}{2} ||\overline{W}^{t+1} - \overline{W}^t||^2 \end{aligned}, \tag{56}$$

for all the $\overline{W}^{t+1}$ and $\overline{W}^t$. Then for the global loss function, since the global model is the average of the local models. Therefore, we define $F(\overline{W}^t) = E\{F_i(\overline{W}^t)\}$ and $\nabla F(\overline{W^t}) = E\{\nabla F(\overline{W_i^t})\}$. Then we have the following:

$$\begin{aligned} E\{F(\overline{W}^{t+1}) - F(\overline{W}^t)\} \leq E\{\nabla (F(\overline{W^t}))^T (\overline{W}^{t+1} - \overline{W}^t)\} \\ + E\{\frac{\rho}{2} ||\overline{W}^{t+1} - \overline{W}^t||^2\}. \end{aligned} \tag{57}$$

Based on the formulas (51), (52) and (54), we have:

$$\begin{aligned} W^{t+1} - \overline{W}^t &= E\{-(\nabla F_i(W) + \lambda * C\sigma\sqrt{n})\} \\ &= -((\nabla F(\overline{W}^t) - E\{\lambda * \sigma\sqrt{n}\frac{W_i^{t+1} - W^t}{C}\}) \\ &= -\frac{\nabla F}{1 + \frac{\lambda\sigma\sqrt{n}}{C}}. \end{aligned} \tag{58}$$

By substituting formulas (54) and (58) into (57), we can obtain:

$$\begin{aligned} E\{F(\overline{W}^{t+1}) - F(\overline{W}^t)\} \leq E\{\nabla F^T(-\frac{\nabla F}{\eta + \frac{\lambda\sigma\sqrt{n}}{C}} + N)\} \\ + \frac{\rho}{2} E\{|| - \frac{\nabla F}{1 + \frac{\lambda\sigma\sqrt{n}}{C}} + N||^2\}. \end{aligned} \tag{59}$$

Then, using triangle inequation, we have:

$$\begin{aligned} E\{F(\overline{W}^{t+1}) - F(\overline{W}^t)\} \leq (\frac{\rho k^2}{2} - k)||\nabla F||^2 \\ + (1 - k\rho)||\nabla F||E\{||N||\} + \frac{\rho}{2}E\{||N||^2\}, \end{aligned} \tag{60}$$

where:

$$k = \frac{1}{1 + \frac{\lambda\sigma\sqrt{n}}{C}}, \tag{61}$$

E. Proof of Theorem 2

By subtracting $E\{F(W^*)\}$ on both sides of formula (60), we have the following:

$$\begin{aligned} & E\{F(\overline{W}^{t+1}) - F(W^*)\} \leq E\{F(\overline{W}^t) - F(W^*)\} \\ & + (\frac{\rho k^2}{2} - k)||\nabla F||^2 + (1 - k\rho)||\nabla F||E\{||N||\} \\ & + \frac{\rho}{2}E\{||N||^2\}, \end{aligned} \tag{62}$$

We know that $||\nabla F(W)|| \leq \beta$ and with formula (4), we can bound $||W^{t+1} - \overline{W}^T||$ as:

$$\begin{aligned} ||W^{t+1} - \overline{W}^T|| &= || - \nabla F(\overline{W}^t) - E\{\lambda * \sigma\sqrt{n}\frac{W_i^{t+1} - W^t}{C}\}|| \\ &\leq -\beta + \lambda * \sigma\sqrt{n}\frac{||W^{t+1} - W^t||}{C} \\ &\leq (\lambda * \sigma n^{\frac{3}{2}} - \beta). \end{aligned} \tag{63}$$

Then with the generating method in Algorithm 1, we can also obtain:

$$E\{||N||\} \leq \frac{\sigma}{\sqrt{n}}(\lambda * \sigma n^{\frac{3}{2}} - \beta). \tag{64}$$

Then by substituting formulas (58), (59), (63) and (64) into (62) and with $F(W) - F(W^*) \leq \frac{1}{2\mu}||\nabla F(W)||^2$, we know that:

$$\begin{aligned} E\{F(\overline{W}^{t+1}) - F(W^*)\} \leq l * E\{F(\overline{W}^t) - F(W^*)\} \\ + \beta q(1 - k\rho) + \frac{\rho}{2}q^2, \end{aligned} \tag{65}$$

where:

$$l = (\mu\rho k^2 - 2k\mu + 1), \tag{66}$$

$$q = \frac{\sigma}{\sqrt{n}}(\lambda * \sigma n^{\frac{3}{2}} - \beta). \tag{67}$$

Finally, since the noise is generated in the same and independent way, we assume the noise for all the communication rounds shares the same expected bound value. By repeating the formula (65) for T communication round, the convergence can be upper bounded as:

$$\begin{aligned} E\{F(\overline{W}^{t+1}) - F(W^*)\} \leq l^T E\{F(\overline{W}^0) - F(W^*)\} \\ + (\beta q(1 - k\rho) + \frac{\rho}{2}q^2) * \frac{(1 - l^T)}{1 - l}. \end{aligned} \tag{68}$$

F. Proof of Theorem 3

Based on formulas (60) and (64), we have the following :

$$F(W^{t+1}) \leq F(W^{t}) + (\frac{\rho k^2}{2} - k)||\nabla F||^2 + \beta q(1-k\rho) + \frac{\rho}{2}q^2. \tag{69}$$

By taking T iteration of formula (69), we have the following:

$$(k - \frac{\rho k^2}{2})\sum ||\nabla F||^2 \leq F(W^0) - F(W^*) + T\beta q(1-k\rho) + T\frac{\rho}{2}q^2, \tag{70}$$

which implies:

$$E||\nabla F||^2 \leq \frac{F(W^0) - F(W^*)}{(k - \frac{\rho k^2}{2}) * T} + \frac{\beta q(1-k\rho) + \frac{\rho q^2}{2}}{(k - \frac{\rho k^2}{2})}. \tag{71}$$

References

[1] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–5555, 2017.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. PMLR, 2017, pp. 1273–1282.

[3] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[4] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *arXiv preprint arXiv:1812.06127*, 2018.

[5] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5132–5143.

[6] Y. Gou, R. Wang, Z. Li, M. A. Imran, and L. Zhang, "Clustered hierarchical distributed federated learning," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 177–182.

[7] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[8] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients–how easy is it to break privacy in federated learning?" *arXiv preprint arXiv:2003.14053*, 2020.

[9] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 2019, pp. 2512–2520.

[10] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "{BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning," in *2020 USENIX annual technical conference (USENIX ATC 20)*, 2020, pp. 493–506.

[11] S. Weng, L. Zhang, D. Feng, C. Feng, R. Wang, P. V. Klaine, and M. A. Imran, "Privacy-preserving federated learning based on differential privacy and momentum gradient descent," in *2022 International Joint Conference on Neural Networks (IJCNN)*, 2022, pp. 1–6.

[12] 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.

[13] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy." *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[14] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[15] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, Aug. 2017. [Online]. Available: http://dx.doi.org/10.1109/CSF.2017.11

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

[17] L. Sun, J. Qian, and X. Chen, "Ldp-fl: Practical private aggregation in federated learning with local differential privacy," *arXiv preprint arXiv:2007.15789*, 2020.

[18] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2020.

[19] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[20] M. Naseri, J. Hayes, and E. D. Cristofaro, "Toward robustness and privacy in federated learning: Experimenting with local and central differential privacy," *arXiv preprint arXiv:2009.03561*, 2021.

[21] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM workshop on artificial intelligence and security*, 2019, pp. 1–11.

[22] M. Wu, D. Ye, J. Ding, Y. Guo, R. Yu, and M. Pan, "Incentivizing differentially private federated learning: A multidimensional contract approach," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 639–10 651, 2021.

[23] B. Jia, X. Zhang, J. Liu, Y. Zhang, K. Huang, and Y. Liang, "Blockchain-enabled federated learning data protection aggregation scheme with differential privacy and homomorphic encryption in iiot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4049–4058, 2022.

[24] 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.

[25] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, "Differentially private learning with adaptive clipping," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 455–17 466, 2021.

[26] V. Pichapati, A. T. Suresh, F. X. Yu, S. J. Reddi, and S. Kumar, "Adaclip: Adaptive clipping for private sgd," *arXiv preprint arXiv:1908.07643*, 2019.

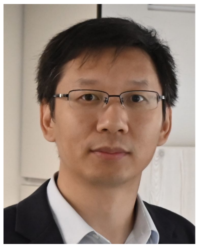
[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[28] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[29] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "{TensorFlow}: a system for {Large-Scale} machine learning," in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, 2016, pp. 265–283.

[30] L. Wang, W. Wang, and B. Li, "Cmfl: Mitigating communication overhead for federated learning," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 954–964.

Shangyin Weng (Student Member, IEEE) received his B.Eng. degree in Electronics and Electrical Engineering from the Glasgow college, the University of Electronic Science and Technology of China (UESTC), Chengdu, Sichuan, China, in 2020. He is currently pursuing the Ph.D. degree in Electronics and Electrical Engineering with the James Watt School of Engineering, University of Glasgow. His research interests include federated learning, distributed systems, and security and privacy protection.



Lei Zhang (Senior Member, IEEE) is a Professor of Trustworthy Systems with the University of Glasgow, Glasgow, U.K. He has academia and industry combined research experience on wireless communications and networks, and distributed systems for the Internet of Things, blockchain, and autonomous systems. His 20 patents are granted/filed in more than 30 countries/regions. He published three books, and more than 200 papers in peer-reviewed journals, conferences, and edited books.

Dr. Zhang received the IEEE ComSoc TAOS Technical Committee Best Paper Award in 2019, the IEEE Internet of Things Journal Best Paper Award in 2022, Digital Communications and Networks Journal Best Paper Award 2023 in addition to several best paper awards in IEEE conferences. He is the Founding Chair of the IEEE Special Interest Group on Wireless Blockchain Networks in the IEEE Cognitive Networks Technical Committee (TCCN). He delivered tutorials in IEEE ICC'20, IEEE PIMRC'20, IEEE Globecom'21, IEEE VTC'21 Fall, IEEE ICBC'21, EUSIPCO'21, and IEEE Globecom'22. He is an Associate Editor of IEEE Internet of Things Journal, IEEE Wireless Communications Letters, IEEE Transactions on Network Science and Engineering, and Digital Communications and Networks.

Xiaoshuai Zhang received the Ph.D. degree in Computer Science from Queen Mary University of London (United Kingdom) in 2020.

He has recently joined the Ocean University of China as an associate professor. He is also a delegate of Permissioned Distributed Ledger work group in European Telecommunications Standards Institute. He was a Research Associate with the James Watt School of Engineering, University of Glasgow, Glasgow, U.K. He has authored or coauthored more than 30 papers in peer-reviewed journals and conferences and has finished several UKRI EPSRC and industrial projects as the lead researcher. Dr. Zhang is a TPC member of IEEE GLOBECOM'21, IEEE ICC'23 and ICC'24. Meanwhile, he is the publicity co-chair of 2024 IEEE Global Blockchain Conference. His current research interests include blockchain systems, distributed consensus, applied cryptography, and privacy preservation.

Muhammad Ali Imran (Fellow, IEEE) is a Professor of Wireless Communication Systems and Dean of Graduate Studies in College of Science and Engineering. His research interests include self-organized networks, wireless networked control systems, and the wireless sensor systems. He heads the Communications, Sensing and Imaging CSI Hub, University of Glasgow, Glasgow, U.K. He is also an Affiliate Professor with The University of Oklahoma, Norman, OK, USA, and a Visiting Professor with the 5G Innovation Centre, University of Surrey, Guildford, U.K. He has more than 20 years of combined academic and industry experience with several leading roles in multimillion pounds funded projects. He has filed 15 patents and has authored/co-authored more than 600 journal and conference publications. He is author/editor of 13 books and author of more than 20 book chapters. He has successfully supervised more than 50 postgraduate students at doctoral level. He has been a consultant to international projects and local companies in the area of self-organized networks. His research interests include self-organized networks, wireless networked control systems, and the wireless sensor systems.

Prof. Imran is a Fellow of the esteemed societies and institutions like IEEE, RSE, IET, EAI and a Senior Fellow of the HEA.