Privacy-Preserving Federated Learning based on Differential Privacy and Momentum Gradient Descent

Shangyin Weng*, Lei Zhang*, Daquan Feng†, Chenyuan Feng‡, Ruiyu Wang*, Paulo Valente Klaine*,
Muhammad Ali Imran*

* School of Engineering, University of Glasgow, Glasgow, G12 8QQ, UK
Email: {2289305W}@student.gla.ac.uk, {r.wang.1}@research.gla.ac.uk,
{Lei.Zhang, Paulo.ValenteKlaine, Muhammad.Imran}@glasgow.ac.uk
† College of Electronics and Information Engineering, Shenzhen University, 518060, Shenzhen, China
Email: fdquan@szu.edu.cn
‡ Information Systems Technology and Design, Singapore University of Technology and Design, 487372, Singapore
Email: chenyuan_feng@mymail.sutd.edu.sg

***Abstract*—To preserve participants' privacy, Federated Learning (FL) has been proposed to let participants collaboratively train a global model by sharing their training gradients instead of their raw data. However, several studies have shown that conventional FL is insufficient to protect privacy from adversaries, as even from gradients, useful information can still be recovered. To obtain stronger privacy protection, Differential Privacy (DP) has been proposed on the server's side and the clients' side. Although adding artificial noise to the raw data can enhance users' privacy, the accuracy performance of the FL is inevitably degraded. In addition, although the communication overhead caused by the FL is much smaller than that of centralized learning, it still becomes a bottleneck of the learning performance and utilization efficiency due to its frequent parameters exchange. To tackle these problems, we propose a new FL framework via applying DP both locally and centrally in order to strengthen the protection of participants' privacy. To improve the accuracy performance of the model, we also apply sparse gradients and Momentum Gradient Descent on the server's side and the clients' side. Moreover, using sparse gradients can reduce the total communication costs. We provide the experiments to evaluate our proposed framework and the results show that our framework not only outperforms other DP-based FL frameworks in terms of the model accuracy but also provides a more powerful privacy guarantee. Besides, our framework can save up to 90% of communication costs while achieving the best accuracy performance.**

***Index Terms*—Privacy-preserving federated learning, differential privacy, momentum gradient descent, gradients sparsification.**

I. Introduction

With the development of Machine Learning (ML) techniques and increasing demand for data communication and processing, data privacy has drawn significant attention from academics and industries. Since the performance of ML models is determined by the quality and amount of the training data, there are two urgent problems that need to be solved to achieve secure data exchange [1]. Firstly, some service providers of network applications would like to collect users' data to train their algorithms, such as recommending systems, on the purpose of providing better experiences for customers. Some users may consider that this action violates their privacy and refuse to share their sensitive data. As a result, ML training becomes more problematic, as it is usually unrealistic to synthesize a large amount of reasonable data similar to those of real users. Secondly, the issue of "data silos" is becoming more and more serious. Specifically, different companies or different departments in the same company may refuse to share their own data because of the lack of effective methods to share data without privacy leakage, as exchanging those data is a privacy violation of their own customers [1]. Since it is impossible to obtain a good ML model without a large amount of data, novel and secure ways of sharing data are needed.

To solve the problem of secure data sharing, Google firstly proposes Federated Learning (FL), which is a distributed ML system without centralizing users' data to train a smart keyboard application[1]. Shortly after that, it is applied to financial, medical or industrial systems [2]–[5]. The key idea of FL is to train a model with users' data locally and transmit intermediate gradients to the server, which are aggregated later in order to generate a global model. In this way, a well-performed model can be achieved and users' data privacy is not broken as users' data never leave their local storage. However, FL still suffers from privacy leakage, heavy communication costs, and statistical heterogeneity [6].

By training in a distributed manner, the server and the clients need frequent model exchanges, which brings huge communication costs. To improve accuracy performance and reduce communication costs of FL, several solutions have been proposed. Fed-Avg is first proposed to use Stochastic Gradient Descent (SGD) to train local models for multiple epochs before transmitting gradients [7]. However, Fed-Avg

[1] https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

has unsatisfactory convergence performance when the training data is Non-Independent and Identically Distributed (Non-IID). Fed-Prox [8] and SCAFFOLD [9] are proposed to optimize the convergence rate of FL with Non-IID data by forcing all local clients' models closer to the global model. In addition, to improve the convergence performance, Adam and Momentum Gradient Descent (MGD) are applied in FL to train local models [10], [11]. All these algorithms are trying to reduce communication costs by accelerating convergence so that the number of the total communication rounds decreases. On the other hand, the amount of data exchanged in a single round can be reduced to save costs through quantization or sparsification [12].

Even though FL is applied to protect users' privacy, studies have shown that useful information can be recovered from gradients, such as in [13], where the authors successfully recover high-resolution images from gradients of a deep neural network. As such, the concept of Differential Privacy (DP) is adopted to enhance privacy protection in FL systems. Authors in [14] first apply DP in single-end ML and derive a tighter bound for calculating privacy losses, which is Moment Accountant (MA). Then, DP is used in the central server of FL, referred to as Central-DP (CDP), by adding artificial noise on the aggregated gradients to hide a single client's contribution from the others [15]. In this way, malicious clients can not know whether a certain client has participated in the last training round or not. In their settings, the server is assumed to be honest. However, central servers are not always trustworthy in reality. Therefore, the authors in [16] implement Local-DP (LDP) on clients by perturbing original gradients before uploading to the server, in order to prevent the server from recovering useful data. In [17], [18], the authors apply LDP and CDP to FL, but they mainly focus on utility, which leads to weak privacy protection. Meanwhile, adding noise to the gradients inevitably degrade the accuracy performance.

As mentioned above, only using one type of DP in FL can protect privacy from either malicious clients or the server. Motivated by these issues, in this paper, we propose a Privacy-Preserving FL (PPFL) framework which achieves both LDP and CDP by adding noise and keeps track of the privacy losses by the MA scheme. Furthermore, to improve the performance, we also implement sparse gradients and MGD on both the server's side and clients' sides. To the best of our knowledge, this is the first well-performed FL framework that combines LDP and CDP to protect privacy via artificial noise and still has great performance. In this work, we accomplish the LDP-FL system by Gaussian Mechanism, while most existing works adopt the Laplace Mechanism, which is harder for implementation in reality. Our main contributions are as follows:

1. First, we apply the Gaussian Mechanism on the clients' side to achieve (ϵ, δ)-LDP so that clients' sensitive data can never be revealed. MA is used to keep track of privacy losses which control the ending of the FL.
2. Second, we also introduce CDP in [15] into our LDP-FL to further protect privacy and We employ two accountants for LDP and CDP separately.
3. Third, as the noise is scaled to the l_2-norm of the gradients, we apply sparse gradients to upload only part of gradients to the server, which can reduce noise and maintain the same privacy protection level. Meanwhile, communication costs are saved.
4. Fourth, we also implement MGD on the clients' side to speed up training and on the servers' side to stabilize the training process under the influence of the adding noise.
5. Finally, we conduct simulations with different settings, provide detailed privacy loss estimations and compare the results with other frameworks to show our proposed PPFL's effectiveness.

II. Preliminaries

In this section, we present definitions and formulas of DP, especially the Gaussian Mechanism for achieving DP, and DP-based FL.

A. Differential Privacy

The DP mechanism guarantees privacy protection for sharing databases. To be specific, for two adjacent databases D, D', which differ from each other at only one data point. Applying the DP scheme to perturb the original values can make the output of these two databases undistinguishable [19]. The DP mechanism is defined as follows:

Theorem 1: [19] A randomised mechanism M achieves ϵ-DP if it satisfies the following constraints, $\forall D, D', \forall S \subset R$:

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

where $\epsilon \in (0, 1)$.

However, this general definition is too strict to be fulfilled. Therefore, an approximate term δ is widely adopted, which means the probability of the general ϵ-DP is violated.

Theorem 2: [19] A randomised mechanism M achieves (ϵ, δ)-DP if it satisfies the following constraints, $\forall D, D', \forall S \subset R$:

$$Pr\left[M(D) \in S\right] \leq e^{\epsilon} Pr\left[M(D') \in S\right] + \delta, \tag{2}$$

where $\epsilon \in (0, 1)$ and $\delta \geq 0$. This mechanism is referred to as the Gaussian Mechanism and a common implementation is to add a zero-mean Gaussian noise to the original databases.

B. Differential Privacy based Federated Learning

The procedure to achieve CDP-FL is introduced as follows [15]. In addition to the initial global model generation and local model training via the SGD method, the server will compute the norm of updated models and transmit the latest gradients $\Delta \mathbf{w}_t^i$ as well as the norm $\|\Delta \mathbf{w}_t^i\|_2$ to the server. On the server's side, it will calculate the median value of all received norms S for a novel global model aggregation. To mitigate each client's influence, the gradients are clipped as follows:

$$\overline{\Delta w_t^i} = \Delta w_t^i / max(1, \frac{||\Delta w_t^i||_2}{S}), \tag{3}$$

where Δw_t^i is the gradients of the client i in communication round t. After that, the clipped gradients are aggregated and the noise, which is computed according to the noise scale and S, is added to the gradients. Finally, the aggregated gradients with noise are used for next round optimization.

III. Proposed Privacy-Preserving Federated Learning framework

In this section, we propose a PPFL framework that achieves LDP combined with CDP to enhance privacy protection. To reduce the total used communication overhead and improve the overall performance, only part of the gradients are sent to the central server. Besides, in our setting, MGD is used to train the local models to speed up training, as well as on the central server to help stabilizing the training process under the effect of artificial noise.

A. Local Differential Privacy and the combination of Differential Privacy techniques

In most existing FL frameworks, central servers are assumed to be honest, so to achieve CDP, artificial noise is only added after the aggregation in the central server. This prevents malicious clients from identifying whether a certain client joins the training process or not [15]. However, the central servers are not totally trustworthy in reality and may try to recover useful data from gradients. Therefore, an LDP mechanism of adding LDP noise on the clients' side should be implemented on top of CDP to further protect clients' data privacy.

In conventional FL, each client trains their model locally, computes the gradients and sends the gradients to the server without modification. As mentioned in [13], the central server can still recover useful information from the original gradients. To avoid such attacks, artificial noise is added to the original gradients and sent to the server. To enhance the privacy protection, in our proposed method, a Gaussian Mechanism is used on each client's side to achieve (ϵ, δ)-LDP.

After computing the gradients on each client locally, clients add Gaussian noise to them, which is scaled to the l_2-norm of the gradients. In our LDP settings, we refer the local data of each client to as a small subset of the entire data set involved in FL. To keep track of the accumulative privacy losses, MA [14] is used for a tighter bound of the privacy loss. Based on MA, the privacy losses are related to the proportion (q) of the batch in the whole dataset, ϵ, δ, the noise scale and communication rounds. In our case, as the noise is added to every client's gradients separately, $q = \frac{1}{N}$, where N is the total number of the clients. Besides, the total privacy losses in a single round can be defined as the sum of the privacy loss of all participants in this round. After applying LDP, the gradients of the client i sent to the server are

$$\Delta\overline{w_t^i} = w_{t-1} - w_t^i + \mathcal{N}(0, C_{LDP}^2 \sigma_{LDP}^2), \tag{4}$$

where w_{t-1} is the model in the previous round, w_t^i is the client i's model in this round, $\mathcal{N}$ denotes a zero-mean Gaussian distribution for the LDP noise, σ denotes the noise scale. Besides, C needs to be chosen properly so that it can protect privacy without seriously hindering the model's performance too much [14]. Therefore, to protect each single data point in the training stage, C is computed as:

$$C = \frac{||\Delta w_t^i||_2}{n} = \frac{||w_{t-1} - w_t^i||_2}{n}, \tag{5}$$

where n is the size of the involved data samples.

As mentioned in Section II-C, the gradients are usually clipped before aggregating and adding noise. However, in our LDP framework, the noise is added before the aggregation in the central server and clipping gradients are used to alleviate the influence of each participant. In LDP, there is no need to limit each sample's influence on the global model. Therefore, we omit the process of clipping in the proposed LDP.

After the central server receives all the clients' gradients, it assigns each client a weight for their contribution. In our framework, the weight is $\frac{n}{m}$, where m is the number of the data samples of every client involved in this round. The server then aggregates weighted gradients and then adds Gaussian noise to them to hide each client's contribution as introduced in [15]. Finally, the aggregated gradients with the combination of two differential privacy implementations are used to compute a new global model w_t in communication round t as follows:

$$\Delta w_t = \frac{1}{M} \sum_{i=1}^{M_t} \Delta\overline{w_t^i} + N(0, C_{CDP}^2 \sigma_{CDP}^2), \tag{6}$$

$$w_t = w_{t-1} - \Delta w_t, \tag{7}$$

where $\Delta\overline{w_t^i}$ is the gradient of the client i in communication round t with DP noise, M is the number of participants in every round, M_t is the set of participating clients in communication round t, C_{center} is the sensitivity of noise and clip bound in CDP, and σ_{center} is the scale for CDP noise.

To calculate the privacy losses of the proposed combination of two DPs, in this paper, we use two separate accountants for LDP and CDP. In this way, whichever accountant runs out of budget, the FL stops.

B. Gradients sparsification

Although the noise scale is well-chosen to preserve useful information for the model, it can still degrade the overall performance. The added Gaussian noise has a mean of zero so that when the number of clients increases, the noise can be offset to a certain degree after aggregation. However, the performance is still degraded when increasing the number of clients and, sometimes, it is impossible to have a massive number of clients joining the FL. To further improve the performance, as the noise is scaled to the l_2-norm of the gradients, by only sending part of gradients, the l_2-norm value is smaller. This means that the artificial noise would have less effect on the gradients while still preserving the data privacy on the same level. Besides, as the FL is usually performed on smart devices which have limited power and communication resources, sparsifying the gradients can save a large proportion of communication costs.

Furthermore, we also consider how to sparsify the gradients. To save as many communication costs as possible, while reducing the effect of DP on the performance, the sparse percentage should be as large as possible. On the other hand, sending a small part of the original gradients slows down the training process, resulting in degraded performance. In ML, the magnitude of the gradients stands for each point's influence on the final losses. Therefore, in our scheme, we only maintain a percentage of gradients with the largest absolute values, which are mathematically more significant than the smaller ones. Then Gaussian noise is computed according to the sparse gradients' l_2-norm and added to them. In this way, the proposed framework can improve the models' performance and also save communications costs.

C. Applying MGD on central server and clients

The accuracy performance can be very unstable due to the noise, which may slow down the training process or result in poor performance. To speed up and help stabilizing training steps, we apply the MGD on the central server after aggregation, referred to as Global-MGD. In a traditional single-end ML, MGD is applied between batches of data to accelerate training. In our case, we not only use MGD during local training to speed up gradient descent, but also apply it to the central aggregation. To be specific, every communication round of FL can be seen as a batch of the combination of selected users' data. After the first communication round, the previous round's aggregated gradients ($V_{dw_{t-1}}$) are used as the next one's momentum. By applying the MGD formulation, every round's gradient is calculated as a combination of the recursion of previous rounds and the current one, which is:

$$V_{dw_t} = \beta * V_{dw_{t-1}} + (1-\beta) \cdot \Delta w_t \quad | \, V_{dw_0} = 0, \tag{8}$$

$$w_t = w_{t-1} - \lambda * V_{dw_t}, \tag{9}$$

where Δw_t is calculated in Eqt (6).

Algorithm 1 outlines our FL framework with the combination of LDP and CDP, referred to as LCDP-FL, and implementation of Sparse Gradients and Global-MGD. At the beginning of the algorithm, we first initialize a global model and two privacy accountants for LDP and CDP, respectively. For communication round t, accountants check whether the privacy losses exceed the budgets. If not, the server chooses a set, M_t, of clients and sends them the current global model. Then, each client in M_t performs local training, computes and sparsifies the gradients, adds LDP noise on the gradients and sends the gradients with noise and their norm values to the server. Next, the server clips and aggregates all the received gradients. Finally, the server computes a new global model through Global-MGD and adds CDP noise to the new model.

IV. Simulation results and discussion

In this section, we conduct a series of simulations to show our proposed framework's performance on the MNIST dataset [2], which is a hand-written digit image dataset and consists of 60,000 training images and 12,000 test images. The proposed PPFL is compared with some well-known algorithms, namely DP-SGD [14], LDP-FL [16] and CDP-FL [15] in terms of accuracy and communication costs in Table 1, where CR is the number of total communication rounds, TC is the number of total clients, CSR is the fraction of users to be selected per round and Acc is the accuracy performance. In this table, all our proposed schemes are implemented with Global-MGD ($\beta = 0.5$) and update only 10% of the gradients with the largest absolute values. Besides, we also compare our framework to some classical schemes, called vanilla-FL, which is Fed-Avg using MGD in local training [7], [11]. The training dataset is divided into shards, where each shard contains data with the same label and each client is assigned with two shards. We run our proposed framework with 100 and 1,000 FL clients. Our LDP privacy budget is set as $\epsilon = 0.5$ and $\delta = 1\mathrm{e}{-6}$ and CDP privacy budget is set to the same as in [15], where $\epsilon = 0.5$, $\delta = 1\mathrm{e}{-3}$ for 100 clients and $\delta = 1\mathrm{e}{-5}$ for 1,000 clients. For each client, the local training model is a multi-layer perceptron which consists of two hidden layers with 200 units per layer and a Relu activation and MGD are employed. Each local model performs ten epochs of MGD per communication round. Our LCDP-FL stops when the privacy budget runs out.

Algorithm 1 LCDP-FL with sparse gradients and MGD

```
procedure CENTRAL SERVER
    Initialize a global model $w_0$ and privacy accountant for server $PA_{server}$ and clients $PA_{clients}$, $V_{\Delta w_0} = 0$
    for communication round $t = 0, 1, 2...$ do
        $\delta_{server} \leftarrow PA_{server}(M, \sigma_{server})$
        $\delta_{clients} \leftarrow PA_{clients}(M, \sigma_{clients})$
        if $\delta_{server} > PB_{server}$ or $\delta_{clients} > PB_{clients}$ then return current model
        end if
        choose a random set of $M$ clients as $M_t$
        for Client $i$ in $M_t$ do
            $\Delta \ w^i_{t+1}$, $||w^i_{t+1}||_2 \leftarrow$ ClientDP($i$,$w_t$)
        end for
        $C_{server} = median\{||w^i_{t+1}||_2\}_{i \in M_t}$
        $\Delta w_{t+1} = \frac{1}{M} \sum_{i=1}^{M_t} \Delta w^i_t$
        $V_{\Delta w_{t+1}} = \beta * V_{\Delta w_t} + (1-\beta) * \Delta w_{t+1}$
        $w_{t+1} = w_t - V_{\Delta w_{t+1}} + N(0, C^2_{center} \sigma^2_{center})$
    end for
end procedure
procedure CLIENTDP($i$,$w_t$)
    $w^i_t \leftarrow E$ epochs of MGD
    $\Delta w^i_{t+1} = w_t - w^i_t$
    $\Delta w^i_{t+1} \leftarrow$ the largest top-rate% absolute values of the $\Delta w^i_{t+1}$
    $\Delta w^i_{t+1} += N(0, C^2_{client_i} \sigma^2_{client_i})$
    return $\Delta w^i_t$,$||w^i_{t+1}||_2$
end procedure
```

A. Proposed PPFL with LDP only

In this part, we first investigate our LDP mechanism's performance and the impact of the sparse gradients and MDG at the server's side on accuracy performance. For LDP-only,

[2] http://yann.lecun.com/exdb/mnist/

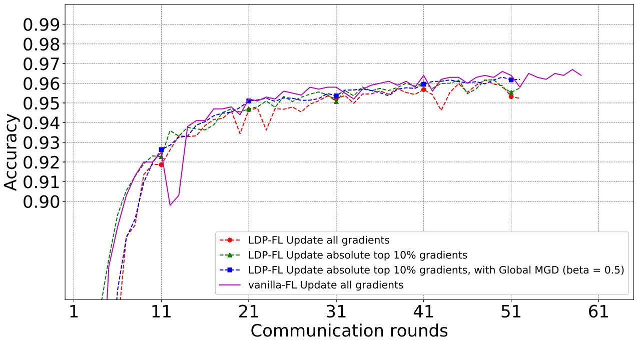

Figure 1. Accuracy performance of Non-PPFL (vanilla-FL) and LDP-FL for 100 clients.

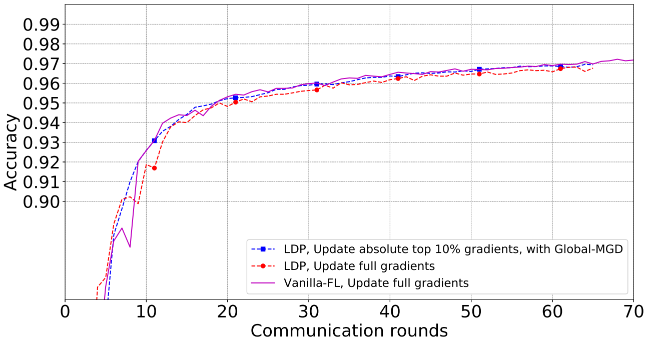

Figure 2. Accuracy performance of Non-PPFL (vanilla-FL) and LDP-FL for 1,000 clients.

the learning rate for the local model optimizer begins at 0.1, decays by 0.96 for the first 20 rounds and is fixed at 0.044 after 20 rounds. The noise scale is $\sigma_{100} = 8$ for 100 clients and $\sigma_{1000} = 2$ for 1,000 clients. Fig. 1 shows the results for the vanilla and our proposed FL framework with 100 clients. We show that although our PPFL has degraded accuracy performance when compared with the vanilla one, the sparse gradients and central-MGD can alleviate the performance degradation caused by the adding noise. To be specific, the final accuracy of the FL with these techniques outperforms the others and reaches a value of about 96.31%. As shown in Table 1, our proposed LDP-FL framework has a higher accuracy performance than the one in [16], while our proposed framework has a more general and easily achieved privacy settings by adding δ in DP. For 1,000 clients, our proposed framework outperforms the original PPFL (without Sparse Gradients and Central-MGD) and obtains almost the same accuracy of 97%, compared with non-PPFL, as shown in Fig. 2.

B. Proposed PPFL with CDP and LDP combined

In this section, we investigate the performance of our proposed framework combined with CDP presented in [15]. In our LCDP-FL, $\epsilon_{LDP} = 8$. As the noise is scaled to the norm of the gradients, the learning rate needs to be chosen properly. When training with 100 clients, the initial learning rate is set as 0.0025, decays by 0.78 for the first 10 rounds and is fixed at 0.000267 after ten rounds[3]. In addition, when training with 1,000 clients, the initial learning rate is set as 0.1, decays by 0.78 for the first 10 rounds and is fixed at 0.0107. The noise scale for LDP is set as $\sigma = 2$, while for LCDP-FL, it is set as $\sigma_{100} = 1.18$ for 100 clients and $\sigma_{1000} = 1.43$ for 1,000 clients on the purpose of comparison with the state-of-the-art CDP-FL [15]. For the PPFL with LCDP with 100 clients, Fig. 3 shows that our proposed method achieves the highest accuracy performance, namely 80.24%, among all the LCDP-FL. At the same time, it can reduce 90% of the total communication costs. Although it has much worse accuracy than the non-PPFL one, it outperforms the CDP-FL [15] with 100 clients, as presented in Table 1. Besides, we also investigate the performance of only updating 1% of the gradients, which has the worst performance, as only very little useful information is updated for the center server each round.

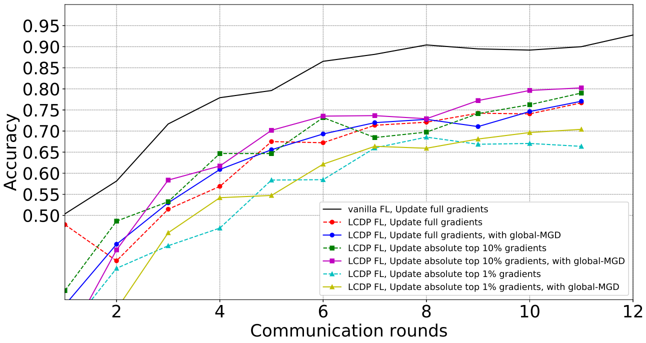

Figure 3. Accuracy performance of LCDP-FL on 100 clients.

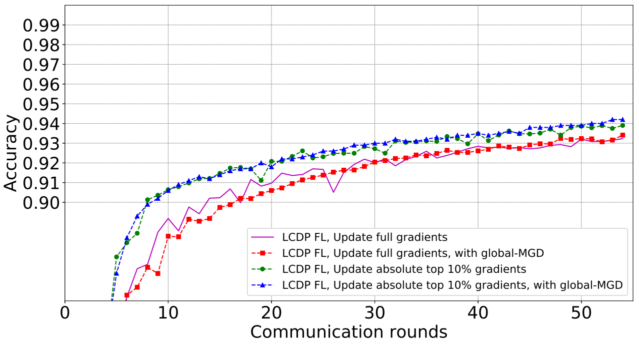

Figure 4. Accuracy performance of LCDP-FL on 1000 clients.

Furthermore, Fig. 4 shows that our proposed framework with 1,000 clients also reaches the highest accuracy performance among all the LCDP-FL, namely 94.2%, which also outperforms the one for the CDP-only in [15], as shown in Table 1. However, when adopting CDP, our PPFL has slightly worse accuracy than non-PPFL, caused by the small learning rate value.

[3] Our experiments show that for learning rates larger than 2.5e−3, the DP noise destroys the training while having a smaller one will make the training process too slow.

Table 1 This table shows performance comparisons between My PPFL and other well-known privacy-preserving ML

Algorithm	CR	TC	CSR	Acc	DP
DP-SGD [14]	700	1	1	97%	(8,1e-5)-DP
Our LDP-FL	52	100	0.5	96.3%	(0.5,1e-6)-DP
Our LDP-FL	63	1000	0.22	**97%**	(0.5,1e-6)-DP
LDP-FL [16]	10	100	1	95.36%	(0.5)-DP
CDP-FL [15]	11	100	0.5	78%	(8,1e-3)-DP
CDP-FL [15]	54	1000	0.22	92%	(8,1e-5)-DP
Our LCDP-FL	11	100	0.5	**80.24%**	(8,1e-3)-CDP & (8,1.07e-107)-LDP
Our LCDP-FL	54	1000	0.22	**94.3%**	(8,1e-5)-CDP & (8,4e-111)-LDP

C. Discussion of privacy losses

In this section, the privacy losses of the LCDP-FL is discussed. As mentioned in Section III, we use two accountants for LDP and CDP separately. However, according to the MA calculation in [14] and that CDP has a much larger q than LDP, CDP has relatively higher losses than the one of LDP. Therefore, the FL always stops when the CDP accountant exceeds the privacy budget. Meanwhile, we calculate the privacy losses for LDP. Through MA, the difference between the privacy losses of LDP and CDP is very large, as shown in Table 1. Even though we desire as small privacy losses for each client as possible, the FL controlled only by CDP's accountant can train for very few rounds, resulting in poor performance. In our case, the privacy protection for LDP is more important, and, with LDP, CDP can be achieved to a certain degree. Thus, the CDP privacy budget can be loosened in further research to achieve better performance.

V. Conclusion

In this paper, we propose a new DP-based framework for PPFL by adding sparse gradients and Global-MGD, in order to improve its convergence performance. To be specific, we propose an LDP framework based on Gaussian Mechanism and MA (to track privacy losses), and also implement CDP [15] to enhance privacy protection. We also propose a new scheme to calculate DP noise scale for our LDP-FL. Experimental results show that for MNIST, our proposed FL framework achieves better performance than other DP-based FL. Besides, our framework can also reduce massive communication costs using sparse gradients. We also provide mathematical results of privacy losses of our framework with all the settings to show the privacy protection level.

It has been shown that under our privacy losses calculation scheme, the privacy losses for LDP and CDP have a huge difference so that the FL training stops when CDP runs out of budget. Therefore, in future work, a new scheme should be investigated to balance the privacy losses for LDP and CDP in order to let CDP and LDP cooperatively decide when to stop FL training. In addition, as our framework still suffers from Non-IID datasets, a new regularization technique should be implemented to adjust DP-based FL for better accuracy performance.

References

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[2] 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.

[3] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated Learning for Healthcare Informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021. [Online]. Available: https://doi.org/10.1007/s41666-020-00082-4

[4] M. Sheller, B. Edwards, G. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. Colen, and S. Bakas, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific Reports*, vol. 10, 2020.

[5] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Federated Learning for Ultra-Reliable Low-Latency V2V Communications," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–7.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in iot," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5986–5994, 2019.

[11] W. Liu, L. Chen, Y. Chen, and W. Zhang, "Accelerating federated learning via momentum gradient descent," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 8, pp. 1754–1766, 2020.

[12] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[13] 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.

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

[16] L. Sun, J. Qian, X. Chen, and P. S. Yu, "LDP-FL: Practical private aggregation in federated learning with local differential privacy," *arXiv preprint arXiv:2007.15789*, 2020.

[17] 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.

[18] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. 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.

[19] 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.