Privacy-Preserving Federated Learning based on Differential Privacy

Shangyin Weng

Submitted in fulfilment of the requirements for the Degree of Doctor of Philosophy

School of Engineering
College of Science and Engineering
University of Glasgow

October 2024

Abstract

Although FL is claimed to guarantee privacy protection, semi-honest servers and local clients can still reconstruct sensitive information from the gradients. Therefore, to enhance privacy protection, differential privacy (DP) is widely adopted in FL by randomizing the gradients before transmitting them to other parties. Nevertheless, randomizing gradients inevitably degrades FL performance in terms of lower accuracy and higher communication overhead. To solve this problem, this thesis focuses on exploring methods to enhance privacy protection and improve the overall utility of DP-based FL (DPFL) frameworks.

This thesis begins with the research question on improving the degraded accuracy performance and reducing communication overhead for centralized DPFL while maintaining a strong privacy protection guarantee. Two different frameworks are proposed to tackle this question. The first framework combines local DP (LDP) and central DP (CDP) to prevent both central servers and clients from recovering private information by adding noise to the local gradients before uploading and to the aggregated gradients on the server side before broadcasting, respectively. To improve the overall utility of the proposed DPFL, a novel sparse mechanism is adopted on the local gradients before adding noise and a global momentum gradient descent is introduced on the server side and the client side. For the second framework, a novel LDP-based FL framework with two performance improvement modifications is proposed. One modification is to calculate the difference between noisy and original gradients, and add the difference to the objective function to be minimized. The other modification is to calculate the expectation of the loss created by noise, which is also incorporated into the objective function to be optimized. For both modifications, the privacy protection levels are the same as those for plain DPFL since no modifications to DP settings have been made. This thesis presents the necessary convergence analysis for the proposed framework under convex and non-convex settings. A series of simulations is conducted to validate both frameworks' effectiveness in terms of higher accuracy and lower communication costs. Specifically, the first framework can outperform other DPFL frameworks while saving 90% of communication costs since sparse mechanism can improve the performance under DP noise. The second framework can save up to 40% of communication and training rounds while achieving better accuracy than plain LDP-based FL.

The second research topic in this thesis is to investigate the impact of DP on privacy protection across various DP noise and clipping settings. To address this, an evaluation method for privacy leakage in the FL is proposed by utilizing reconstruction attacks to analyze the difference between the original images and reconstructed ones. Furthermore, this thesis studies the accumulative privacy loss under two different reconstruction attack settings and demonstrates that anonymizing local clients can decrease the probability of privacy leakage. Next, the effects of different clipping methods on privacy protection are analyzed. Simulations are conducted to characterize the trade-off between privacy protection and learning accuracy and demonstrate that there is an optimal DP setting to provide the desired privacy guarantee. The summarized theoretical findings and simulation results in this work can be utilized to guide heterogeneous DP settings for DPFL.

The third research topic of this thesis explores privacy enhancement and accuracy improvement in decentralized DPFL. To address these challenges, a novel anonymous decentralized DPFL framework is proposed. Specifically, two decentralized DPFL methods based on the gossip and fake-centralized manners are first introduced, where the training clients selection rate (TCSR) in each round for both methods and the model exchange rate (MER) in the gossip method are researched. To enhance privacy protection, an anonymous mechanism, is proposed where all clients are unaware of whom they are communicating with and cannot determine whether they are communicating with the same client across several rounds. Next, the required noise scale is derived in terms of the DP settings, TCSR and MER. Subsequently, the convergence bound for the proposed framework is provided, which suggests that an optimal number of clients for is needed to achieve the best convergence performance. Finally, a series of simulations is conducted to evaluate the performance. The simulation results show that the proposed framework only has a small degradation in accuracy compared to the non-private FL and validate our theoretical results.

In conclusion, this thesis provides insight into increasing overall utility and enhancing privacy protection in DPFL. The convergence and privacy analysis of the proposed frameworks provides a basis for future research focusing on further improving the performance of DPFL. Moreover, the proposed privacy leakage evaluation method can provide a more intuitive understanding of privacy loss, which can be utilized to improve accuracy and promote privacy audits for regulatory compliance and user assurance.

University of Glasgow

College of Science & Engineering

Statement of Originality

Name: Shangyin Weng

Registration Number: xxxxxxxx

I certify that the thesis presented here for examination for a PhD degree of the University of Glasgow is solely my own work other than where I have clearly indicated that it is the work of others (in which case the extent of any work carried out jointly by me and any other person is clearly identified in it) and that the thesis has not been edited by a third party beyond what is permitted by the University's PGR Code of Practice.

The copyright of this thesis rests with the author. No quotation from it is permitted without full acknowledgement.

I declare that the thesis does not include work forming part of a thesis presented successfully for another degree.

I declare that this thesis has been produced in accordance with the University of Glasgow's Code of Good Practice in Research.

I acknowledge that if any issues are raised regarding good research practice based on review of the thesis, the examination may be postponed pending the outcome of any investigation of the issues.

Signature: ....Shangyin Weng.......

Date: ....2024.10.01.......

Contents

List of Tables

List of Figures

List of Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
CDP	Central Differential Privacy
CNN	Convolutional Neural Network
DC	Distributed Client
DFL	Decentralized Federated Learning
DL	Deep Learning
DLG	Deep Leakage from Gradients
DML	Distributed Machine Learning
DP	Differential Privacy
DPDFL	Differential Privacy-based Decentralized Federated Learning
DPFL	Differential Privacy-based Federated Learning
FL	Federated Learning
FNN	Feedforward Neural Network
GAN	Generative Adversarial Network
GD	Gradient Descent
GDPR	General Data Protection Regulation
HE	Homomorphic Encryption
HFL	Horizontal Federated Learning
IoT	Internet of Things
LDP	Local Differential Privacy
MA	Moment Accountant
MER	Model Exchange Rate
MGD	Momentum Gradient Descent
ML	Machine Learning
MLP	Multi-Layer Perceptron
Non-IID	Non-Independent and Identically Distributed

PFA	Projected Federated Averaging
PPFL	Privacy-Preserving Federated Learning
PSNR	Peak Signal-to-Noise Ratio
UDP	User-level Differential Privacy
RDP	Rényi Differential Privacy
RNN	Recurrent Neural Network
ReLU	Rectified Linear Unit
SE	Single-End
SGD	Mini-batch Stochastic Gradient Descent
SMC	Secure Multi-party Computation
SOTA	State of The Art
SPOF	Single Point of Failure
TCSR	Training Client Selection Rate
TPS	Trusted Proxy Server
TV	Total Variation

List of Important Symbol Notations

α_1	The scale factor of the total variation
α_2	The weighting coefficient in the privacy measuring metric
β_1, β_2, st	The factors in the Adam optimizer
$\varepsilon, \delta, \alpha$	The privacy budget parameters for DP
γ	The learning rate
$\lambda_{nore1}, \lambda_{nore2}$	The factors used to control the Fed-nore
$\mathbb{B}, \mathbb{S}$	The masks in the SMC
$\mathbb{L}$	The loss function
$\mathbb{N}, \mathbb{T}$	The number of parts in the SMC
$(a_{k-1})^T, (w_k)^T$	The transpose operation of a_{k-1}, w_k, respectively
$\nabla F(w_i^t)$	The derivative of the objective function of the i-th client in t-th round
∇W_i^t	The gradients of the i-th client in t-th round
$\nabla \overline{W_i^t}$	The clipped and noisy gradients of the i-th client in t-th round
ρ_{cs}, ρ_{ed}	The bias factors in the privacy mesauring metric
ρ_2	The second largest eigenvalue of a graph matrix
σ	The base noise variance of the DP mechanism
cpc	The number of classes of data per client
C	The clipping bound in the DP mechanism
C_i	The clipping bound of the i-th client in UDP mechanism
CS	The average cosine similarity between the dummy images and true images
CS_i	The cosine similarity between the i-th dummy image and true image
$d\bar{z}, d\overline{w}, d\overline{b}$	The expected changes on the gradients in Fed-nore-2
dw, db, dz, da	The partial derivatives in ML
dz^{conv}, dz^{pool}	The expected changes on the gradients of the convolution layer and pooling layer

D, D'	Two neighboring datasets
D_i	The dataset of the i-th client
ED	The average Euclidean distance between the dummy images and true images
ED_i	The Euclidean distance between the i-th dummy image and true image
Δf	The l_2-sensitivity of a function f
$F(W)$	The global objective function
$F_i(W_i)$	The local objective function
$g(x)$	The generated dummy gradients at the attacker
$g(\hat{x^*})$	The clipped gradients
$\hat{g_i^t}$	The clipped gradients of the i-th client in t-th round
$\overline{g(x^*)}$	The received noisy gradients at the attacker
$\overline{g_i^t}$	The noisy gradients of the i-th client in t-th round
$g(x^*)$	The received original gradients at the attacker
g_i^t	The gradients of the i-th client in t-th round
$g_{act_k}(z), g'_{act_k}(z)$	The activation function of the k-th layer and its derivative
$J, \nabla J$	The objective function of the reconstruction attack and its derivative
J_{nore_i}	The distance between noisy gradients and original ones in Fed-nore-1
K	The final layer of the model
L, ζ, ρ	The parameters for convergence analysis
M	The number of the clients in total
m_{tcs}, m	The number of the training clients in each round
m_e	The number of the models exchanging clients
n	The number of data of each client
n'	The number of the input data of training
n_i	The number of data of the i-th client
$nvar$	The noise variance
$N(\mu, \sigma^2)$	A Gaussian distribution with a mean of μ and a variance σ^2
$N(dw), N(db)$	The noisy gradients of hidden layer and bias
N_c	The number of total classes
N_i^t	The added noise on the i-th client in t-th round
o	The index of the label in the one-hot output
p	The index of the true label in the one-hot output
p_i	The averaging weight of i-th client
pl_j	The probability of privacy leakage of the image j

$P(pl)$	The practical privacy leakage probability
PA	The privacy accountant
r_{tcs}, r_{mer}	The clients selecting rates of training and model exchange
R	A randomized mechanism
s	The probability of sign flipping in the noisy reconstruction of signed Adam
S'	The set of the possible output
S^t	The set of the training clients at t-th round
S_i^t	The set of the clients' models received at the i-th client in t-th round
S_i	The sensitivity of the i-th client
t	The t-th communication round
T	The number of total communication rounds
TV	The total variation function
v_1^t, v_2^t	The first order and second order momentum of the gradients in Adam optimizer
V_{dWt}	The first order global momentum
w_k, b_k	The parameters of the k-th hidden layer and bias
W^t	The global model in the t-th round
$\overline{W^t}$	The noisy global models in the t-th round
W^*	The optimal global model
W_i^t	The model of the i-th client in the t-th round
$x*$	The true input images
z_k, a_k	The k-th layer's intermediate optimization parameters

Acknowledgements

First and foremost, I would like to express my most sincere gratitude to my supervisor, Prof. Lei Zhang, for his exceptional guidance and support during my research. His careful guidance contributed to the rapid growth of my research ability and quality, which greatly promoted my academic and professional development. I would also like to thank him for his great help and suggestions in my daily life in the UK. It is a great honor to have had the opportunity to research under his supervision.

I am also deeply thankful to Dr. Ruiyu Wang, Dr. Xiaoshuai Zhang, Dr. Paulo Valente Klaine, and Prof. Muhammad Ali Imran for their encouragement and suggestion in my research area. Their guidance has greatly enriched my research experience.

I would also like to thank all my friends for their support and companionship during this journey.

To my girlfriend, I appreciate her care, support, and constant presence since the start of this journey. Her love and understanding have been a consistent source of comfort and strength throughout this experience.

Finally, I would like to express my appreciation to my dear family, especially my parents, for their enduring belief in me. I am grateful for their care and support, allowing me to pursue this journey without any concerns. Their encouragement has been the foundation of this work.

Chapter 1

Introduction

With the rapid development of Internet of Things (IoT), all kinds of smart devices, including mobile phones, autonomous vehicles, IoT sensors, and others, have been deployed [1, 2]. Recent estimates suggest that the current deployment of IoT devices has surpassed 20 billion worldwide [3]. This number is expected to see a significant rise to approximately 125 billion by the end of this decade [3]. Concurrently, these devices are generating an unprecedented volume of data. According to recent research, the total data output from IoT devices is expected to exceed one yottabyte in the near future [4]. By utilizing this tremendous amount of IoT data, artificial intelligence (AI), including machine learning (ML) and deep learning (DL) algorithms, has been rapidly developed in many fields, such as biomedical, natural language processing, recommendation systems, auto-driving, and so on [5–8]. Traditional learning algorithms are normally executed on a central server, requiring extensive data collection. Although these algorithms play a crucial role in enhancing our lives, the centralization of sensitive data has raised significant privacy concerns.

To be specific, there are two general types of data privacy-related problems. First of all, application service providers would like to collect users' data to train their algorithms for the purpose of enhancing the overall experience for all users. Local users may consider that this data-collecting action violates their privacy, and they refuse to share their data, which makes the ML training more problematic [9]. Secondly, the issue of data silos is becoming increasingly serious. Different organizations and even distinct departments within the same organization usually refuse to exchange their own data due to the lack of effective privacy-preserving data-sharing methods [9]. This isolation can lead to the delay of strategic capabilities and the decision-making process of the organizations. For example, two hospitals may refuse to share patient histories and laboratory test records, which can delay the patients' treatment, leading to fatal consequences. However, exchanging plain data (unprotected data) is a privacy violation for their customers, making plain data exchange impossible for different organizations. In addition to the direct disagreements from the users and organizations, the General Data Protection Regulation (GDPR), California Consumer Privacy Act and many other privacy laws have been enacted to force big companies to legally collect and utilize users' data only when the users provide clear consent [10, 11].

With the difficulty of collecting adequate data or synthesizing a large amount of high-quality data, it becomes impractical for these data-driven learning algorithms to meet acceptable service performance requirements. Therefore, to overcome the privacy challenges in training AI models, distributed ML (DML) is introduced [12]. DML allocates the learning tasks to edge devices. The edge devices ask the server for a pre-trained or new model and use their own data to train for personalization. Once the training is finished, they can use the model to infer output for their personal needs. However, the lack of model and data exchange in DML between local clients and the central server prevents both parties from using external data to train the model. As a result, the models are less generalized and may perform poorly on heterogeneous applications and data.

To solve the aforementioned challenges, federated learning (FL) is proposed [13, 14]. A diagram of FL is illustrated in Fig. 1.1. FL is introduced by Google and implemented in a smart keyboard application for typing recommendations [13]. Unlike traditional learning, FL allows users to train local models with their personal data for an epoch and upload their trained model gradients instead of their raw data. The server then averages all the received gradients to generate a new global model, which is broadcast to the users for the next round of training. This framework is also known as FedSGD. To reduce the communication overhead of exchanging gradients and accelerate convergence in FedSGD, FedAvg is proposed to perform multiple epochs of local training before uploading the gradients [14].

Ever since its introduction, FL has been widely adopted in many applications [15–18]. For example, the authors in [15] adopt FL to learn from electronic health records from different hospitals to collaboratively train a model. Besides, FL is also used in tumor classification and other medical areas where privacy protection is very important for patients [16]. In addition, FL is also adopted to let users train an autonomous driving model based on their own driving pattern locally and then aggregate the local models to be a new model [17]. Except for the aforementioned applications, FL is also used for industrial IoT data sharing, smart home improvement and other areas [18, 19].

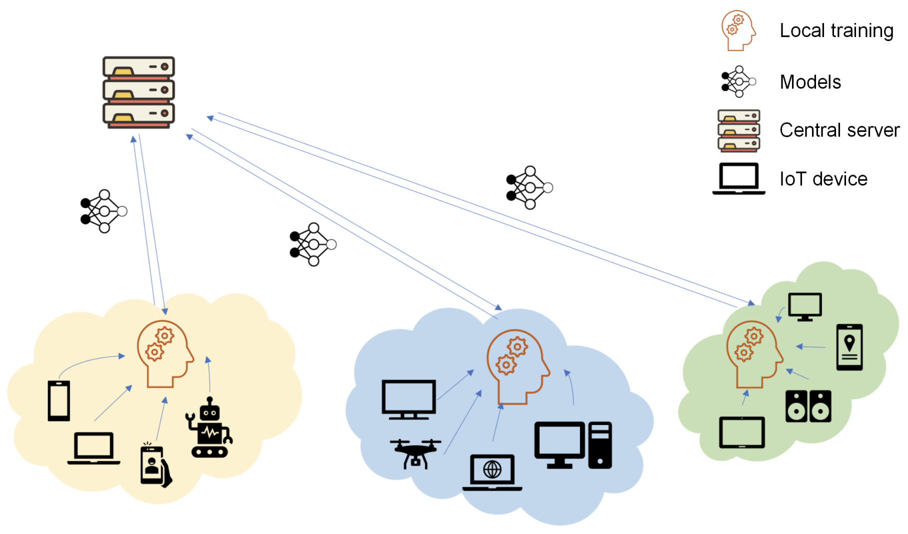


Figure 1.1: An illustration of traditional FL.

Compared to traditional ML, FL has the following advantages:

- **Privacy protection:** During FL, the local clients only upload their model parameters, and there is no need to collect users' data and store it in the central server so that their raw data will never leave local storage [13]. Therefore, data privacy is preserved. Meanwhile, FL enables different organizations to collaboratively train models without leaking their data to others [20].

- **Low latency and bandwidth requirement:** When using traditional ML, all the model inferences (outputs) are conducted in the central server. Therefore, FL can reduce the latency of transmitting data to the server to obtain outputs from the ML models [20, 21]. Meanwhile, transmitting all the IoT data to the server increases the bandwidth requirements. By storing them locally, FL can reduce the overall communication costs [20].

- **Scalability:** FL can be performed on numerous smart devices, which intuitively increases the scalability of FL [22].

- **Reduced computation burden:** By distributing the job of training to numerous smart devices, the computational burden of the central server can be reduced without bringing too much computation burden for the local user. In Google's proposal, FL training can be scheduled during the idle evening time of most IoT devices, which prevents interference with users' customary usage [14].

1.1 Categories of Federated Learning

Reviewing recent FL frameworks, this thesis categorizes FL based on two aspects: data distribution and topologies.

1.1.1 Data Distribution

The data distributions used in FL can be categorized into three types based on their sample and feature spaces [9], as illustrated in Fig. 1.2.

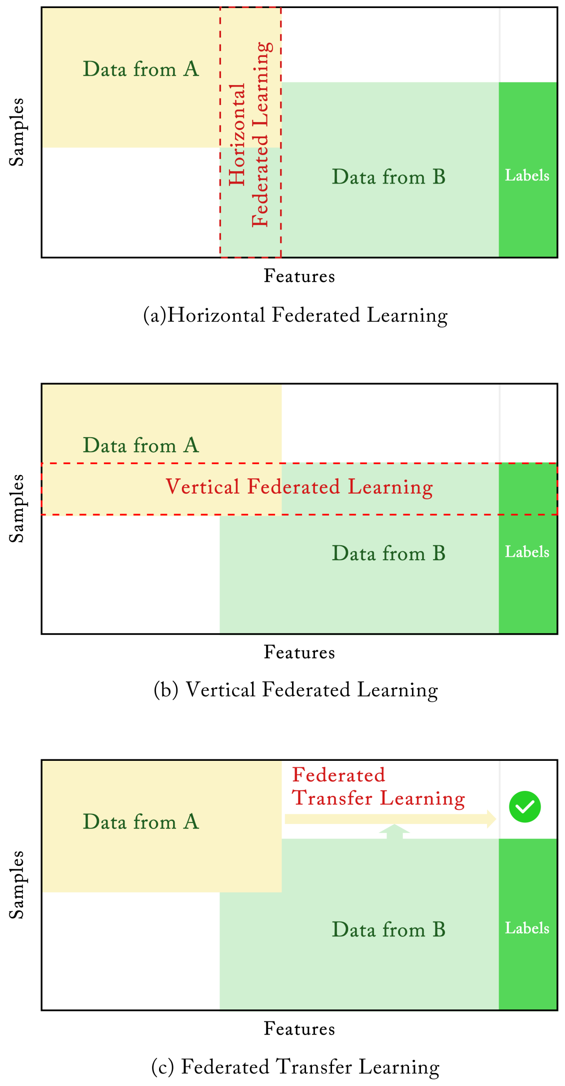

Samples
Data from A
Horizontal Federated Learning
Data from B
Labels
Features
(a)Horizontal Federated Learning
Samples
Data from A
Vertical Federated Learning
Data from B
Labels
Features
(b) Vertical Federated Learning
Samples
Data from A
Federated Transfer Learning
Data from B
Labels
Features
(c) Federated Transfer Learning

Figure 1.2: Different data distributions of FL based on sample and feature spaces [9].

1.1.1.1 Horizontal Federated Learning

In horizontal FL (HFL), all the local users jointly train the global model using data with similar feature spaces but from different sample spaces [9, 23]. For example, the data used in HFL can be driving patterns from different groups of users. With HFL, the amount of training data on the same task-learning algorithm is increased, and the model becomes more generalized, bringing better performance to all the participants.

1.1.1.2 Vertical Federated Learning

In vertical FL, all the local users have similar sample spaces while having different feature spaces [24, 25]. For example, an e-commerce company can utilize their clients' banking status and browsing histories to deliver more tailored recommendations for products. By applying vertical FL, the feature spaces of the dataset expand.

1.1.1.3 Federated Transfer Learning

Federated transfer learning is able to transfer cross-domain knowledge among users who have data with only a few overlapping features and sample spaces [26, 27]. For example, hospital A, with a larger dataset, has a pre-trained model, and another hospital B, with a much smaller dataset, would like to contribute to the model. The hospital B can use transfer learning [28] to utilize the model of A to supervise its local training and transmit B's gradients back to hospital A. Hospital A can aggregate the gradients of B with its model to generate a new model. In this way, both hospitals can have a model benefiting from each other without sharing their own data.

1.1.2 Federated Learning Topologies

There are three types of general FL topologies [9], which are categorized as described below.

1.1.2.1 Centralized Federated Learning

Centralized FL, which is introduced in the previous section as FedAvg, requires a server to collect and aggregate local gradients to form a new global model, which is then broadcast [14].

1.1.2.2 Decentralized Federated Learning

Decentralized FL (DFL) assigns the aggregation tasks to local clients, eliminating the need for a central server [29]. For example, some DFL frameworks choose a temporary leader from local clients every round to perform aggregation and broadcast the global model [30].

1.1.2.3 Hierarchical Federated Learning

Hierarchical FL can work in both centralized and decentralized ways, which applies some intermediate parties to first collect nearby clients' gradients to aggregate into a new model [31, 32]. In the next step, the intermediate parties send their aggregated models to the central server in centralized FL or to the temporary leader in DFL for the final aggregation and broadcasting. Hierarchical FL can reduce the overall bandwidth burden of transmitting gradients compared to the other topologies.

1.2 Challenges and Solutions in Federated Learning

Although FL sounds promising, it still faces many challenges, including statistical heterogeneity, high communication costs, privacy leakage and other problems [33, 34]. In this section, some challenges of FL and their corresponding solutions are introduced. Since this thesis focuses on privacy-preserving FL, a more detailed introduction on the privacy issues in FL is presented.

1.2.1 Statistical Heterogeneity

FL requires local users to train a local model using their personal data. However, local users may have different distributions of data [36,37]. Formally, this type of data is defined as non-independent and identically distributed (non-IID) data. As shown in Fig. 1.3, different users can have various sizes and classes of data in a ten-class classification task.

When FedAvg is performed on non-IID data, local users have different data distributions so that their trained model may have different optimization directions. Therefore, the clients may lead the global model to diverge and degrade the convergence performance. In this thesis, the degree of non-IID is measured by different classes of data per client, referred to as *cpc*. The smaller the *cpc* is for a fixed number of total classes, the more heterogeneously the data is distributed. Therefore, the local models may have very different optimizing directions, leading the global model to diverge. For example, the performance of FedAvg with ten clients and different values of *cpc* from total classes of ten is shown

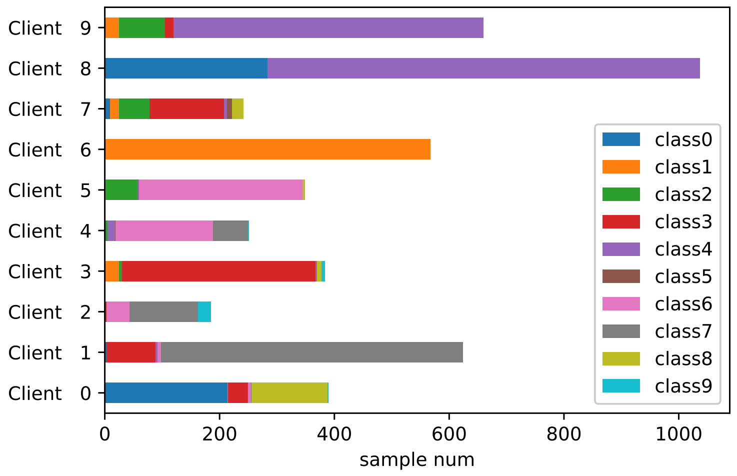


Figure 1.3: A possible non-IID data distribution example [35].

in Fig. 1.4, which shows that as the *cpc* decreases, accuracy also decreases. In some real-world applications, the degree of non-IID is significant. For example, in some video recommendation systems, some local users may only prefer one type of video, while there may be more than twenty types of videos. Therefore, improving the performance of FL under non-IID data needs to be addressed.

To tackle this problem, the recent works focus on designing different optimization strategies [39, 40]. For example, the authors in [39] propose adding the Euclidean distance between the global model and local model in the local optimizer to minimize their difference.

1.2.2 Heavy Communication Costs

FL does not require local users to upload their data, but it needs frequent model exchange to train a global model collaboratively. Even though the size of the transmitted model parameters may be smaller than that of raw data, the frequent model exchange brings huge communication overhead. Therefore, reducing the cost of transmitting gradients is a very important topic for FL.

To reduce the communication size of each round, there are two general approaches [41]. The first approach is to directly reduce the size of transmitted gradients in each round through compression or quantization after training [41]. The second approach is to restrict the updates to prefixed structures and train directly on those prefixed updates using low rank [41]. The low rank method is to express the update to the product of two

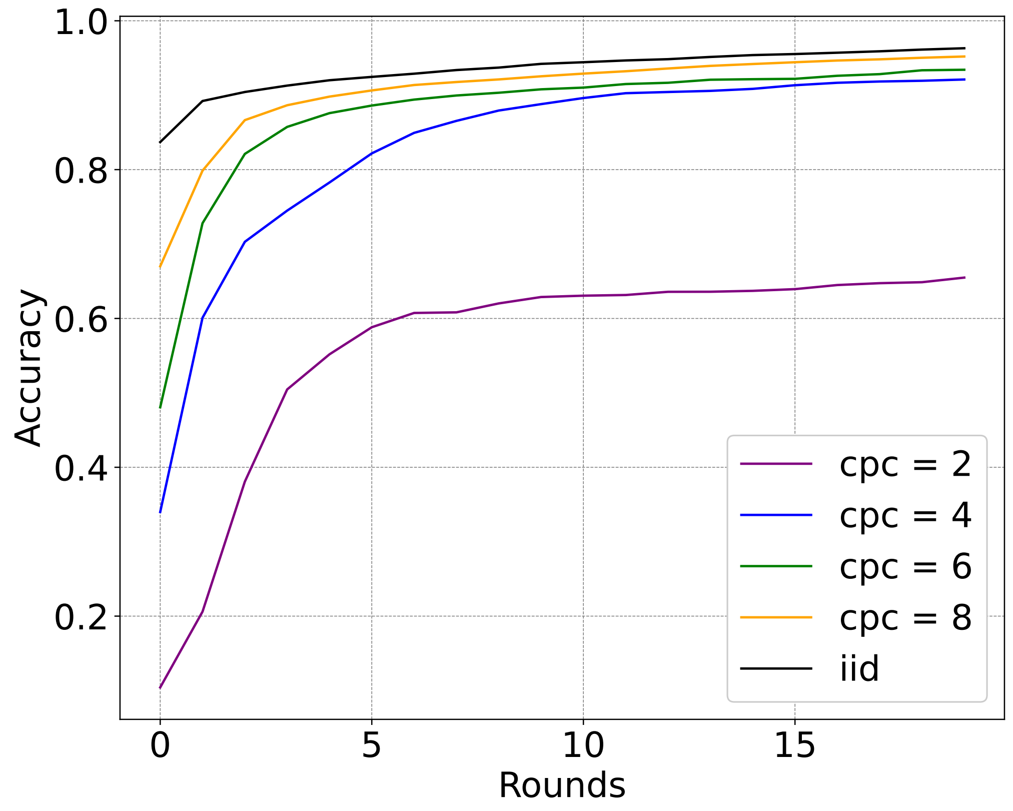


Figure 1.4: Accuracy of FedAvg on MNIST [38] (ten classes of data in total) with different *cpc* and ten clients.

low ranks' matrix and update one matrix while keeping the other constant the whole time. Both approaches can reduce the update size with an acceptable drop in accuracy.

Other than directly reducing the communication size of each round, a better optimizer can be proposed to speed up convergence so that fewer communication rounds are needed to achieve the same accuracy as the one of FedAvg [39, 40].

1.2.3 Single Point of Failure

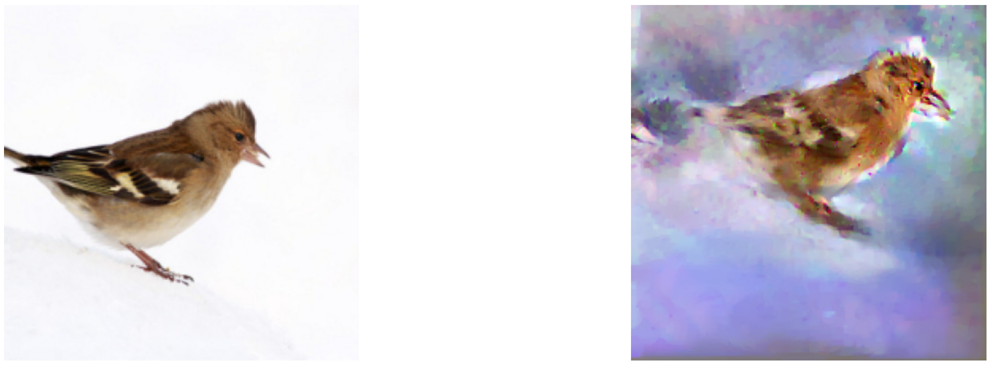
In centralized FL, the central server is heavily relied upon to perform aggregations and model broadcasting, which leads to the problem of the single point of failure (SPOF) [42, 43]. To be specific, if the central server is unable to perform aggregation, the FL cannot be conducted due to the lack of aggregation after local training, since it will have a less generalized performance. Therefore, to address this problem, many studies have proposed DFL frameworks [44,45].

1.2.4 Privacy Leakage

While most central servers are honest in performing the FL procedure (collecting, aggregating and broadcasting gradients), some may be curious about users' original data. This honest-but-curious server is also defined as the semi-honest server. As mentioned before, FL is claimed to protect privacy since no data leaves local storage. However, recent studies have shown that even from transmitted gradients, original inputs can be recovered [46,47], namely reconstruction attack. To perform the reconstruction attack, most existing research first generates fake inputs based on the FL tasks and then feeds these inputs into the FL models to compute gradients [46,47]. Following that, these works use optimization techniques, such as L-BFGS, to minimize the differences between received local gradients and generated fake gradients in order to update the fake inputs. Finally, the generated fake gradients can be very similar to the received gradients after adequate training, which means the data creating those fake gradients may be similar to the original inputs so that the privacy is violated.

In privacy-sensitive scenarios, even partially exposed features can significantly compromise data privacy. In medical IoT data applications, the patients may not want the central server to know even part of their medical conditions. Fig. 1.5 shows an example of reconstructing an image from the gradients using the work in [48]. The gradients are obtained by passing an image from the ImageNet dataset [49] through a pre-trained ResNet18 model [50]. Although the reconstructed image is blurry, it can still be identified as a bird from the contour of the object. In this example, the partial exposure of the objective's contour leads to a full leakage of the image's class, highlighting the severity of reconstruction attacks. Therefore, techniques to defend against reconstruction attacks in FL need further research.

To provide strong privacy protection for FL, three methods are mostly adopted in FL [51]. The first method is homomorphic encryption (HE) [52]. To be specific, the local clients first encrypt their local gradients before uploading them to the server. Upon receipt, the server aggregates the encrypted gradients to create the new global model. Based on the properties of the additional homomorphic encryption, the sum of encrypted data is equal to the encryption of the sum of the same data. Therefore, after decrypting the aggregated encrypted gradients, the global model is the same as the direct aggregation of all the local gradients, which can be used for further training. In this way, the server can perform the aggregation tasks without accessing the original gradients. However, original HE-



(a) The true image (b) The reconstructed image

Figure 1.5: An example of reconstructed attack [48].

based FL can result in huge computation overload when computing the encryption of the gradients every round. Additionally, there is also a lack of secure methods of generating and sharing keys to prevent servers from cooperating with malicious clients to decrypt the encrypted gradients.

The second method is secure multi-party computation (SMC), which provides a way in which every participant can use their data to jointly train a model safely when no one can be trusted. Each user knows only the input and output of the training protocol. Bonawitz et al. have proposed a practical secure aggregation framework [53]. Their framework relies on Shamir's $\mathbb{T}$-out-of-$\mathbb{N}$ secret sharing [54], which splits a secret into $\mathbb{N}$ parts so that having less than $\mathbb{T}$ parts cannot recover the secret. The main idea is to first generate a pair of public and secret keys for each user and broadcast all the public keys to everyone. Subsequently, every client splits its secret key into several parts and encrypts and signs every part with its public key. Next, each client will generate two masks, $\mathbb{S}$, which is related to the private data, and $\mathbb{B}$, using its secret key and a pseudo-random generator, and add all the masks to the private data. When aggregation is needed, the server can request ciphertexts and $\mathbb{T}$ parts of the secret keys of each user to reconstruct the text. Furthermore, the server can only receive enough shares of secret keys for aggregating while the mask is still maintained to protect privacy. After aggregating, all the masks $\mathbb{S}$ are eliminated, and the aggregated data is sent back to each client to decrypt for the next round. The framework uses Shamir secret sharing, authenticated encryption, key agreement, pseudo-random generator and signature scheme to make sure that the central server cannot trick normal clients into revealing their data. However, this framework introduces significant computation overheads. Additionally, it still needs a totally trusted server for key generation.

The third method is differential privacy (DP), which is adopted in FL [55]. DP is a rigorous mathematical tool to bound the probabilities of the outputs of two neighboring distributions (differ at one bit) as similar as each other [56]. As such, no one can tell which distribution the output is generated from. To be specific, DP is first implemented in database queries to reduce the impact of whether one certain sample exists in the database. For example, an attacker wants to know whether a person has a certain disease, but he cannot directly get the answer. To obtain that, he first asks the database how many patients of that disease exist in the database and then asks how many patients of that disease exist in the database without that person. If he gets the same answer, he knows the person does not have that disease. On the other hand, if the answers are different by one, he will know the person has the disease. By implementing DP, the attacker would have similar answers whether the person does or does not have the disease, thereby preserving privacy. In order to achieve DP, adding noise or permuting the outputs based on different probabilities can be performed.

As DP provides a strong guarantee in protecting privacy, it is adopted in FL to make two sets of gradients indistinguishable. There are generally two types of DP usage, based on where the randomization is applied, which are defined as described below.

1.2.4.1 Central DP

Centralized DP (CDP) for FL has been proposed to add noise to the aggregated gradients at the central server side [57]. Therefore, the curious clients cannot distinguish a specific client from the others by analyzing the noisy global models across multiple rounds. However, the curious server may still try to reconstruct inputs from the received local gradients.

1.2.4.2 Local DP

Local DP (LDP) for FL is achieved by adding noise to the gradients of each training batch during local training, which is also known as sample-level DP [55]. On the other hand, LDP can also be achieved by applying randomization to the local gradients over multiple epochs before uploading the gradients to the server [58], namely user-level DP (UDP). Both sample-level DP and UDP can prevent privacy leakage against curious servers. Even though sample-level DP may offer the best privacy protection effect on every sample, it has much worse accuracy and may be too excessive in privacy protection for FL. Meanwhile, UDP-based FL can hide whether a client has joined the training from other participants and prevent the server from recovering private data. Therefore, the LDP implementation in this thesis mostly focuses on UDP.

Before applying DP mechanisms for both CDP and LDP, the l_2-norm of the gradients needs to be bounded [57, 59, 60]. This process is known as clipping, which is to limit the influence of each training sample or client. After clipping, the noise is added to the clipped gradients and aggregate the noisy gradients to generate a new global model for the next round of training. If the added noise has suitable variance or the strength of probabilistic permutation is sufficient, the server and the clients cannot reconstruct identifiable inputs from the gradients.

Although the noise variance is carefully chosen, randomizing the models inevitably decreases accuracy [61]. Therefore, effective solutions to improve the overall utility of DPFL need to be explored.

1.3 Motivations and Objectives

Based on the discussion of the advantages and drawbacks of the existing FL frameworks, this thesis is driven by the need for improvement in FL. LDP is first integrated into FL in this thesis to provide a strong privacy guarantee for local users, as SMC and HE may bring a huge computation cost, which may be impractical for resource-constraint devices. Then, this thesis focuses on exploring more effective DPFL frameworks to enhance privacy protection and overall utility. In this section, the motivations of this thesis are outlined below.

The first motivation is the need for utility improvements in DPFL. To improve accuracy of DPFL, most existing work focuses on relaxing the DP bound for privacy loss so that they can achieve the same DP guarantee with decreasing required noise variance. However, the risks of the central server reconstructing identifiable inputs may be increased. Additionally, more communication rounds may be needed to achieve the same accuracy in DPFL as in non-private FL. Therefore, improving the overall utility of DPFL while maintaining the same privacy protection level needs to be considered.

The second motivation is the need for privacy loss measurement for DPFL, which can be utilized to increase accuracy. Under the first motivation, some techniques have been proposed to improve the utility and are claimed not to sacrifice privacy protection. However, they are not proven. Meanwhile, adopting DP in FL may lead to degraded accuracy. Choosing an appropriate noise variance for DPFL is worth researching. Meanwhile, most DP mechanisms require clipping on the gradients, so determining the best clipping methods for the DPFL also needs to be studied. Therefore, a privacy loss measurement methodology is needed to select the optimal DP settings in terms of clipping bounds and noise variances.

The third motivation is to propose privacy-enhancing decentralized frameworks for DPFL. Traditional FL relies on the central server to aggregate and broadcast the gradients, which introduces a heavy burden of bandwidth for the central server and even the problem of SPOF. Meanwhile, transmitting all the gradients to only one place every time may increase the success rate of reconstruction attacks. Therefore, a novel decentralized framework for DPFL needs to be designed. Furthermore, there is a lack of privacy analysis on relaxed DP bound for the decentralized DPFL framework, which may differ from the centralized ones. Thus, a privacy-enhanced DP-based DFL (DPDFL) framework with performance analysis is needed.

This thesis has the following objectives based on the aforementioned motivations. The first is to explore the area of privacy-preserving FL based on DP and identify the state-of-the-art (SOTA) solutions and their challenges. Thus, this thesis provides a literature review of DPFL algorithms. Based on the review, it is recognized that existing research on DPFL focuses on improving accuracy, reducing communication overhead and enhancing privacy protection. Building on this foundation, this thesis aims to propose new optimizers and frameworks to further advance these areas.

1.4 Research Contributions

This thesis focuses on improving privacy protection and overall performance in DPFL. As such, a privacy leakage measurement framework and three distinct DPFL frameworks are proposed. Furthermore, theoretical convergence and privacy analysis are derived for the proposed frameworks. Finally, extensive simulations for each framework are conducted to evaluate the performance by comparing the results with baselines. The contributions of this thesis are summarized as follows:

1. An LDP-based FL (LDP-FL) framework is proposed based on the Gaussian mechanism and moments accountant (MA) to track privacy loss and combined with CDP to enhance privacy protection. A new scheme is also proposed to calculate the DP noise base variance for the LDP-FL. Moreover, this framework adopts a top-sparse mechanism and a global momentum gradient descent (MGD) to improve overall utility. Experimental results show that, for non-IID MNIST data distribution in clients, the proposed FL framework achieves better performance than other DP-based FL. Meanwhile, the framework significantly reduces communication costs by using sparse gradients. This work corresponds to the publication of [62].

2. A novel LDP-based FL scheme is proposed by adding Gaussian noise to the local gradients before uploading to satisfy Rényi-DP (RDP). Second, this thesis proposes two modifications to the local objective function and detailed derivations to improve the accuracy of the noisy training. The first modification is to minimize the difference between the noisy and original gradients during training under the same DP settings. The other one is to calculate the expected change in the loss due to the noise. By incorporating the expected change into the local objective function, the FL can also minimize the loss created by noise and converge faster. Finally, multiple simulations are conducted on a multi-layer perceptron (MLP) and a convolutional neural network (CNN) to evaluate the effectiveness of the proposed framework and modifications, which can save up to 40% communication rounds to reach the same accuracy as the original DPFL. This work corresponds to the publication of [63].

3. A framework for measuring privacy loss is proposed based on reconstruction attacks, and a metric of privacy leakage measurement is designed. This thesis shows that implementing an anonymous mechanism for local clients can decrease the probability of data privacy leakage. Extensive simulations are conducted to show the effect of different DP noise and clipping settings on reconstruction attacks and FL training. The summarized findings can be used to improve FL utility under the DP mechanism and guide privacy protection level settings for a personalized privacy protection scheme. This work corresponds to the publication of [64].

4. This thesis proposes a DPDFL framework and integrates a novel anonymous mechanism into it. It considers two decentralized methods and uses the local training client selection rate (TCSR), as well as the model exchange rate (MER), to control the decentralized topology. Based on the proposed framework, a detailed privacy analysis is provided to ensure the DP guarantee and the necessary noise base variance is derived for different decentralized methods. Next, this thesis derives the convergence bound for the proposed framework in terms of the DP budget and two different client selection rates. Finally, extensive simulations are conducted to present the effectiveness of the proposed framework under different client selection rates, which further validates the theoretical results. This work corresponds to the publication of [65].

1.5 Thesis Outline

The remainder of this thesis is organized as follows:

Chapter 2 starts with the basic procedures and formulas of traditional ML and FL. The basic workflow of the reconstruction attack is presented. Next, it reviews the definitions and theorems of common DP mechanisms and DPFL. Finally, a literature review on reconstruction attacks and various frameworks of FL, especially DPFL, is introduced.

Chapter 3 proposes two different frameworks of DPFL to improve the convergence performance. It starts with the combination of LDP and CDP-based FL framework, which introduces two novel techniques to improve accuracy. Furthermore, a novel LDP-FL framework with two modifications on the local optimizer is introduced, along with derivations of the modifications. In addition, a convergence and complexity analysis is given. Finally, the evaluation and discussion of both the proposed frameworks are presented.

Chapter 4 presents a methodology for evaluating privacy loss in DPFL based on reconstruction attacks. The detailed evaluation methodology is introduced in four aspects, followed by the convergence analysis of the reconstruction attacks on the noisy gradients. Finally, it presents the simulation results and summarizes the theoretical and empirical findings.

Chapter 5 proposes a novel anonymous DPDFL framework under two decentralized methods. It first provides detailed procedures of the framework and an essential privacy analysis of the framework for both settings, followed by the convergence analysis of the proposed framework for both decentralized methods. Finally, it evaluates the effectiveness of the proposed framework and summarizes this chapter.

Chapter 6 concludes the proposed works in this thesis and discusses the challenges and potential future work in the area of DPFL.

1.6 Publications

1. **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.

2. **S. Weng**, L. Zhang, X. Zhang, and M. A. Imran, "Faster convergence on differential privacy-based federated learning," *IEEE Internet of Things Journal*, vol. 11, no. 12, pp. 22578-22589, June 15, 2024.

3. **S. Weng**, Y. Gou, L. Zhang, and M. A. Imran, "Evaluating privacy loss in differential privacy-based federated learning," in *Future Generation Computer Systems*, Major

revision.

4. **S. Weng**, L. Zhang, Y. Gou, N. Truong, and M. A. Imran, “Anonymous differential privacy-based decentralized federated learning with performance analysis,” under review in *IEEE Transactions on Dependable and Secure Computing*.

5. Y. Gou, **S. Weng**, M. A. Imran, and L. Zhang, “Voting consensus-based decentralized federated learning,” *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16267-16278, May 1, 2024.

6. Y. Gou, **S. Weng**, D. Liu, M. A. Imran, and L. Zhang, “Distilling knowledge in decentralized federated learning with enhanced personalization,” under review in *2025 International Conference on Distributed Computing Systems*.

7. Huanyu Wu, **Shangyin Weng**, Lei Zhang, and Muhammad Ali Imran, "ReDAG: An Adaptable Redactable Directed Acyclic Graph-based Distributed Ledger System", Under review in *2025 IEEE International Conference on Distributed Computing Systems*.

Chapter 2

Background and Related Work

To address the challenges of DPFL, it is essential to review the basic knowledge of DPFL and its related fields before introducing the proposed frameworks in this thesis. Therefore, as DL is the fundamental model for local training in FL, the general background of DL, especially the feedforward neural network (FNN) and backpropagation, is first introduced in this chapter, followed by the procedures of FL and reconstruction attacks. Subsequently, the definitions and principles of different DP mechanisms are introduced. In addition, the general process of DPFL is demonstrated. Finally, the literature on DPFL is reviewed in this chapter.

2.1 Background

2.1.1 Deep Learning

Since the key idea of FL is to allow local clients to train DL models locally, DL is first introduced in this subsection.

DL constitutes a significant subset within the field of ML and has emerged as the most important and popular research area in AI and ML. DL follows the principles of artificial neural network (ANN), which is inspired by the human brain's structure [66]. It consists of multiple layers of fully connected artificial neurons to process data sequentially. Each neuron receives inputs, computes outputs, and sends the outputs to the next layer for further computation with other neurons' outputs. The outputs from the final layer's neurons will be utilized based on the tasks. Three general types of DL are introduced in the remainder of this subsection.

2.1.1.1 Feedforward Neural Network

The FNN, also known as the MLP, is a fundamental model for DL, as most other DL models are based on FNNs [67]. The FNN only allows the data to flow directly from input to output without any recurrent operations. There are three types of layers in FNN:

- **Input layer:** The input layer is the first layer of the FNN to receive the input data.
- **Final layer:** The final layer, also known as the output layer, processes the final outputs based on the tasks, such as classification tasks.
- **Hidden layers:** The intermediate layers between the input layer and the final layer process the most important computations to extract features from the inputs for learning. Multiple hidden layers can be implemented, leading to a deep neural network (DNN).

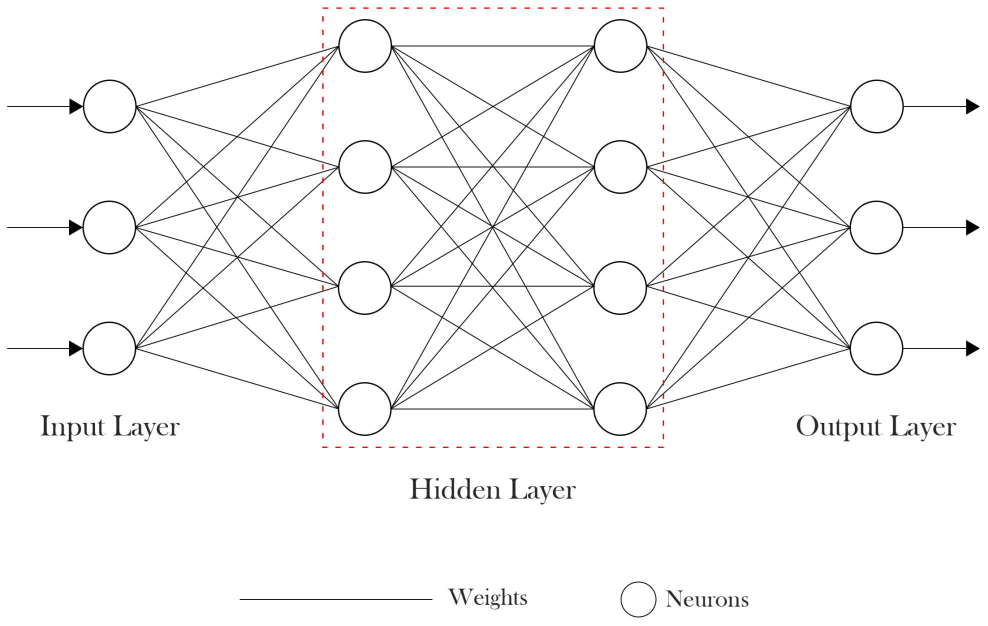


Figure 2.1: An example of an FNN with two hidden layers.

A typical FNN framework with two hidden layers is shown in Fig. 2.1. After passing the input layer, the computation follows:

$$z = w * x + b, \tag{2.1}$$

$$a = g_{act}(z), \tag{2.2}$$

where a, z are intermediate parameters for training, $g_{act}(z)$ is the activation function to provide a non-linear computation for the FNN, w is the weights between neurons to control the input's strength, and b is the bias to help improve the fit of the predicted value to the true value. For the subsequent layers, the computation is similar to (2.1) and (2.2), while x is replaced by a from the previous computation. As regards the activation functions for hidden layers, there are many choices, including Sigmoid, Tanh and rectified linear unit (ReLU), where ReLU is the most widely used since it brings higher accuracy of learning in most cases. Meanwhile, the activation function of the output layer depends on the learning tasks. For example, Softmax is used for multi-class classifications, and Sigmoid is used for binary and multi-label classifications. Finally, by integrating the computations of all layers, the system can generate an output based on the current input. In other words, the purpose of an FNN is to construct a complex non-linear multi-variable representation to map the input to the correct output by combining simple computations [68].

After passing the input through the model to infer a predicted value, which is often different from the true value at the beginning of the training, the distance between the predicted value and the true value needs to be minimized in order to provide a better outcome. Therefore, training mechanisms are proposed to adaptively adjust the weights and biases of the model, enabling it to produce predicted values that are more similar to the true ones. To evaluate the model, a loss function is needed to measure the difference between the predicted and true values, which are chosen based on the tasks. Common loss functions include mean square error, cross-entropy, etc. In order to minimize the loss, $\mathbb{L}$, backpropagation is introduced [69]. The basic idea of backpropagation is to compute the partial derivatives (gradients) of the parameters from the loss and update the parameters with the gradients in the direction that reduces the loss. To compute each layer's gradients, the chain rule of computing derivatives for complex functions is primarily used. The gradients of the final output ∇a_K after the activation function are first calculated, where K is the final layer's index. Then, the gradients of the intermediate parameters ∇z_K can be computed following the chain rule. Subsequently, the gradients of weights w_K can be computed as:

$$\frac{\partial \mathbb{L}}{\partial w_K} = \frac{\partial \mathbb{L}}{\partial a_K} \frac{\partial a_K}{\partial z_K} \frac{\partial z_K}{\partial w_K}. \tag{2.3}$$

The gradients for b are also computed similarly. By extending the chain rule, all the previous layers' gradients can be computed. As regards the activation function of the output layer, this thesis focuses on multi-class classification so that Softmax is used, which is defined to output the probability of each class for every input as:

$$a_K^o = Softmax(z_K^o) = \frac{e^{z_K^o}}{\sum_{o=1}^{N_c} e^{z_K^o}}, \tag{2.4}$$

where N_c is the total number of classes, and z_K^o is the output of the o-th class before activation based on the input. This thesis focuses on multi-class classification, where categorical cross-entropy is the most suitable choice for the loss function. The categorical cross-entropy is defined as:

$$\mathbb{L} = -\sum_{o}^{N_c} Y_o \log(a_K^o), \tag{2.5}$$

where Y_o is the true output of o-th class.

In summary, the overall process of backpropagation can be computed by taking the derivatives of each feedforward computation as follows:

$$dz_K = a_K - Y, \tag{2.6}$$
$$dz_k = da_k \times g'_{act_k}(z_k), \tag{2.7}$$
$$dw_k = \frac{1}{n'} dz_k \cdot (a_{k-1})^T, \tag{2.8}$$
$$db_k = \frac{1}{n'} \sum^{j=n'} dz_k^j, \tag{2.9}$$
$$da_{k-1} = (w_k)^T \cdot dz_k, \tag{2.10}$$

where d represents the gradients for the corresponding parameters, k is the index of the layer, n' is the number of the inputs, $\times$ means matrix-wise multiplication and $(a_{k-1})^T, (w_k)^T$ is the transpose operation of a_{k-1}, w_k, respectively.

After computing the gradients for every layer, the parameters are updated as follows:

$$w^t = w^{t-1} - \gamma dw^t, \tag{2.11}$$
$$b^t = b^{t-1} - \gamma db^t, \tag{2.12}$$

where w^t and b^t are the model parameters of the t-th training round, γ is the learning rate and dw^t, db^t are the gradients of all layers. The training will continue for multiple rounds until the model converges, meaning it reaches the lowest loss or the largest accuracy.

The aforementioned training process is also known as gradient descent (GD) [70].

Typically, the training rounds for GD are measured in epochs, during which the entire dataset is iterated to compute the gradients. Initially, the entire dataset is used as input for the feedforward in one step, which leads to complex computation and long training time. In order to speed up training, stochastic GD (SGD) is proposed, which uses one sample in every step to train, compute gradients, and update parameters until the whole dataset is iterated. In SGD, there are n steps of training in every epoch for a dataset with n samples [70]. Nevertheless, using only one sample for training may cause the gradients to descend in various directions so that the convergence of the training is slowed. Therefore, mini-batch SGD is proposed, where the whole dataset is split into multiple batches, and the gradients are computed from every batch in one epoch of training [70]. The mini-batch SGD is mostly used in training DL since it normally achieves faster convergence than the other two. For convenience, this thesis will use the term SGD to refer to mini-batch SGD in the following chapters, aligning with its common use in current research.

In addition to different input data settings, many different optimizers are proposed to improve the convergence performance of FNN. For example, MGD is proposed by adding a fraction of the previous update to the current update, also known as momentum, helping to smooth out oscillations and speed up learning [70]. To be specific, the gradients of each training step are the exponentially weighted moving average of all previous gradients. Besides, Adam optimizer introduces the second-order momentum into the MGD to adapt the learning rates, speeding up convergence [71].

2.1.1.2 Convolution Neural Network

In order to improve the performance of FNN on images, CNN is proposed, which is designed to automatically and adaptively learn spatial hierarchies of features through convolutional layers [72]. The convolutional layers are computed by applying a set of filters (kernels) to the input data, performing element-wise multiplications, sliding the filters over the inputs and finally summing up all the multiplication results to form the feature maps. This process can be viewed as computing the discrete convolution between the input data and the filters. The feature maps are fed into pooling layers to reduce the dimension by splitting the feature maps into multiple small areas and choosing the averages or maximum values in every small area of the feature maps. Multiple convolution layers can be added based on the needs. Finally, the feature maps are first flattened and fed into a FNN for classification.

2.1.2 Federated Learning Workflow

In this subsection, the basic workflow and formulas of FL are introduced. The goal of the traditional FL is to train a global model based on the following objective [14]:

$$arg \min_{W^t} \sum_{i=1}^{M} \frac{1}{M} \mathbb{L}(W_i^t), \tag{2.13}$$

where W^t is the global model in the t-th round, W_i^t is the local model of the i-th client in the t-th round and M is the number of the local clients.

FL is performed by the following steps:

Step 1: The server starts the FL process and initializes a global model W^0, which is broadcast to all the clients.

Step 2: Every client i trains the W^t for multiple epochs with its local data to obtain a new local model and compute the local gradients $\nabla W_i^t = W^t - W_i^t$ in the t-th round, which are sent back to the server.

Step 3: The server computes the average of the received local gradients and generates the new global model as $W^{t+1} = W^t - \sum_{i=1}^{M} \frac{1}{M} \nabla W_i^t$.

Step 4: The server broadcasts the new global model W^{t+1} to clients for the next round of training.

Steps 2-4 may be repeated until the global model achieves acceptable accuracy on the test data.

2.1.3 Reconstruction Attacks

In this subsection, the reconstruction algorithm in [48] is introduced, which will be implemented and used to derive theoretical results in Chapter 4. The algorithm first generates some random images, namely dummy inputs, which are input into the FL model to generate dummy gradients. It calculates the cosine similarity between the dummy gradients and the true gradients. Following that, the algorithm defines the objective function for the dummy inputs based on the cosine similarity and total variation (TV) [73] as follows [48]:

$$J = \arg \min_{x \in (0,1)^n} 1 - \frac{< g(x), g(x^*) >}{||g(x)||||g(x^*)||} + \alpha_1 TV(x), \tag{2.14}$$

where J is the objective function of the attack, $g(x)$ is the dummy gradients, $g(x^*)$ is the transmitted gradients, and α_1 is used to control TV. The objective function is optimized using the signed Adam optimizer, which means the first order of the momentum of the

Adam optimizer is either 1 or -1, and the second order of the momentum is 1. Meanwhile, the accumulative moment is still unsigned. The unsigned Adam optimizer is defined as follows:

$$v_1^t = (1-\beta_1)v_1^{t-1} + \beta_1 g(t), \tag{2.15}$$

$$v_2^t = (1-\beta_2)v_2^{t-1} + \beta_2 g(t)^2, \tag{2.16}$$

$$\hat{v}_1^t = \frac{v_1^t}{1-\beta_1}, \tag{2.17}$$

$$\hat{v}_2^t = \frac{v_2^t}{1-\beta_2}, \tag{2.18}$$

$$x_{t+1} = x_t - \frac{\gamma * \hat{v}_1^t}{\sqrt{\hat{v}_2^t + st}}, \tag{2.19}$$

where v_1^0 and v_2^0 are zero, v_1^t is the first order momentum, v_2^t is the second order momentum, $g(t)$ is the gradients of the t-th round, β_1 and β_2 are factors to control the momentum, and st is a small term, typically set to 10^{-8}, to prevent the denominator from being zero.

2.1.4 Differential Privacy

In this subsection, the basic concepts, theorems and lemmas of DP mechanisms are introduced.

The DP mechanism guarantees privacy protection when sharing data. To be specific, two adjacent databases, D and D', differ from each other at only one data point. Applying a DP scheme to perturb the original values can make the output of these two databases indistinguishable [56]. To measure the difference between two output distributions, Max-divergence is used, which is formalized as:

Definition 1 *(Max-divergence [56]).*

$$D_\infty(Y||Z) = \max_{y\in Y} ln[\frac{Pr[Y=y]}{Pr[Z=y]}], \tag{2.20}$$

where Pr is the probability, y is the output, and Y,Z are two distributions.

If the Max-divergence between two distributions is bounded by ε (where $\varepsilon \geq 0$) and e is introduced on both sides of the Max-divergence, as shown in Fig. 2.2, ε-DP can be ensured between these two distributions. The formal definition of an ε-DP mechanism is given as follows:

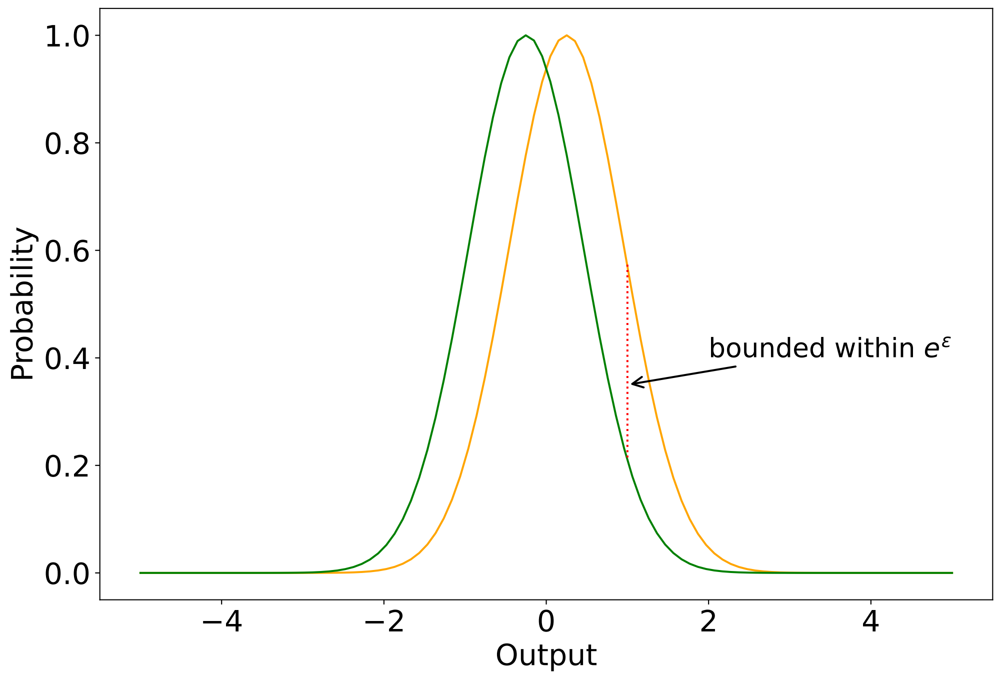


Figure 2.2: The probability of outputs from two distributions.

Definition 2 *(ε-DP [56]). A randomized mechanism R achieves ε-DP if for any two neighbouring distributions, D, D', and any possible output, $S' \subset \mathbb{R}$, it satisfies the following constraints:*

$$Pr\left[R(D) \in S'\right] \leq e^{\varepsilon} Pr\left[R(D') \in S'\right]. \tag{2.21}$$

To achieve ε-DP, l_1-sensitivity first needs to be computed, which represents the maximum change in the outputs between two datasets differing at one sample. It is defined as follows:

Definition 3 *(l_1-sensitivity [56]). For two neighbouring datasets D, D', the l_1-sensitivity Δf of a function f is defined as $\Delta f = sup_{D,D'}||f(D) - f(D')||_1$.*

After calculating the l_1-sensitivity, the Laplace mechanism and exponential mechanism can be applied. For the Laplace mechanism, Laplace noise can be added to the data, which follows the Laplace distribution with its probability density function as $Lap(x|b_1) = \frac{1}{2b_1} exp(-\frac{|x|}{b_1})$, where $b_1 = \frac{\Delta f}{\varepsilon}$. For the exponential mechanism, a score function *score* must first be introduced, which assigns scores to the input based on the tasks. Subsequently, the exponential mechanism creates the output based on a probability that is proportional to

$exp(\frac{\varepsilon * score(x,r)}{2\Delta score})$, where x is the input, r is a possible output, and $\Delta score$ is the sensitivity of the score function.

However, this general ε-DP is too strict to be applied in certain cases while maintaining acceptable utility. Therefore, an approximate mechanism, (ε, δ)-DP, is proposed. In (ε, δ)-DP, for two neighboring datasets D, D', the difference between their outputs can also be bounded by e^{ε}, while δ represents the probability that the difference cannot be bounded by e^{ε}. (ε, δ)-DP is formalized as follows:

Definition 4 *((ε, δ)-DP [56]). For any $\varepsilon > 0$ and $0 \leq \delta \leq 1$, a randomized mechanism R achieves (ε, δ)-DP if $\forall D, D', \forall S' \subset \mathbb{R}$ and satisfies the following constraints:*

$$Pr\left[R(D) \in S'\right] \leq e^{\varepsilon} Pr\left[R(D') \in S'\right] + \delta. \tag{2.22}$$

To satisfy (ε, δ)-DP, it first needs to compute l_2-sensitivity as follows:

Definition 5 *(l_2-sensitivity [56]). For two neighbouring datasets D, D', the l_2-sensitivity Δf of a function f is defined as $\Delta f = sup_{D,D'} ||f(D) - f(D')||_2$.*

With l_2-sensitivity, (ε, δ)-DP is generally achieved by adding Gaussian noise $N(0, \sigma^2)$, where σ is a base noise variance and needs to satisfy $\sigma \geq c\Delta f / \varepsilon$ for $c > \sqrt{2ln(1.25/\delta)}$.

Some basic properties and theorems of applying DP are also introduced. The first one is the post-processing, which is defined as:

Lemma 1 *(Post-processing [56]). Let $\mathbb{M} : \mathbb{X} \to \mathbb{Y}$ be a mechanism satisfying (ε, δ)-DP and let $f : \mathbb{Y} \to \mathbb{Z}$ be any random function. Then, $f \circ \mathbb{M} : \mathbb{X} \to \mathbb{Z}$ also satisfies (ε, δ)-DP.*

Due to this property, privacy protection is not affected by any post-processing. In LDP-FL, the noisy gradients first need to be aggregated, and a new model is computed with the noisy gradients and the global model from the previous round. All these operations are post-processing of the DP, which guarantees that LDP-FL has the desired privacy protection level after FL training.

In many applications, including FL and ML, where DP needs to be applied multiple times, the privacy loss is accumulated. To record accumulative privacy, the composition theorem is introduced.

Theorem 1 *(Composition theorem [56]). Let R_i be ε_i-DP for $i \in [1, k']$. Then, if a mechanism is defined to be $R = (M_1, M_2, ..M_{k'})$, R is $(\sum_{i=1}^{k'} \varepsilon_i)$-DP.*

Next, to provide a tighter bound for calculating the privacy loss of the Gaussian mechanism, an advanced composition theorem has been proposed, which is defined as follows:

Theorem 2 *(Advanced composition theorem [56]). For all $\varepsilon, \delta, \delta' \geq 0$, k'-fold multiple usage of the (ε, δ)-DP satisfies $(\varepsilon', k'\delta + \delta')$-DP if:*

$$\varepsilon' = \sqrt{k' \ln(1/\delta')\varepsilon} + k'\varepsilon(e^{\varepsilon} - 1). \tag{2.23}$$

To provide a tighter bound under k'-fold usage than the above composition theorems, zero-concentrated-DP and RDP are proposed. Unlike traditional DP, they rely on Rényi divergence to measure the difference of the output distributions, which is formalized as follows:

Definition 6 *(Rényi Divergence [74]).*

$$D_{\alpha}(Y||Z) \cong \frac{1}{\alpha - 1} \ln \mathbb{E}_{y \in Y} [\frac{Pr(Y = y)}{Pr(Z = y)}]^{\alpha}, \tag{2.24}$$

where $\alpha \in (1, \infty)$ and $\mathbb{E}$ is the expectation.

RDP is proposed by focusing on one moment at a time, which is formally defined as follows:

Definition 7 *(RDP [75]). A randomized mechanism R achieves (α, ε)-RDP if $\forall D, D'$, $\forall S' \subset \mathbb{R}$, it satisfies the following constraints:*

$$Pr\left[R(D) \in S'\right] \leq (e^{\varepsilon} Pr\left[R(D') \in S'\right])^{\frac{\alpha-1}{\alpha}}, \tag{2.25}$$

where $\varepsilon > 0$ and $\alpha \in (1, +\infty)$. It is also proved that if a mechanism satisfies (α, ε)-RDP, it also satisfies $(\varepsilon + \frac{\log \frac{1}{\delta}}{\alpha - 1}, \delta)$-DP.

Next, some useful lemmas for RDP are introduced, which will be used to derive key theoretical results for privacy analysis in Chapter 5.

Lemma 2 *(Gaussian mechanism for RDP [75]). Given a function $f : D \to \mathbb{R}$, the Gaussian mechanism $R = f(D) + N(0, \sigma^2 \Delta^2 f)$ satisfies $(\alpha, \frac{\alpha}{2\sigma^2})$-RDP.*

Lemma 3 *(Subsampling mechanism for RDP [76, 77]). If R is applied to a subset of samples using uniform sampling at a rate r without replacement and R satisfies (α, ε)-DP,* *the new randomized mechanism R_{sample} satisfies (α, ε')-RDP, where*

$$\varepsilon' \leq \frac{1}{\alpha-1}\log((1+r^2\binom{\alpha}{2}\min\{4(e^{\lambda(2)}-1), 2e^{\lambda(2)}\}\sum_{j=3}^{\alpha} r^j\binom{\alpha}{j}2e^{(j-1)\lambda(j)})), \tag{2.26}$$

and $\lambda(j) = j/2\sigma^2$. If $\sigma^2 \geq 0.7$ and $\alpha \leq (2/3)\sigma^2\Delta^2 R(x)log(1/r\alpha(1+\sigma^2))+1$, R_{sample} satisfies $(\alpha, 3.5r^2\alpha/\sigma^2)$-RDP.

Lemma 4 *(Composition theorem for RDP [75]). For any randomized mechanism R_1 and R_2 applied on the same dataset, if R_1 satisfies (α, ε_1)-RDP and R_2 satisfies (α, ε_2)-RDP, the composition of R_1 and R_2 satisfies $(\alpha, \varepsilon_1 + \varepsilon_2)$-RDP.*

2.1.5 Differential Privacy-based Federated Learning Implementation

In this subsection, the general implementation of DP in FL algorithms is introduced. To satisfy DP in learning algorithms, the gradients first need to be clipped. The clipping mechanism varies for CDP-based FL (CDP-FL), and LDP-FL, which are introduced in the remainder of this subsection. An overview of the LDP-FL and CDP-FL framework is given in Fig. 2.3 to demonstrate where the DP is applied.

2.1.5.1 Central Differential Privacy-based Federated Learning

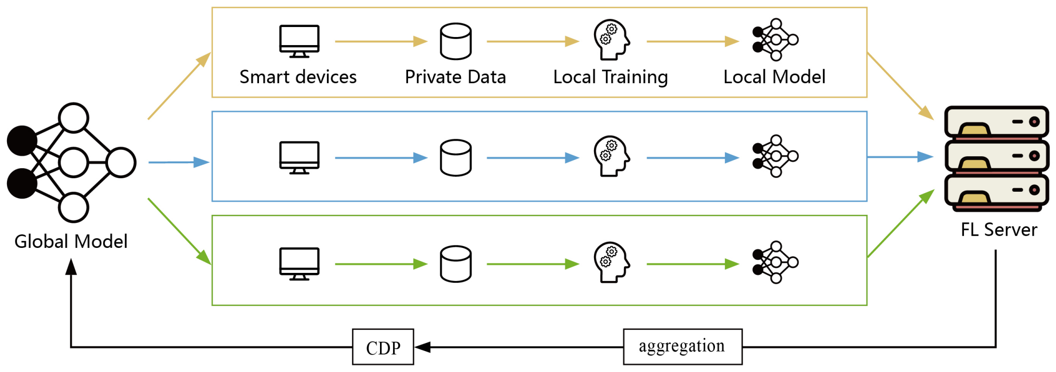
In CDP-FL [57], the authors aim to protect the clients from being identified as participating in training in FL, where all clients are assumed to be semi-honest. Therefore, CDP-FL clips each client's gradients W_i^t as follows:

$$\hat{W}_i^t = \frac{W_i^t}{max(1, \frac{||W_i^t||_2}{median(||W_i^t||_2)})}, \tag{2.27}$$

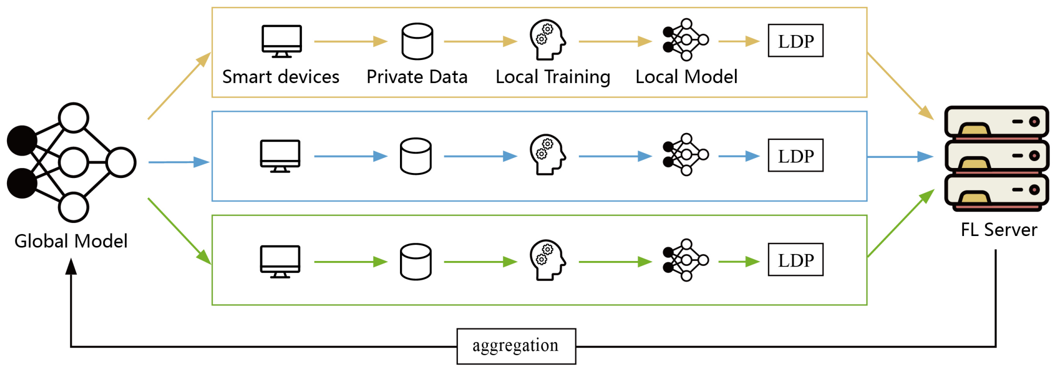
where $median(||W_i^t||_2)$ is the median value of all the received gradients in each round. The required noise is added to the gradients after aggregating the clipped gradients. Then, the noisy gradients are broadcast to participants for training.

2.1.5.2 Local Differential Privacy-based Federated Learning

In LDP-FL, the authors aim to protect the private data and participation of every client from a curious server, where UDP is applied to satisfy LDP. Before applying DP, each



(a) CDP-FL



(b) LDP-FL

Figure 2.3: An overview of DPFL frameworks.

client also needs to clip its gradients as follows:

$$\hat{W}_i^t = \frac{W_i^t}{max(1, \frac{||W_i^t||_2}{C_i})}, \tag{2.28}$$

where C_i can be the median of the gradients of the i-th client during training or a fixed constant. The noise is added to the local gradients before uploading them to the server for aggregation.

2.2 Related Work

2.2.1 Federated Learning

With the emerging development of ML and the increasing attention on privacy, FedAvg has been proposed [14]. However, it still has drawbacks, including poor convergence performance under heterogeneous data distribution and even privacy leakage. In this subsection, the development of fundamental FL frameworks is introduced.

Zhao et al. show that the decrease in the accuracy of FL on non-IID data is caused by the divergence of the model, and they propose improving the accuracy by sharing a small global dataset for local clients to train on [78]. To further improve the convergence performance of FL, Fed-Prox [39] has been proposed to achieve faster convergence under heterogeneous systems by adding a proximal term related to the difference between the global model and the local model. Furthermore, it considers the devices' heterogeneity and proposes an adaptive training setting scheme for local devices to obtain optimal performance according to their capability of computing and power. The authors in [40] present the issue of local updates drifting from the global model and propose SCAFFOLD to force the local update to move towards the global model by using Control Variate. Wang et al. propose a layer-wise FL algorithm called Matched Averaging by matching similar neurons for MLP, matching filters for CNN and matching hidden states for RNN in local clients before averaging them [79]. Li et al. propose a model-contrastive FL framework by using the similarity among the models to manage the local training [80]. Acar et al. propose a novel dynamical regularization method for FL by dynamically updating the risk objective for each local client to ensure that the local model converges to the stationary points of the global model [81].

In addition, the authors in [82] adopt MGD in local training to accelerate FL training. Mills et al. adopt the Adam optimizer in FedAvg along with a compression mechanism, which not only reduces the convergence time but also reduces communication sizes in each round [83]. Reddi et al. introduce several adaptive optimizers into FL, including Adagrad, Adam, and Yogi, and derive the corresponding convergence bound with non-IID data for non-convex settings in these FL frameworks [84]. Their work also characterizes the trade-off between client heterogeneity and communication efficiency. In addition to using different optimization, several personalized FL frameworks are proposed to improve the local test accuracy by separately designing local and global models [85, 86].

2.2.2 Gradients Reconstruction

Although FL is claimed to protect data privacy, several studies have shown important information can be obtained from the gradients [87–91]. The authors in [87] use a generative adversarial network (GAN) to generate images similar to real ones. However, their algorithm only works when all the classes are similar to each other, such as face recognition or number classification tasks. Melis et al. conduct membership attacks in collaborative learning and demonstrate that their work can also reveal unimportant features, such as whether a person wears glasses in a human gender classifier [88]. The authors in [89] not only successfully recover the exact true samples using GAN but also reveal specific local clients' data, breaching identity-level privacy.

In [90], the authors propose Deep Leakage from gradients (DLG) to first generate dummy images and compute the dummy gradients through the same learning model. By optimizing the Euclidean distance between the dummy gradients and true gradients, they successfully recover the true images. In addition to DLG, the authors in [91] propose improved-DLG to recover the ground truth label of the true data. However, the aforementioned works rely on an L-BFGS optimizer, which is computationally expensive. Moreover, these works may fail because of a bad initialization and may also perform poorly, when the training batch size is larger than one and the model is deeper.

The authors in [48] suggest that previous works mostly focus on the magnitude of the difference between the true and dummy gradients, while the high-dimensional direction of the difference is also important. Therefore, they propose using cosine similarity to measure the difference and a signed Adam optimizer for optimization, which can recover most true images even if the model has a large number of layers. However, their work fails to recover images with high quality when the size of the batch is larger than eight, and they assume that the attacker knows the BatchNorm statistics.

Huang et al. make two strong assumptions for existing reconstruction attacks: 1) the attackers do not know the BatchNorm statistics, and 2) they do not have the private labels [92]. They demonstrate that these statistics and labels are not necessary to transmit during FL training but can greatly improve the quality of the reconstructed images. They also systematically evaluate the effectiveness of defense for privacy leakage, including gradient pruning and weak encryption on the existing reconstruction attacks under those two assumptions.

Therefore, Yin et al. propose DeepInversion by inferring the BatchNorm statistics of the original gradients with the noisy gradients' mean and variance and minimizing the difference between the inferred BatchNorm statistics of the fake and real gradients [46].

In addition to DeepInversion, they propose adaptive DeepInversion by introducing a new loss term to encourage disagreement on the synthesized images for GAN-based reconstruction attacks, which can improve the attack performance. The authors in [47] propose GradInversion, which allows the attack algorithm's gradient descent in various directions at a time and bound all the directions by adding a group consistency regularization term. Their method can reconstruct high-quality images from gradients obtained from a very large batch of data. Additionally, they propose a novel batch label restoration mechanism by using the gradients of the final fully connected layer, achieving more than 97% of label recovery accuracy for both the training set and validation set.

2.2.3 Differential Privacy-based Federated Learning

As introduced previously, FL alone cannot protect data privacy, and HE and SMC generates a heavy computation workload, which may be unsuitable for local devices with limited resources. Therefore, DP is applied in FL to ensure strong data privacy protection. In this subsection, the implementations and improvements of DPFL frameworks are introduced, with a sole focus on HFL.

With regard to privacy protection for learning algorithms, Martin et al. [93] have applied DP into single-end DL, namely DP-SGD. To record the accumulative privacy loss during the multiple DP procedures, they propose MA, which has a tighter bound on privacy loss for DP-based learning, compared to the DP advanced composition theorem [56].

Geyer et al. [57] propose a general CDP-based FL framework to add noise on the aggregated clipped gradients on the server side, aimed at hiding every client's participation in training from curious local clients. Additionally, their work has a significant accuracy decrease. Yang et al. propose a novel method that adds noise, which is positive and can be arbitrarily large, to the global models to provide a strong privacy guarantee [94]. To improve accuracy, they propose a novel technique to allow the central server to remove the added noise on the aggregated gradients while maintaining a high level of privacy protection under reconstruction attacks from curious clients.

However, the CDF-FL framework cannot prevent curious servers from recovering the original data. As a result, LDP is adopted to randomize local gradients to protect data privacy [59, 95]. For instance, Sun et al. propose an LDP framework that perturbs the gradients within an adaptive range with different probabilities in terms of the DP noise and clipping settings [59]. In addition, they implement a novel parameter-shuffling mechanism by splitting the gradients into multiple parts and shuffling the gradients part-wise. Aside from perturbing the gradient with probabilities, the authors in [95] propose adding Laplace noise to the local gradients before uploading them. However, their frameworks satisfy the ε-DP, which may be too strict to guarantee feasibility in real-world applications.

On the other hand, the Gaussian mechanism is widely adopted in DPFL by clipping the local gradients and adding Gaussian noise to them before uploading to the server [55]. McMahan et al. also propose a LDP-FL framework, which combines the DP-SGD and FedAvg, known as DP-FedAvg [96]. Their work adopts gradient clipping during client training and investigates two different types of clipping: per-layer clipping and flat clipping (applied to the whole model). The primary focus of their work is on differential private natural language processing tasks, where they achieve acceptable accuracy, compared to non-private learning models. In addition to adding Gaussian noise, the authors in [97] introduce a multiplicative perturbation mechanism to obfuscate the local gradients by multiplying the noise with the gradients.

To prevent privacy leakage from both curious servers and clients, many studies have combined LDP and CDP in FL by adding noise to the local gradients on the client side and aggregated gradients on the server side [61,98,99]. Bernau et al. empirically show that the noise added on the local gradients provides a weak level of privacy protection for CDP, as demonstrated through a white-box membership inference attack [100]. Therefore, a smaller noise scale is needed for aggregated gradients to achieve the given CDP guarantee [61,98,99]. Based on this finding, several studies investigate how to compute the essential noise scale and derive the corresponding convergence bound under the privacy settings [61,98,99]. Additionally, Wei et al. [61] present the convergence bound for different client selection rates for centralized DPFL and prove that there is an optimal communication round for the largest accuracy under specific noise parameters. Zhou et al. also derive the optimal number of training rounds for local clients, given the DP guarantee and total FL communication rounds [99]. To reduce communication costs, they propose a novel scheduling mechanism for local client participation.

Even though DP can protect private data from adversaries, the DPFL frameworks have a degraded convergence in terms of longer convergence time and lower accuracy due to the randomization of the gradients [55]. Therefore, improving the performance of DPFL is worth researching.

Extensive studies have evaluated the privacy protection of DPFL and researched the relationship between privacy protection and utility [61, 101–103]. Wei et al. propose a framework to evaluate the gradient leakage of non-private FL under different settings based on reconstruction attacks [104]. They also study how different compression ratios to transmitted gradients affect privacy leakage. The studies [61, 101] have characterized the trade-off between privacy protection and accuracy under different privacy protection levels to show insight into DP budget selection. Zhang et al. propose a framework to formulate the trade-off from the information-theoretic perspective between privacy and utility loss in FL under multiple privacy protection mechanisms, including DP, sparsity and HE [102]. Moreover, some work has evaluated the impact of other attacks on DPFL. Lu et al. quantify the accuracy-privacy trade-off based on membership inference attacks [103]. In addition, Naseri et al. have demonstrated that applying CDP and LDP in FL can prevent the models from backdoor attacks, while none of them can stop property inference attacks [101].

Current research also focuses on the relaxation of privacy guarantees in order to reduce privacy loss and the required noise scale. Based on (ε,δ)-DP, some studies have implemented various DP guarantees in FL to provide tighter privacy guarantees and more flexible trade-offs between utility and privacy, especially in scenarios involving continuous usage. For instance, Triastcyn et al. improve the DPFL framework by utilizing Bayesian DP, which focuses on the prior distribution of data [98]. By relaxing the privacy budget allocation to obtain more FL training rounds, their work can improve the accuracy of DPFL. The authors in [105–107] propose a DPFL framework with relaxed privacy bound based on RDP, which measures the privacy loss using Rényi divergence instead of Max-divergence in the standard DP guarantee. The authors also derive the convergence bound for their proposed frameworks.

Some research proposes adaptive DP mechanisms for DP-based learning in order to improve accuracy [108–114]. For example, Pichapati et al. propose a coordinate-wise adaptive gradient clipping mechanism for DP-SGD to reduce the necessary noise and improve the convergence performance [108]. In addition, the authors in [109] propose a personalized DPFL framework to allow local clients to choose their own privacy budget based on their needs. By decreasing their own required privacy budget, accuracy can be improved. Hu et al. utilize the clustering technique and an objective function based on standard deviation to adaptively tune the clipping bound [111]. Golatkar et al. propose a novel DPFL framework by incorporating cross-modal zero-shot learning on public data prior to private fine-tuning, which can significantly decrease the training loss of DPFL [110]. The authors in [112] propose a novel DPFL mechanism, which derives the importance factors for the gradients based on analyzing the value of the size of the gradients, the absolute value of the model parameters, and the difference between the direction of the global and local updates. The important factors are then used to compute the DP noise. The authors in [113] design a novel method to adaptively adjust the value of the clipping bound by tracking a given quantile of the update norm distribution during DPFL training. Yang et al. propose a novel personalized DP-FL framework, which utilizes fisher information to retain more important information for the local personalized model and only uploads less important information to the server so that less noise can be introduced and local accuracy can be enhanced [114]. To further improve the accuracy degradation caused by the clipping, they propose two constraints for the l_2-norm of local models.

Furthermore, some studies propose different mechanisms to enhance privacy protection and provide a relaxed bound on privacy budget. The authors in [115, 116] adopt sub-sampling of clients in each round, where a subset of data is sampled uniformly from the entire dataset for each round's training. This sampling mechanism can reduce the privacy budget, ε, by a factor proportional to the sample rate, which is theoretically proven for Laplace, Gaussian and RDP mechanisms in DP-based learning. In DPFL, the server can choose a subset of clients uniformly in each training round, which is similar to the sub-sampling mechanism if each client is regarded as a data unit [76]. Therefore, client-level sampling has also been proven to decrease the privacy budget in proportion to the sample rate. In addition, the authors in [117, 118] propose using a trusted server to shuffle the noisy gradients of all clients before sending them to the central server. The shuffling mechanism is theoretically proven to decrease the privacy budget by a factor proportional to the square root of the total number of clients in the Laplace mechanism. In addition, several studies have combined secure multi-party computation with DP in FL [119, 120], and the study in [121] applies HE along with DP. By implementing other techniques to protect privacy, the noise can be decreased, leading to better accuracy.

Overall, the existing frameworks for DPFL still face challenges related to accuracy degradation and significant communication overhead, particularly when striving to maintain equivalent levels of privacy protection. To address these issues, this thesis proposes two novel frameworks aimed at enhancing utility without compromising privacy. Additionally, a privacy loss evaluation methodology is introduced to optimize differential privacy (DP) noise and clipping settings for better privacy auditing. Furthermore, as most current DPFL frameworks with privacy analysis are centralized, this thesis identifies the need for and develops a decentralized DPFL framework with integrated privacy analysis, offering greater flexibility for real-world applications.

2.3 Summary

This chapter presents an overview of DPFL with the fundamental concepts and formulas. It begins with an introduction to the principles of DL and FL. Next, a reconstruction attack method is presented, followed by a detailed background on DP, including its mechanisms, categorizations, and properties. Moreover, the methodologies of DP-based learning, with a focus on the clipping technique, are covered. Finally, a literature review of FL, reconstruction attack and DPFL is presented to explore the existing frameworks and their issues. The fundamental knowledge provided in this chapter is crucial for understanding the proposed frameworks and theoretical results discussed in the following chapters.

Chapter 3

Improving Communication Efficiency and Accelerating Convergence in DPFL

In this chapter, two different frameworks of DPFL to improve the utility are proposed. The first proposed framework focuses on reducing communication costs by reducing the size of the uploaded gradients, while the second one focuses on speeding up the convergence and reducing the communication rounds. Both frameworks improve accuracy performance and save communication costs compared with plain DPFL.

3.1 Introduction

With the enormous amount of data generated by the IoT, AI has been broadly developed and deployed in recent years in many sectors, including finance, industries, network service applications, etc. Since AI 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 GDPR law [10], it has become more difficult to collect raw data to train a good ML model. To solve this issue, Google first proposes FL in the smart keyboard application for typing recommendation [14]. The key idea of FL is to allow local users to train models with their data and upload the gradients, which will 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 model training.

A baseline FL model, FedAvg, is proposed to train local models to reduce the communication rounds between the server and clients [14]. However, when training with non-IID data, FedAvg has unsatisfactory convergence performance [122]. Several local optimizers for FL are proposed to improve the FL convergence performance with non-IID data and are proven to converge much faster than FedAvg [39, 40].

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 [48, 89]. In many FL settings, the servers are considered to be semi-honest. To be specific, they finish the FL tasks honestly, 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, HE can be used by each client to encrypt the gradients before they are uploaded to the server [52]. 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 IoT devices.

Therefore, as a mathematical tool, DP is widely adopted in FL by perturbing original gradients before uploading to the server to provide a strong privacy guarantee [55], also known as LDP. Although DP is a strong privacy protection tool, perturbing gradients inevitably decrease the convergence performance, causing longer training time and lower accuracy. To solve this issue, some studies have proposed adaptive DP frameworks to improve accuracy [108–114]. However, their works are achieved by reducing the required noise. Thus, improving the convergence performance while preserving the same DP noise settings still needs to be researched.

In addition, frequent gradient exchanges between clients and servers are required to achieve an acceptable accuracy performance of FL, bringing expensive communication costs. This issue is more serious in DPFL since the noise brings more divergence on the global model, leading to longer training time. Therefore, reducing the communication costs in DPFL is also worth researching. In general, there are two main research directions for reducing communication costs. The first one is to reduce the communication overhead in each round by reducing the size of the transmitted gradients, which can be achieved by quantification or sparsification. The second direction is to design different optimizers to reduce the convergence time, which can reduce the communication rounds. Even though some research has considered using sparsification in sample-level LDP-FL to save communication costs [123], there is a lack of work to save communication costs for user-level LDP-FL frameworks.

Therefore, in this chapter, two different frameworks are proposed to solve the aforementioned issues. The first framework is a novel Privacy-Preserving FL (PPFL) framework, which achieves both LDP and CDP by adding noise and keeps track of the privacy loss by the MA scheme. Furthermore, to improve the performance, sparse gradients are applied to the local gradients before uploading, and MGD on both the server side and client side is implemented. This framework is introduced in Section 3.2, whose contributions are listed as follows:

- First, the Gaussian mechanism is applied on the client side to achieve (ε, δ)-LDP and CDP is introduced to the LDP-FL to further protect privacy, namely LCDP-FL.
- Second, as the noise is scaled to the l_2-norm of the gradients, the sparse gradients technique is applied to upload only a part of the gradients to the server, which can reduce noise scale and save communication costs.
- Third, MGD is adopted on the clients' side and the servers' side to speed up the training process under the influence of the added noise.
- Finally, extensive simulations with different settings are conducted and the results are compared with other frameworks to show the proposed PPFL's effectiveness.

The second framework is an LDP-based framework and two novel strategies for modifying the local objective function to reduce the convergence time and improve the accuracy performance while maintaining the same privacy protection level, namely Federated Noise Reduction 1&2 (Fed-Nore-1&2). An LDP-based FL framework is first proposed by adding Gaussian noise before uploading their gradients and using RDP to keep track of the privacy loss. This framework is introduced in Section 3.3, whose main contributions are listed as follows:

- The 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 loss. The second proposed strategy is to calculate the additional term due to the noise in the gradients and then incorporate the term into the local objective function. Detailed formulas for deriving modifications to the local objective function for different learning models are provided.
- A Theoretical convergence bound on the first modified local objective function is developed for convex and non-convex settings, which presents the expected increment in the loss function of one round and then the upper convergence bound after multiple rounds.

- A series of simulations of the proposed framework is performed, and the results present that the proposed framework can spare up to 40% training rounds to reach the same performance as plain DPFL under certain settings. Besides, the proposed work also achieves higher accuracy performance compared with other DPFL frameworks.

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

In this section, the first proposed DPFL framework, along with the simulations, is introduced.

3.2.1 The Proposed Differential Privacy-based Federated Learning Framework

In this subsection, the LCDP-FL framework is proposed to enhance privacy protection. To reduce the total communication overhead and improve the overall performance, only a part of the gradients are sent to the central server. Besides, in this framework's setting, MGD is used to train the local models to speed up training, as well as on the central server to help stabilize the training process under the effect of DP noise.

3.2.1.1 Local Differential Privacy and the Combination of Differential Privacy Techniques

In most existing DPFL frameworks, central servers are assumed to be honest and not try to infer sensitive information so they only adopt CDP. 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 [57]. 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 client side should be implemented on top of CDP to further protect clients' data privacy. To enhance privacy protection, in the proposed method, the Gaussian Mechanism is used on each client to achieve (ε, δ)-DP.

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 the LDP settings, the local data of each client is referred to as a small subset of the entire data set involved in FL. To keep track of the accumulative privacy loss, MA [93] is used for a tighter bound of the privacy loss. Based on MA, the privacy loss is related to a proportion q of the batch in the whole dataset, ε, δ, the noise scale and communication rounds. As the noise is added to every client's gradients separately, it has $q = \frac{1}{M}$, where M is the total number of clients. Besides, the total privacy loss 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 i-th client sent to the server are:

$$\nabla\overline{W_i^t} = W^{t-1} - W_i^t + N(0, S_{LDP_i}^2 \sigma_{LDP}^2), \tag{3.1}$$

where S_{LDP_i} is the sensitivity for LDP for the i-th client, W^{t-1} is the model in the previous round, W_i^t is the i-th client's model in this round, N denotes a zero-mean Gaussian distribution for the LDP noise, and σ_{LDP} denotes the base noise scale. Besides, S_{LDP_i} needs to be chosen properly so that it can protect privacy without seriously hindering the model's performance too much [93]. Therefore, to protect each single data point in the training stage, S_{LDP_i} is computed as:

$$S_{LDP_i} = \frac{||\nabla W_i^t||_2}{n_i} = \frac{||W^{t-1} - W_i^t||_2}{n_i}, \tag{3.2}$$

where n_i is the number of data in the i-th client.

As mentioned in Chapter 2, the gradients are usually clipped before aggregating each data sample and adding noise to alleviate the influence of each data sample. In this section, it is assumed that all clients are honest and have similar computing capabilities, leading to that their gradients have a similar scale and sensitivity. Hence, there is no need to limit each sample's influence on the global model so that the process of clipping in the proposed LDP is omitted, but the sensitivity is still used to control the scale of the noise.

After the central server receives all the clients' gradients, it clips all the received gradients as (2.27) and assigns each client a weight for their contribution. In this framework, the weight is $\frac{n_i}{m}$, where m is the number of the total data samples of every client involved in this round. The server aggregates weighted gradients and adds Gaussian noise to them to hide each client's contribution as introduced in [57]. Finally, the aggregated gradients with the combination of two differential privacy implementations are used to compute a new global model W^t in the t-th communication round as follows:

$$\nabla W^t = \sum_{i \in M^t} \frac{n_i}{m} \nabla\overline{W_i^t} + N(0, S_{CDP}^2 \sigma_{CDP}^2), \tag{3.3}$$

$$W^t = W^{t-1} - \nabla W^t, \tag{3.4}$$

$$S_{CDP} = median\{||\nabla W_i^{t+1}||_2\}_{i\in M^t}, \tag{3.5}$$

where $\nabla \overline{W_i^t}$ is the gradients of the i-th client in the t-th communication round with DP noise, S_{CDP} is the sensitivity in CDP, and σ_{CDP} is the scale for CDP noise.

To calculate the privacy loss of the proposed combination of two DPs in this framework, two separate accountants for LDP and CDP are used. In this way, whichever accountant runs out of budget, the FL stops.

3.2.1.2 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, it is impossible to have a massive number of clients joining the FL in some cases. To improve the performance, as the noise is scaled to the l_2-norm of the gradients, by only sending a part of the gradients, the l_2-norm value is smaller. This means that DP noise can have less effect on gradients while still preserving data privacy at the same level. Besides, as the FL is usually performed on smart devices that have limited power and communication resources, sparsifying the gradients can save a large proportion of communication costs.

Furthermore, the method of sparsifying gradients is considered. 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 loss. Therefore, in this scheme, a percentage of gradients with the largest absolute values are maintained, 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.

3.2.1.3 Applying Momentum Gradient Descent 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 stabilize training steps, MGD is not only used during local training to speed up gradient descent but also applied to 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. 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 ($V_{\nabla W^{t-1}}$) 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:

$$V_{\nabla W^t} = \beta_1 * V_{\nabla W^{t-1}} + (1-\beta_1) * \nabla W^t, \tag{3.6}$$

$$W^t = W^{t-1} - \gamma * V_{\nabla W^t}, \tag{3.7}$$

where $V_{\nabla W^0} = 0$, β_1 is a preset constant, γ is the learning rate, and ∇W^t is calculated in (3.3).

3.2.1.4 The Overview of the Proposed Framework

Algorithm 3.1 outlines the proposed LCDP-FL framework, and the implementation of sparse gradients and Global-MGD. At the beginning of the algorithm, a global model and two privacy accountants for LDP and CDP, respectively, are first initialized. For the t-th communication round, accountants check whether the privacy loss exceeds the budget. If not, the server chooses a set M^t of clients and sends them the current global model. If any budget remains, each client in M^t performs local training, computes the sparse gradients, adds LDP noise on the gradients, and sends the noisy gradients 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 which is broadcast for future training. A diagram of the proposed work is shown in Fig. 3.1

3.2.2 Simulation Results

In this subsection, a set of simulations is conducted to show the proposed framework's performance on the MNIST dataset, which is a hand-written digit image dataset and consists of 60,000 training images and 10,000 test images. Besides, this subsection also compares the framework to some classical schemes, called vanilla-FL, which is non-private FedAvg using MGD in local training [14, 82]. The training dataset is divided into shards, where each shard contains data with the same label, and each client is assigned two shards. The proposed framework runs with 100 and 1,000 FL clients, while for 1000 clients, the

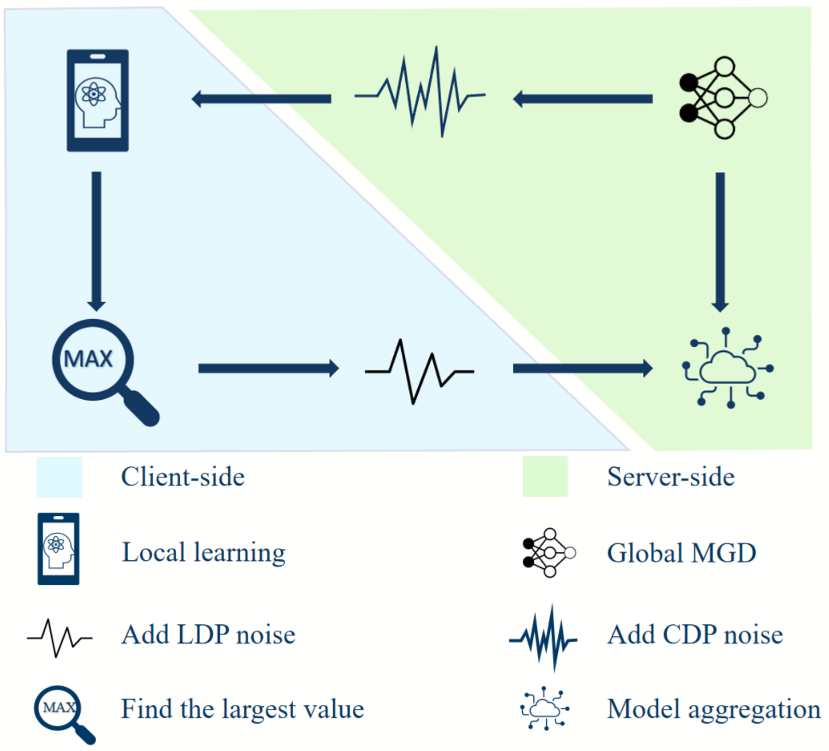


Figure 3.1: The diagram of the proposed LCDP-FL framework.

MNIST is repeated for ten times, leading to a dataset with 600,000 training data. The LDP privacy budget is set as $\varepsilon = 0.5$ and $\delta = 1e-6$, and the CDP privacy budget is set to the same as in [57], where $\varepsilon = 0.5$, $\delta = 1e-3$ for 100 clients and $\delta = 1e-5$ for 1,000 clients. For each client, the local training model is an MLP, which consists of two hidden layers with 200 units per layer and the ReLU activation is employed for each layer except the output layer. Each local client performs ten epochs of MGD per communication round. The LCDP-FL stops when the privacy budget runs out.

The proposed PPFL is compared with some well-known algorithms, DP-SGD [93], LDPFL [59] and CDP-FL [57] in terms of accuracy in Table 3.1, 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 the proposed schemes are implemented with Global-MGD ($\beta = 0.5$), and they update only 10% of the gradients with the largest absolute values.

Algorithm 3.1 LCDP-FL with sparse gradients and MGD

```
1: procedure CENTRAL SERVER
2:     Initialize a global model $W_0$ and privacy accountants for server, $PA_{server}$, and clients, $PA_{clients}$, $V_{\nabla W_0} = 0$
3:     for communication round $t = 0, 1, 2...T$ do
4:         if $\delta_{server} \leq PB_{server}$ or $\delta_{clients} \leq PB_{clients}$ then return current model
5:         end if
6:         Choose a random set of $m$ clients as $M^t$
7:         for Client $i$ in $M^t$ do
8:             $\nabla W_i^{t+1}$, $||\nabla W_i^{t+1}||_2 \leftarrow$ ClientDP($i$,$W^t$)
9:         end for
10:        $S_{CDP} = median\{||\nabla W_i^{t+1}||_2\}_{i \in M^t}$
11:        Clip gradients
12:        $\nabla W^{t+1} = \sum_{i=1}^{M^t} \frac{n_i}{m} \nabla W_i^t$
13:        $V_{\nabla W^{t+1}} = \beta * V_{\nabla W^t} + (1-\beta) * \nabla W^{t+1}$
14:        $W^{t+1} = W^t - (V_{\nabla W^{t+1}} + N(0, S_{CDP}^2 \sigma_{CDP}^2))$
15:    end for
16: end procedure
17: procedure CLIENTDP($i$,$W^t$)
18:    $W_i^t \leftarrow E$ epochs of MGD
19:    $\nabla W_i^{t+1} = W^t - W_i^t$
20:    $\nabla W_i^{t+1} \leftarrow$ the largest top-rate% absolute values of the $\nabla W_i^{t+1}$
21:    $\nabla W_i^{t+1} = \nabla W_i^{t+1} + N(0, S_{LDP_i}^2 \sigma_{LDP}^2)$
22:    return $\nabla W_i^t$,$||\nabla W_i^{t+1}||_2$
23: end procedure
```

3.2.2.1 Evaluation of the Proposed Local Differential Privacy-only Framework

This section first investigates the proposed LDP-FL's performance and the impact of the sparse gradients and MDG at the server's side on accuracy performance. For the proposed LDP-FL, 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. 3.2 shows the results for the vanilla and the proposed FL framework with 100 clients. It is shown that although the 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 added noise. To be specific, the final accuracy of the FL with these techniques outperforms the others and reaches 96.31%. As shown in Table 3.1, the proposed LDP-FL framework has a higher accuracy performance than the one in [59], while the proposed framework has a more general and easily achieved privacy settings by adding δ in DP. For 1,000 clients, the

Table 3.1: Maximum accuracy comparison between the proposed LDP-FL and LCDP-FL and other well-known DP-based learning.

Algorithm	CR	TC	CSR	Acc	DP
DP-SGD [93]	700	1	1	97%	(8,1e-5)-DP
The proposed LDP-FL	52	100	0.5	**96.3%**	(0.5,1e-6)-DP
The proposed LDP-FL	63	1000	0.22	**97%**	(0.5,1e-6)-DP
LDPFL [59]	10	100	1	95.36%	(0.5)-DP
CDP-FL [57]	11	100	0.5	78%	(8,1e-3)-DP
CDP-FL [57]	54	1000	0.22	92%	(8,1e-5)-DP
The proposed LCDP-FL	11	100	0.5	**80.24%**	(8,1e-3)-CDP & (8,1.07e-107)-LDP
The proposed LCDP-FL	54	1000	0.22	**94.3%**	(8,1e-5)-CDP & (8,4e-111)-LDP

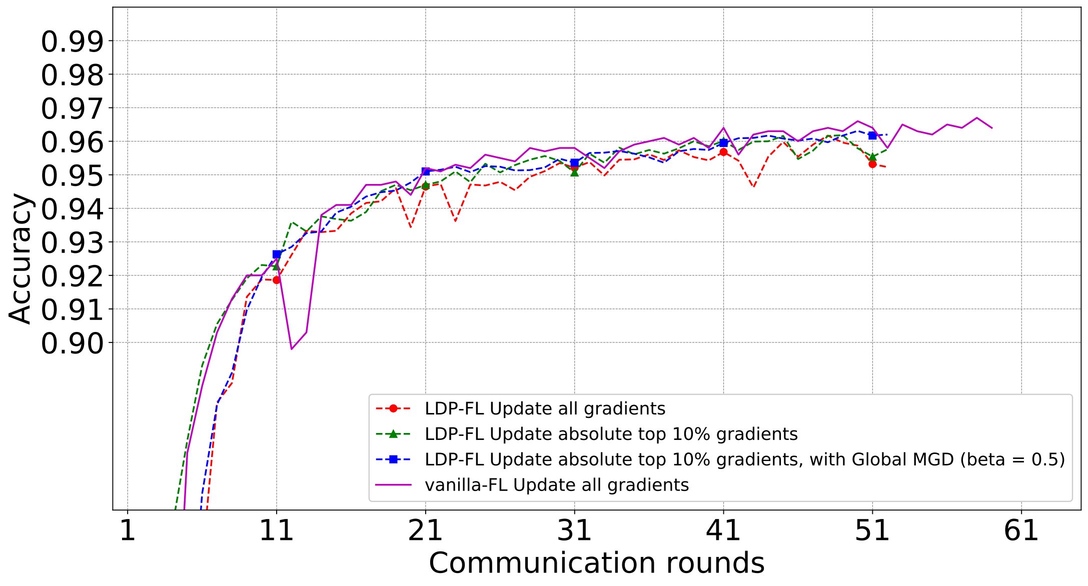


Figure 3.2: Accuracy of vanilla-FL and the proposed LDP-FL with 100 clients.

proposed framework outperforms the plain LDP-FL (without sparse gradients and Central-MGD) and obtains almost the same accuracy of 97%, compared with vanilla FL, as shown in Fig. 3.3.

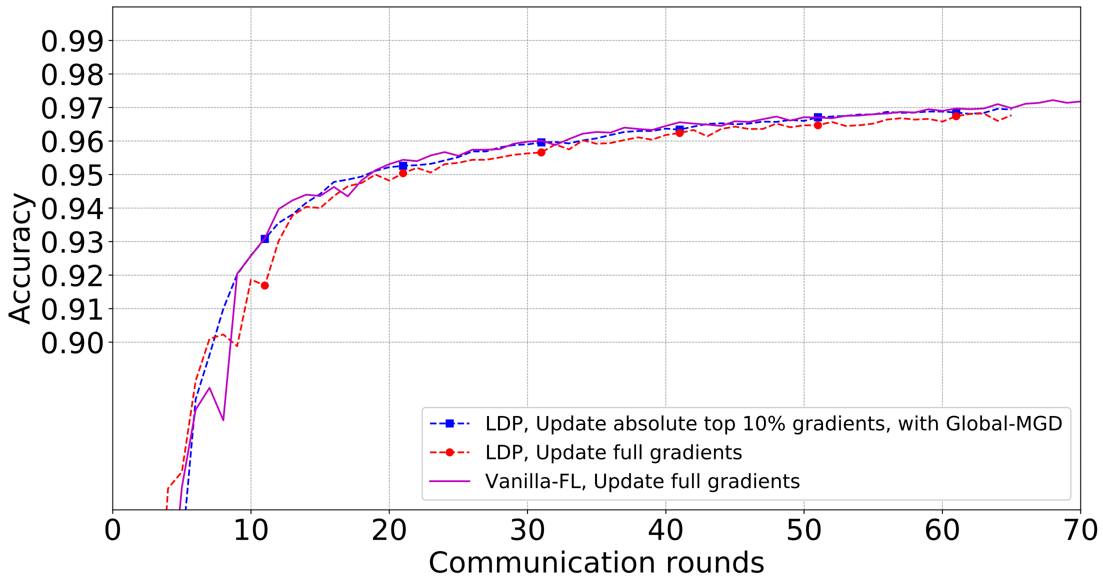


Figure 3.3: Accuracy of vanilla-FL and the proposed LDP-FL with 1000 clients.

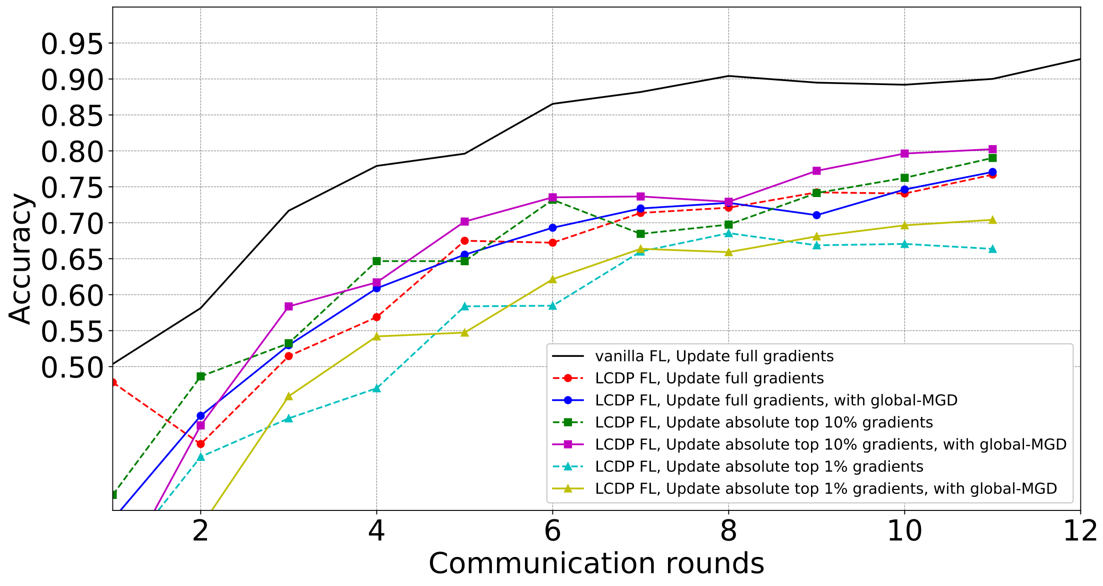


Figure 3.4: Accuracy of the proposed LCDP-FL with 100 clients.

3.2.2.2 Evaluation of the Proposed Combined Differential Privacy Framework

The performance of the proposed LCDP-FL is studied. In the LCDP-FL, $\varepsilon_{LDP} = 8$. 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 to 0.0025, decays by 0.78 for the first ten rounds and is fixed at 0.000267 after ten rounds since the experiments

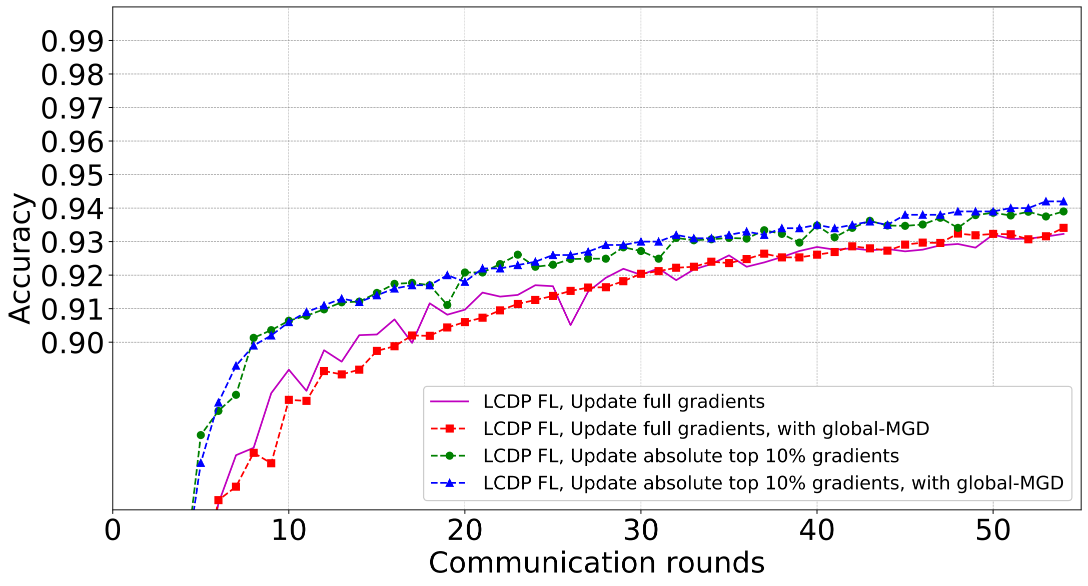


Figure 3.5: Accuracy of the proposed LCDP-FL with 1000 clients.

show that for learning rates larger than 0.0025, the DP noise destroys the training while having a smaller one will make the training process too slow. In addition, when training with 1,000 clients, the initial learning rate is set to 0.1, decays by 0.78 for the first ten rounds and is fixed at 0.0107. The noise scale for LDP is set as $\sigma = 2$, while for CDP, 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 CDP-FL [57]. For the PPFL with LCDP with 100 clients, Fig. 3.4 shows that the proposed method achieves the highest accuracy performance, 80.24%, among all the LCDP-FL. At the same time, it can reduce 90% of the total communication costs. As shown in the Fig. 3.4, it has much worse accuracy than that of the vanilla FL, which suggests that a more delicate CDP mechanism may be needed. Meanwhile, It outperforms the CDP-FL [57] with 100 clients, as presented in Table 3.1. Besides, the performance of only updating 1% of the gradients is also introduced, which has the worst performance, as only very little useful information is updated for the center server each round.

Furthermore, Fig. 3.5 shows that the proposed LCDP-FL with 1,000 clients also reaches the highest accuracy performance, 94.2%, which also outperforms the one for the CDP-only FL framework in [57], as shown in Table 3.1. However, when adopting CDP, the LCDP-FL has slightly worse accuracy than LDP-FL, caused by the small learning rate value.

3.2.2.3 Discussion of Privacy Loss

The privacy loss of the LCDP-FL is discussed, where two accountants for LDP and CDP are used separately. However, according to the MA calculation in [93], and that CDP has a much larger q than LDP, CDP has relatively higher loss than the one of LDP. Therefore, the FL always stops when the CDP accountant exceeds the privacy budget. Through MA, the difference between the privacy loss of LDP and CDP is very large, as shown in Table 3.1. Even though as small privacy loss for each client as possible is desired, the FL controlled only by CDP's accountant can train for very few rounds, resulting in poor performance. In this case, the privacy protection for LDP is more important, and with LDP, CDP can be achieved to a certain degree [100]. Thus, the CDP privacy budget can be loosened in further research to achieve better performance.

3.3 Faster Convergence on Differential Privacy-based Federated Learning

In this section, the second proposed DPFL framework is introduced, along with a theoretical analysis and simulations.

3.3.1 The Proposed Federated Noise Reduction Framework

In this subsection, an LDP-FL scheme is proposed by adding Gaussian noise. Based on the scheme, two modifications on the local cost function $F(W)$ are introduced to improve the convergence under noise. The first modification can work on universal models, while the second one slightly varies for different DL models. In this section, the CNN and the DNN are used as the training models along with the two modifications, and the corresponding derivations of the modifications on the local objective function are provided.

3.3.1.1 Threat Model and Problem Formulation

Even though in FL, only gradients are uploaded, and the data is stored locally, useful information can still be recovered. In this section, the reconstruction attack is considered as the threat model, where it is assumed that the server is semi-honest. To be specific, they execute the FL honestly, but they try to infer the private data from the transmitted data. Therefore, this section aims to prevent the server from obtaining the real data while reducing the accuracy performance degradation caused by the noise.

3.3.1.2 The Plain Differential Privacy-based Federated Learning Model

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

$$\nabla W_i^t = \nabla W_i^t / max(1, \frac{||\nabla W_i^t||_2}{C_i}), \tag{3.8}$$

so that it can eliminate their effect on average value to make them close to the global gradients, where C_i is the clipping bound of the i-th client. C_i is set to the median value of the l_2-norm gradients of the i-th clients across the training. The DP noise is generated as $N(0, S_i^2\sigma^2)$, where S_i is the sensitivity of the i-th client. The σ is a preset base noise variance and S_i is computed as follows:

$$S_i = \frac{C_i}{n_i} = \frac{median||\nabla W_i^t||_2}{n_i} = \frac{median||W^t - W_i^t||_2}{n_i}, \tag{3.9}$$

where n_i is the size of the involved data samples in the i-th client, and S_i is calculated for each layer separately and by taking the median value of all unclipped gradients in each client. Following that, the noise is added to the local gradients as

$$\nabla\overline{W_i^t} = \nabla W_i^t + N(0, S_i^2\sigma^2), \tag{3.10}$$

where $\nabla\overline{W_i^t}$ is the clipped noisy gradients. Finally, the new global model can be aggregated as follows:

$$\overline{W^{t+1}} = W^t - \sum_{i \in m^t} \frac{1}{m} \nabla\overline{W_i^t}. \tag{3.11}$$

To track the privacy loss, this framework uses RDP [75] to calculate the final privacy loss (ε, α) with a fixed δ and the total training rounds. In the 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 inadequate clients for training. Since the clients are chosen randomly for every round, the client drop-out pattern satisfies a uniform distribution, which will not lead to an unbalanced FL model.

3.3.1.3 The Proposed Modifications on Local Objective Function

Two different modifications are proposed to improve the convergence performance under DP noise while maintaining the same protection level, namely as Fed-nore-1&2.

In the proposed framework, a modification term related to the noise is added to the local objective function, which 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, the difference J_{nore_i} between the noisy and the original gradients is considered, which is computed as:

$$\begin{aligned} J_{nore\_i} &= ||\nabla W_i^t - \nabla \overline{W_i^t}||_2 \\ &= ||\nabla W_i^t - (\nabla W_i^t + N(0, S_i^2\sigma^2))||_2 \\ &= ||N(0, S_i^2\sigma^2)||_2, \end{aligned} \tag{3.12}$$

where J_{nore_i} can be simplified to $S_i\sigma$. 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. The formal implementation of Fed-nore-1 is given, where the difference term is added to the local objective function h as follows:

$$\arg\min_{W_i^t} h(W_i^t; W^t) = F(W_i^t) + \lambda_{nore1} \cdot S_i\sigma, \tag{3.13}$$

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, it is assumed that 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. A normal DNN with ReLU as the activation function for hidden layers and the Softmax as the activation function for the output layer is first considered. 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 shown in (2.6)-(2.10).

The gradients are used to update the model. Based on the (2.6), to calculate the expected change in the final loss, the proposed framework needs to calculate the expected change on each layer's $d\bar{z}$. According to (2.7)-(2.10) during the backpropagation, dz_k are used to obtain dz_{k-1} (only when $k > 1$), dw_k and db_k. Therefore, if the backpropagation is reversed, $d\overline{z_k}$ 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 5.

Lemma 5

$$dz_k = dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g'_{act_k}(z_{k-1}), \tag{3.14}$$

$$d\overline{w_k} = N(w_k), \tag{3.15}$$

$$d\overline{b_k} = N(b_k), \tag{3.16}$$

$$d\overline{z_k} = N(w_k) \cdot a_{k-1} + N(b_k) + w_k \cdot d\overline{z_{k-1}} \times g'_{act_k}(z_{k-1}), \tag{3.17}$$

where $N(w_k), N(b_k)$ are the noisy gradients of the layer k, the (3.14) describes the procedure of reversing the original gradients (without noise), and (3.17) is the expression of the expected change on the noisy gradients.

The proof of the Lemma 5 is presented in Appendix A.

Then, the Theorem 3 of calculating the expected change on the final loss is proposed.

Theorem 3 *If DP is applied 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 Softmax as the output layer's activation function, the expected change to the loss is scaled to itself, which is obtained by dividing (3.14) by (3.17) and the noise generation method discussed in Section 3.3.1.2. Finally, the expected change on the loss is simplified to:*

$$d\overline{a_K} = \frac{(a_K - Y) \cdot \sigma}{\sqrt{n_i}}. \tag{3.18}$$

The proof of the Theorem 3 is presented in Appendix B.

With Theorem 3, Fed-nore-2 is proposed. To improve the accuracy performance, the proposed framework assumes that by incorporating the expected change term into the original loss function, the new gradients with noise addition can reach the same performance as the original ones. Since this framework focuses on the modification term as one term related to the parameters, the term of Y is discarded in the final notation. 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:

$$\arg\min_{W_i^t} h(W_i^t; W^t) = -ln(a_K^p * (1 - \lambda_{nore2}(\frac{\sigma}{\sqrt{n_i}})), \tag{3.19}$$

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

Furthermore, this section considers a CNN model with several convolutional layers followed by max pooling (the hyper-parameters of these layers do not affect the results) and a final output layer. The output layer uses Softmax as the activation function while ReLU is used for convolutional layers. Similar to DNN, the backpropagation needs to be reversed. For the CNN model, the gradients of the fully connected layer are the same as those of the hidden layer in the DNN. With regards to the convolution layer and pooling layer, the gradients are obtained as follows:

$$dz_{k-1} = dz_k^{conv} * rot180(a_k) \times g_k'(z_{k-1}), \tag{3.20}$$

$$dw_k = dz_k^{conv} * a_{k-1}, \tag{3.21}$$

$$dz_{k-1} = upsample(dz_k^{pool}), \tag{3.22}$$

where dz_k^{conv} is the gradients of the k-th convolutional layer, $rot180$ is to rotate the a_k by 180 degree, dz_k^{pool} is the gradients of the k-th pooling layer, 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. Based on (3.20)-(3.22), this subsection proposes Corollary 1 to compute the expected change in the loss due to the noise on the convolution layers.

Corollary 1 *If DP is applied through the Gaussian mechanism on a regular CNN model, there is an expected change in the final loss. With Theorem 3, the expected change on the convolution layer is computed as follows:*

$$\frac{d\overline{z_k}}{dz_k} = \frac{\sigma}{\sqrt{n_i}}. \tag{3.23}$$

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 3 and Corollary 1, the expected change in the loss of CNN can be formalized as the same with (3.18). And the Fed-nore-2 on CNN is formalized as the same with (3.19).

Algorithm 3.2 LDP-FL with Fed-nore-1&2

```
1: procedure SERVER
2:     Generate a global model $W^0$, the number of remaining clients $\overline{M}^0$ and privacy budget for clients $(\varepsilon, \delta)$
3:     for round $t = 0, 1, 2...$ do
4:         $\overline{M}^t \leftarrow$ DP-Client
5:         if $\overline{M}^t \leq m$ then
6:             return $W^t$
7:         end if
8:         Select a list of $m$ clients as $m^t$
9:         for all Client $i$ in $m^t$ do
10:            $\nabla\overline{W_i^t} \leftarrow$ Fed-nore-client($t$,$i$,$W^t$)
11:        end for
12:        $\overline{W^{t+1}} = W^t - \sum_{i \in m^t} \frac{1}{m} \nabla\overline{W_i^t}$
13:    end for
14: end procedure
15: procedure DP-CLIENT
16:     for every client $i'$ in $M$ do
17:         Calculate its privacy loss based on the number of its participated communication rounds
18:         if the privacy loss $\leq \varepsilon$ then
19:             Client $i'$ drops out the training
20:         end if
21:     end for
22:     return remaining clients $\overline{M}^t$
23: end procedure
24: procedure FED-NORE-CLIENT($t$,$i$,$W^t$)
25:     if Fed-nore-1 then E epochs of
26:         $W_i^t = argmin_{W_i^t} F(W_i^t) + \lambda_{nore1} * S_i \sigma$
27:     end if
28:     if Fed-nore-2 then E epochs of
29:         $W_i^t = argmin_{W_i^t} [-ln(a_K^p * (1 - \lambda_{nore2}(\frac{\sigma}{\sqrt{n_i}})))]$
30:     end if
31:     $\nabla W_i^t = W^t - W_i^t$
32:     Gradients clipping
33:     $\nabla\overline{W_i^t} = \nabla W_i^t + N(0, S_i^2\sigma^2)$
34:     return $\nabla\overline{W_i^t}$
35: end procedure
```

3.3.1.4 The Overview of the Proposed Fed-nore Framework

The proposed framework with DP through Gaussian Mechanism and Fed-nore-1&2 is introduced in Algorithm 3.2. At first, the server initializes the FL training and creates an initial model W^0. In this framework, every client can choose their privacy budget and base noise variance σ on the purpose of personalized privacy protection level. Second, in each round, all the clients check for their remaining privacy budget and drop out of training if it runs out. Third, the server randomly selects m clients from the remaining clients and broadcasts the model to the selected clients. Fourth, 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 (3.13) in Fed-nore-1 or following (3.19) in Fed-nore-2. The clients 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. Finally, 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.

3.3.1.5 Complexity Analysis

The difference in complexity among the proposed frameworks, traditional FL (FedAvg) and traditional DPFL (without Fed-nore), for one local client is discussed. In this work, the time complexity of the proposed algorithm is derived using Big-O notation, which provides an upper bound on the growth rate of the running time as a function of the input size.

First, compared with FedAvg, the major changes in traditional DPFL are gradient clipping and noise computing. As for gradients clipping, the weights need to be clipped element-wise as $\nabla W_i^t = \nabla W_i^t / max(1, \frac{||\nabla W_i^t||_2}{C_i})$ so that the complexity is $O(size(\nabla W_i^t))$. Meanwhile, the C_i is computed as $C_i = median||W^t - W_i^t||_2$, which brings $O(size(\nabla W_i^t))$ time. Second, the noise generates as $N(0, S_i^2\sigma^2)$, which takes $O(size(\nabla W_i^t))$ time. Third, adding the noise to the gradients as $\nabla\overline{W_i^t} = \nabla W_i^t + N$ will take $O(size(\nabla W_i^t))$ time. In addition, the difference between the proposed work and traditional DPFL is only the modification of the local objective function, which only takes $O(1)$ time.

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

3.3.2 Convergence Analysis

In this subsection, the theoretical convergence guarantee for the proposed framework is presented, which analyzes the expectation of the decrease in the loss function and the convergence bound for the models. For the Fed-nore-2, as the expectation of the effect of the noise on the loss is computed and eliminated during optimizing, the convergence bound of Fed-nore-2 is expected to be the same with FL without noise. Therefore, this subsection only focuses on the convergence bound for the Fed-nore-1.

For the derivation, Assumption 1 is first provided for the proposed framework:

Assumption 1 *(a) $F_i(W)$ is $\beta-$Lipschitz, implying that $||\nabla F_i(W)|| \leq \beta$;*

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

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

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

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

Lemma 6 *The expected decrease of the loss for the global loss function in one round is given as follows:*

$$\begin{aligned}\mathbb{E}\{F(\overline{W}^{t+1})-F(\overline{W}^{t})\} \leq (\frac{\rho l_1^2}{2}-l_1)||\nabla F||^2\\ +(1-l_1\rho)||\nabla F||\mathbb{E}\{||N||\}+\frac{\rho}{2}\mathbb{E}\{||N||^2\},\end{aligned} \tag{3.24}$$

where

$$l_1=\frac{1}{1+\frac{\lambda\sigma\sqrt{n}}{S}}. \tag{3.25}$$

The proof of Lemma 6 is presented in Appendix D.

Then, this framework assumes the noise generated in all the rounds shares the same bound value since they are generated in the same and independent way, and the $F(W)$ is convex. By using Lemma 6 and Assumption 1, the convergence of Fed-nore-1 is upper bounded after T communication rounds by:

Theorem 4 *After T communication rounds of FL training with noise and Fed-nore-1 as local loss function, the convergence upper bound of the proposed framework is presented as:*

$$\begin{aligned}\mathbb{E}\{F(\overline{W}^{t+1})-F(W^*)\} \leq l_3^T(F(W^0)-F(W^*))\\ +(\beta q_1(1-l_1\rho)+\frac{\rho}{2}q_1^2)*\frac{(1-l_3^T)}{1-l_3},\end{aligned} \tag{3.26}$$

where

$$l_3 = (\mu\rho l_1^2 - 2l_1\mu + 1), \tag{3.27}$$

$$q_1 = \frac{\sigma}{\sqrt{n}}(\lambda * \sigma n^{\frac{3}{2}} - \beta). \tag{3.28}$$

The proof of Theorem 4 is presented in Appendix E.

In addition, it is considered that $F(W)$ is non-convex, which brings the following convergence analysis:

Theorem 5 *If $F(W)$ is non-convex and ρ-Lipschitz smooth, this framework can have the following bound after T communication rounds of FL:*

$$\mathbb{E}\{||\nabla F||^2\} \leq \frac{F(W^0) - F(W^*)}{(l_1 - \frac{\rho l_1^2}{2}) * T} + \frac{\beta q_1(1 - l_1\rho) + \frac{\rho q_1^2}{2}}{(l_1 - \frac{\rho l_1^2}{2})}. \tag{3.29}$$

The proof of Theorem 5 is shown in Appendix F.

3.3.3 Simulation Results

In this subsection, to validate the convergence performance of the proposed frameworks, multiple simulations of both Fed-nore-1&2 are performed mostly with the MNIST (a dataset hand-written number image with 60000 training data and 10000 testing data) [38]. In this subsection, the FL with 100 simulated clients is performed, 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, Fed-nore-1&2 with different initial learning rates and different λ values is deployed, where all the learning rate will decay by 0.96 for the first 20 round and be fixed after that. The initial learning rate is set to 0.1 in most cases unless otherwise specified. For the DP mechanism, $\delta = 1e-6$ is used for all the simulations. An RDP-based privacy analysis framework [124] is used to keep tracking privacy loss and calculate the final privacy parameters.

The Fed-nore is studied on two models. The first is an 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 feedforward and back-propagation 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 ten epochs. Besides, due to the randomness of the noise generation and training, the data in all the figures is the smoothed averages of multiple simulations of the same hyper-parameters. To show the effectiveness of the Fed-nore, the results of the plain DPFL as a baseline model are provided, which is the original DPFL model without the improvements. The training curve after 52 rounds is truncated, where the increase in accuracy is marginal.

3.3.3.1 Evaluation of the Proposed Framework with Multi-Layer Perceptron

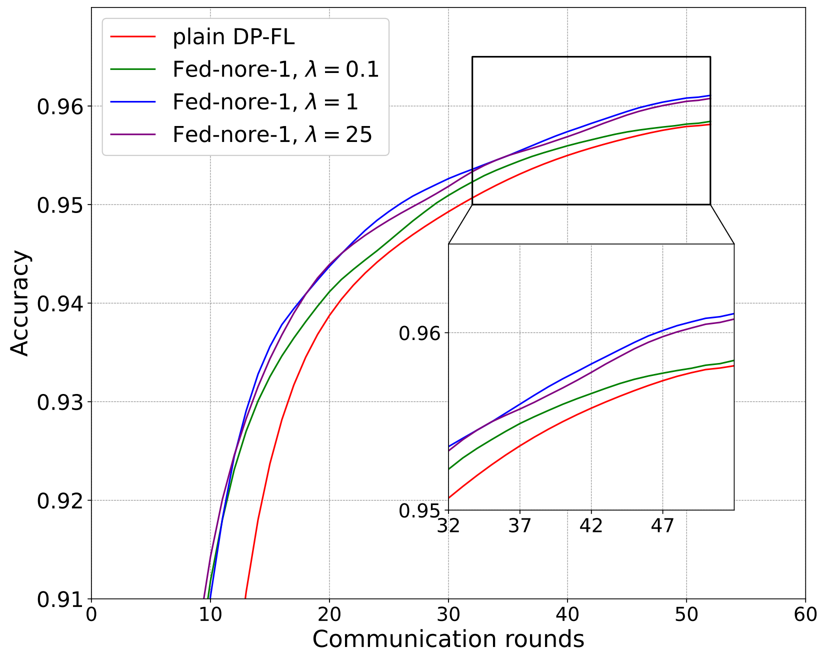


Figure 3.6: Accuracy of the Fed-nore-1 on DNN with different scaling factors (λ) compared with the plain DPFL.

The performance of Fed-nore-1 on the MLP is first presented, where Fed-nore-1 with different λ_{nore1} values $(0.1, 1, 25)$ is evaluated. The results of the Fed-nore-1 are compared with the plain DPFL under the same hyper-parameters settings, where the plain DPFL has an accuracy performance of 95.8% around 50-th round. The results in Fig. 3.6 show that when the λ_{nore1} value is larger than zero, the proposed framework can improve the accuracy performance under DP-noise. In addition, when the λ_{nore1} is one, its accuracy performance reaches the highest, 96.2%. However, when λ_{nore1} is getting larger, the performance is the same with the plain DPFL and even worse.

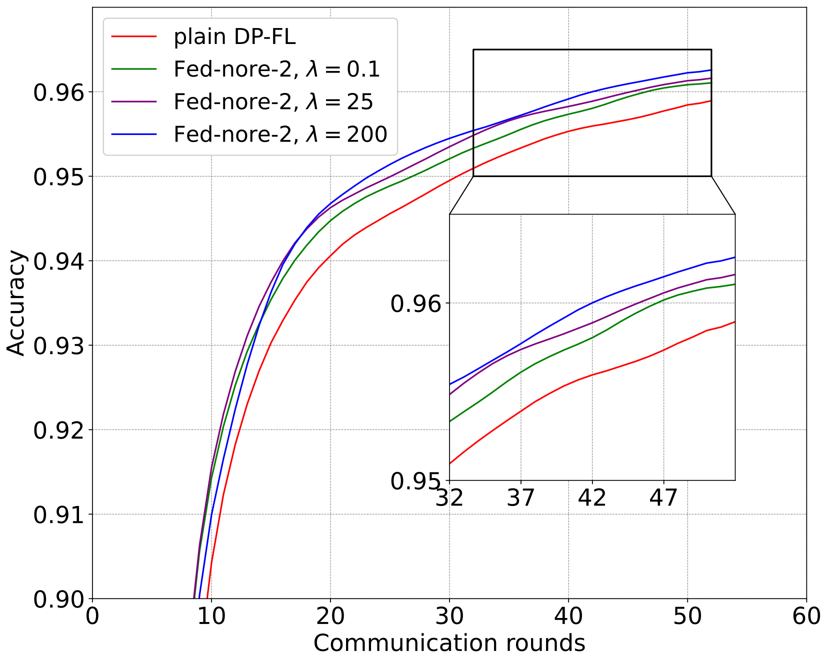


Figure 3.7: Accuracy of the Fed-nore-2 on DNN with different scaling factors (λ) compared with the plain DPFL.

The performance of the Fed-nore-2 on the DNN is also evaluated where the model is tested with λ_{nore2} in $(0.1, 25, 200)$. As shown in Fig. 3.7, the accuracy performance of Fed-nore-2 outperforms the plain DPFL when the selected λ_{nore2} is larger than zero. 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 the Fed-nore-2 has the best accuracy performance of 96.3% when λ_{nore2} is 200. Meanwhile, it can reach 95.8% in the 38-th round, which means that the proposed Fed-nore-2 can save up to 30% of the communication and computation costs compared with the plain DPFL and 5% compared with 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 IoT devices having limited bandwidth, computational capability and power [31, 125], the Fed-nore-2 can converge faster and reach an acceptable accuracy performance while saving a huge amount of communication and computation costs.

Meanwhile, Fed-nore-1&2 with different base noise variance are evaluated to show their robustness by choosing σ in the range of $(4, 6, 10, 12, 16, 24, 40)$. After calculating the communication round using RDP for corresponding σ, the number of communication rounds exceeds 52 when the noise is larger than eight. For these settings, only the first

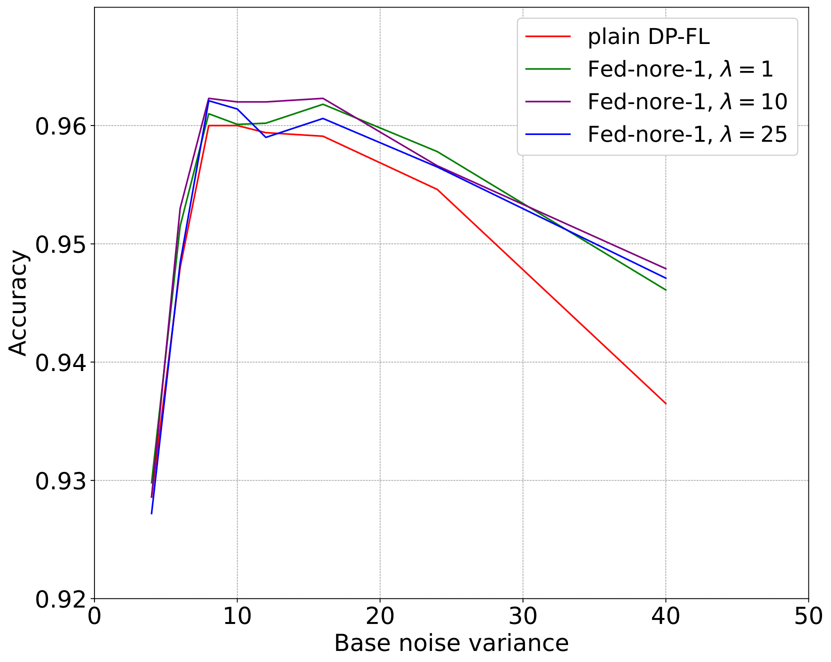


Figure 3.8: Accuracy of the Fed-nore-1 on DNN with different base noise variance and scaling factors (λ) compared with the plain DPFL.

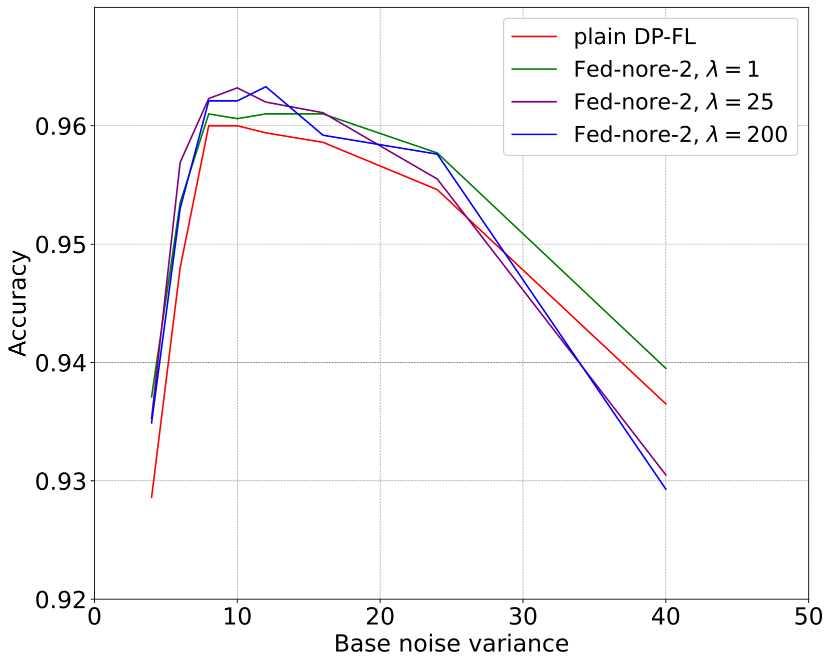


Figure 3.9: Accuracy of the Fed-nore-2 on DNN with different base noise variance and scaling factors (λ) compared with the plain DPFL.

52 rounds of the training results are presented for comparison with the previous results

(where the base noise variance is eight). When the base noise variance is smaller than eight, the communication round for FL is much smaller, leading to bad accuracy. It is shown in Fig. 3.8 that under a small noise, the Fed-nore-1 can not improve the accuracy performance. Besides, it is shown that with a larger base noise variance and an optimal scaling factor, Fed-nore-1 can perform much better than the plain DPFL. For Fed-nore-2, it is shown in Fig. 3.9 that the proposed Fed-nore-2 can greatly improve the accuracy performance compared with the plain DPFL when the base noise variance is smaller than 24. 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 DPFL.

3.3.3.2 Evaluation of the Proposed Framework with Convolutional Neural Network

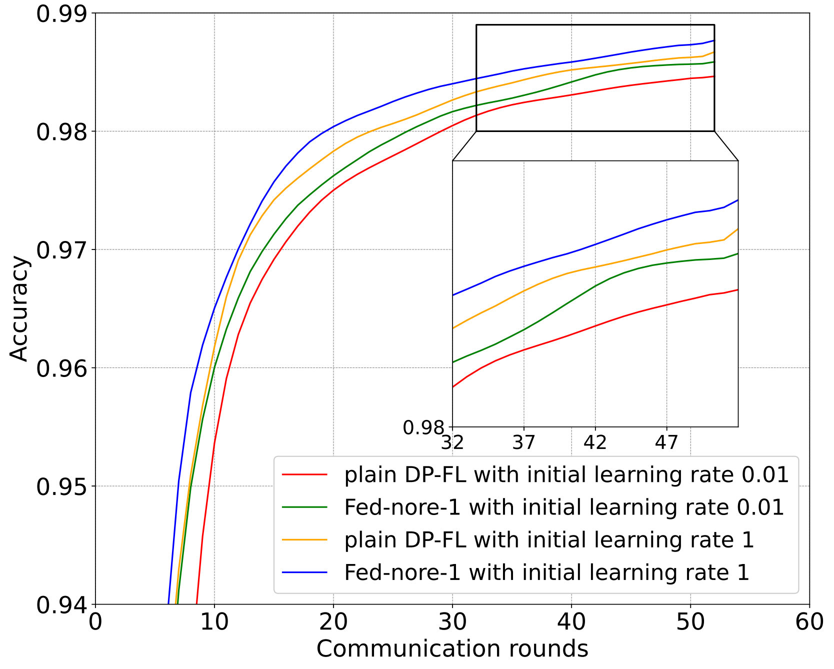


Figure 3.10: Accuracy of the Fed-nore-1 on CNN with a scaling factor of ten and different learning rates compared with the plain DPFL.

The performance of the Fed-nore-1&2 on the 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), the Fed-nore-1&2 with different learning rates are conducted to show their performance. In this simulation, the learning rate is chosen from $(0.01, 0.1, 1)$. The results for the Fed-nore-1 on CNN with the learning rates of $(0.01, 1)$ are first presented. As shown in Fig. 3.10, Fed-nore-1 can slightly improve the convergence performance and accuracy performance for CNN with all the initial learning rates when the scaling factor

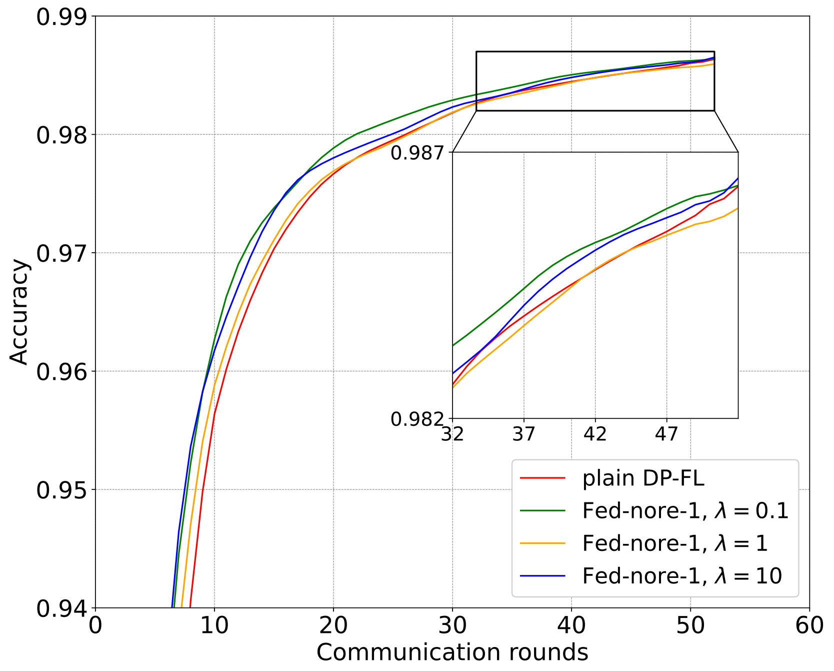


Figure 3.11: Accuracy of the Fed-nore-1 with different scaling factors (λ) compared with the plain DPFL.

is set to ten. In addition, the simulation with an initial learning rate of 0.1 and different scaling factors is performed to further test its effectiveness. It is shown in Fig. 3.11 that the 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, Fed-nore-2 with an initial learning rate of 0.1 and different scaling factors are evaluated. As shown in Fig. 3.12, the 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 ten, the improvement is the best, and it can save 40% of communication rounds to achieve the same results with the plain DPFL. The Fed-nore-2 with different initial learning rates is also tested. The results in Fig. 3.13 show that the 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 is larger than 0.1, the Fed-nore-2 performs worse than the plain DPFL.

In addition, simulations with CIFAR10 on the same CNN model are also performed. The data is assigned to 100 clients in the same way as MNIST. The best accuracy of the proposed frameworks and plain DPFL within 50 communication rounds with different settings is presented in Table 3.2, which shows that the proposed framework can increase the test accuracy from 48.7% for plain DPFL to 52.0% for Fed-nore-1 and to 53.7% for

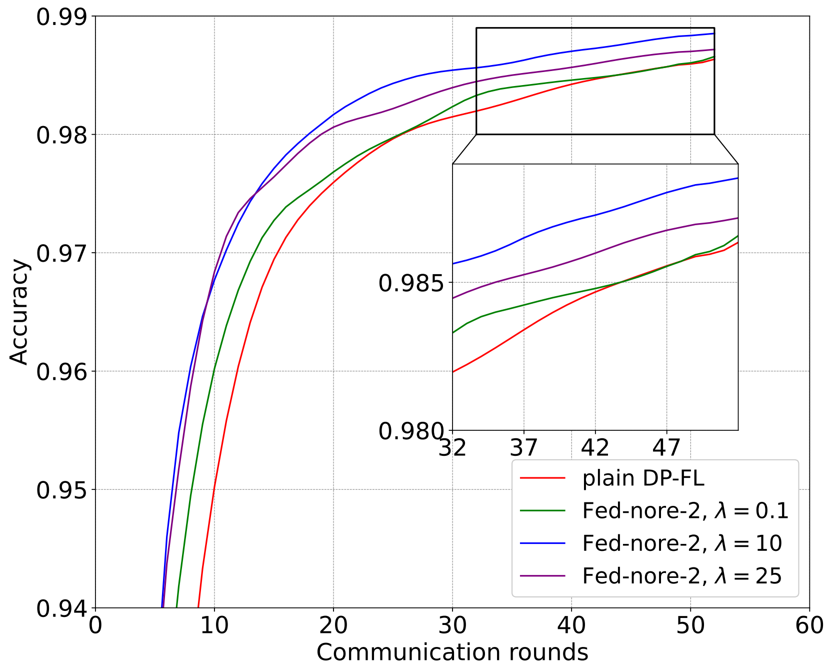


Figure 3.12: Accuracy of the Fed-nore-2 with different scaling factors (λ) compared with the plain DPFL.

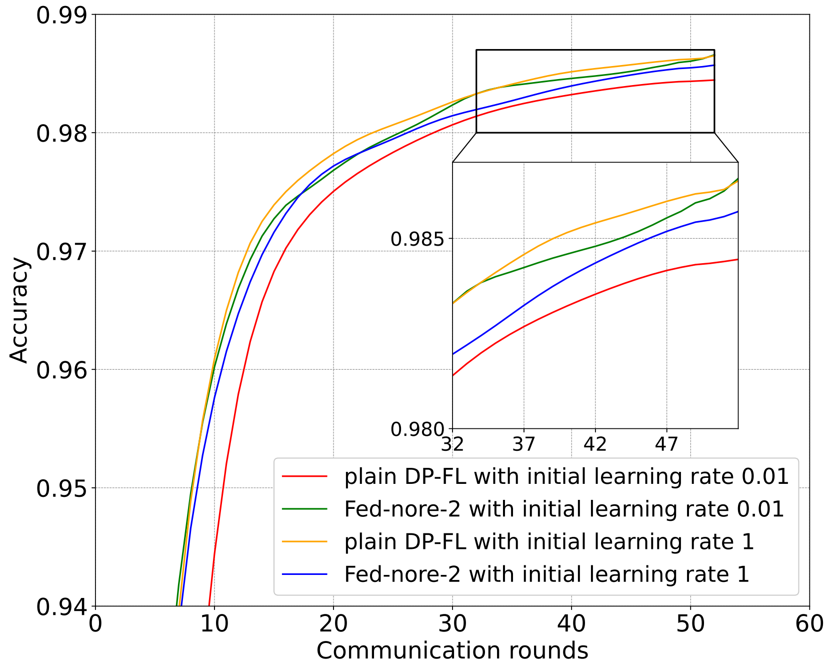


Figure 3.13: Accuracy of the Fed-nore-2 with a scaling factor of ten and different learning rates compared with the plain DPFL.

Fed-nore-2. Meanwhile, both Fed-nore-1&2 reach the highest accuracy with at least 10% fewer communication rounds than plain DPFL. Therefore, the proposed Fed-nore-1&2

show a greater accuracy improvement on more complicated dataset.

Moreover, two strategies are implemented on CNN with CIFAR10 at the same time with an initial learning rate of 0.1, base noise variance of eight and several scaling factors. However, the accuracy performance is worse than using only one, as shown in Table 3.2. A possible reason for this phenomenon is that they may interfere with each other during local training.

Table 3.2: Accuracy comparison among the plain DPFL and the proposed frameworks on CIFAR10.

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 DPFL	0	48.7%

3.3.3.3 Performance Comparison and Discussion

The performance of the proposed framework is compared with the plain DPFL (the DPFL framework without the Fed-nore-1&2 modification), FedAvg, and some well-known DP-based learning in Table 3.3, 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. The best accuracy performance results of the proposed framework are provided, where the results in the blankets in the column R show the communication round to reach the same accuracy performance (the blankets in the column ACC) with the plain DPFL. Meanwhile, unlike the average results in all figures, Table 3.3 contains the best results under the optimal settings, where the base noise variance is eight. It is shown in Table 3.3 that the proposed Fed-nore-1 can slightly improve the accuracy performance for the MLP 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, it is found 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 has much better performance when the value of the noise variance is moderate.

Table 3.3: Maximum accuracy comparison among other FL, DP-SGD, the plain DPFL and the proposed framework on MNIST.

Framework	R	SR	ACC	DP budget
FedAvg	380	1	97%	
DP-SGD(MLP) [93]	700		97%	(8,1e-5)-DP
LDP-FL(CNN) [59]	10	1	95.36%	(0.5)-DP
Plain DPFL(MLP)	52	0.5	95.9%	(0.27,63)-RDP
Plain DPFL(CNN)	52	0.5	98.7%	(0.27,63)-RDP
Fed-nore-1(MLP)	52(42)	0.5	96.4%(96%)	(0.27,63)-RDP
Fed-nore-2(MLP)	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

3.4 Summary

In this chapter, two DPFL frameworks are proposed to improve accuracy and reduce communication costs.

First of all, a new DP-based framework for PPFL by adding sparse gradients and Global-MGD is proposed in order to improve its convergence performance. To be specific, an LDP framework is proposed based on the Gaussian Mechanism and MA, which also implements CDP to enhance privacy protection. This framework also proposes a new scheme to calculate the DP noise scale for the LDP-FL. Experimental results show that for MNIST, the proposed FL framework achieves better performance than other DP-based FL. Besides, the framework can also reduce massive communication costs using sparse gradients.

Secondly, a novel FL framework is proposed to enhance convergence performance in FL in terms of accuracy and time consumption. An LDP-FL scheme is first designed by adding Gaussian noise on the local gradients before uploading to satisfy RDP. Second, to improve its performance, two modifications on the local objective function and detailed derivations are proposed 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 local loss due to the noise. By integrating the expected change into 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, multiple simulations on DNN and CNN with the corresponding modification are conducted to show the effectiveness of the proposed frameworks. For both CNN and DNN, the results show that, compared with the original DPFL, the Fed-nore-1&2 can both increase the accuracy performance and greatly reduce 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 DPFL under optimal settings. On the other hand, when using CNN as the training model, Fed-nore-2 has a higher accuracy performance than Fed-nore-1 for most scenarios. Besides, the accuracy improvement of CIFAR10 is greater than that of MNIST. Furthermore, it is found that Fed-nore-1 works better with a relatively larger noise, while Fed-nore-2 works better when the noise is moderate.

Chapter 4

Evaluating Privacy Loss in Differential Privacy-based Federated Learning

4.1 Introduction

Even though FL has a great promise to protect data privacy by transmitting only gradients, several studies have shown that FL is still vulnerable to privacy leakage [93, 126]. Specifically, the servers perform the FL process honestly, but they may still be curious about the original data. This curiosity can lead to reconstruction attacks, where sensitive information can be recovered from the received gradients [47, 48, 89, 90]. The authors in [89, 90] have successfully reconstructed the real images on a small batch size of training data, where they generate some dummy images, which are fed into the FL models to compute dummy gradients. The dummy images are updated with an L-BFGS by minimizing the difference between the dummy gradients and true gradients. In addition, the authors in [47, 48] have shown that the true labels of the images can be recovered.

To enhance the privacy protection of FL, HE and DP can be adopted. For the HE application in FL [52], the local gradients are encrypted by the local clients before transmitting to the server. The server aggregates these encrypted gradients to create an encrypted global model, which is dispatched to local users for decryption and further training. However, a malicious client may collude with the server and share the HE key with the server to decrypt the gradients. Thus, a reliable key-sharing scheme is required. In addition, due to the large size of gradients in effective DL models, encrypting the gradients is computationally expensive.

On the other hand, DP is used to enhance FL's privacy protection by adding artificial noise on the gradients [57, 59]. The noise can be either added to the local gradients to protect data privacy [61] or on the aggregated gradients to protect identity-level privacy [57]. Given that privacy loss accumulates with repeated use of the DP mechanism, and FL may require many rounds to achieve acceptable performance, it is essential to develop a method to accurately calculate cumulative privacy loss [93]. Some studies have proposed tighter bounds for calculating privacy loss, enabling smaller privacy loss under the same number of DP mechanism usage. This allows more rounds and smaller noise scale in DPFL to achieve better accuracy before the privacy budget is exhausted [75]. Nevertheless, the accuracy of FL is degraded due to the randomization of the gradients. Thus, it is important to balance the trade-off between privacy protection and accuracy. Moreover, the existing works measure the privacy loss with the theoretical DP parameters, ε and δ, which makes it hard to reflect an intuitive privacy loss in real-world applications. To address this issue, reconstruction attacks may be applied to provide a more intuitive measurement of privacy loss, as they directly show how much useful information can be recovered.

While most existing studies have performed reconstruction attacks on gradients with random noise, they typically apply noise at the sample-level [104]. However, most DP-based FL methods apply noise following UDP [55]. Additionally, no study has directly examined the relationship between DP settings and reconstruction attacks. With the rapid growth of data and the increasing use of smart applications, a more practical privacy loss scheme is needed, allowing users to select their desired level of privacy protection based on their specific applications. Furthermore, although many studies have explored the clipping effect on learning performance [127, 128], more detailed research is needed on how clipping affects privacy leakage.

As such, s methodology is designed to evaluate the privacy leakage in DPFL by conducting reconstruction attacks on DPFL's gradients. A new metric is proposed to quantify the practical privacy leakage under different DP settings based on the reconstruction attacks. This chapter empirically shows the accumulative privacy loss under two different reconstruction attack settings (the sum of all rounds' gradients and each round's gradients) and proposes several findings on DPFL. The contributions of this chapter are listed as follows:

- Reconstruction attacks are conducted on gradients modified with DP, namely noisy gradients, to demonstrate the effectiveness of DP in protecting privacy. Additionally, this chapter proposes a two-factor metric for measuring privacy leakage based on similarities.

- This chapter investigates and characterizes the relationship between DP mechanisms and privacy leakage through reconstruction attacks. The findings in this chapter demonstrate that implementing an anonymous mechanism for local clients can reduce the probability of data privacy leakage. It analyzes how the gradient clipping term impacts both the level of privacy protection and learning accuracy.

- The convergence bound is derived for two reconstruction strategies with DP noise by computing the expected impact of the DP mechanism on the reconstruction loss.
- Extensive simulations are performed to evaluate the effects of various DP settings on reconstruction attacks and FL training. A cross-analysis between privacy protection and accuracy is conducted to provide a direct trade-off between privacy protection and FL learning performance. The findings in this chapter can be used to enhance FL utility under the DP mechanism and guide personalized privacy protection settings.

4.2 Privacy Loss Evaluating Methodology

In this section, the threat model and the detailed privacy loss evaluating methodology are presented.

4.2.1 Threat Model and Problem Formulation

This chapter only considers protecting privacy from reconstruction attacks, where it assumes all the clients honestly perform local training and updating, and the server honestly performs the aggregating and broadcasting of the global model, while they may be curious about the sensitive information carried in the gradients and try to reconstruct it. This kind of clients and servers is also known as semi-honest adversaries. Even though DP is widely adopted to enhance privacy protection in FL, the accuracy is degraded. Therefore, in the following subsections, a method for evaluating the privacy loss of DPFL is proposed by first inverting gradients to recover the noisy gradients and analyzing the difference between the recovered and true gradients. The proposed evaluation method consists of four focal points. Additionally, a new metric is designed to estimate the privacy loss under different reconstruction settings. The results can be used to evaluate the effect of DP in FL and show how much privacy protection FL needs.

4.2.2 Differential Privacy-based Federated Learning Procedure

This subsection introduces the UDP-based FL framework, where the reconstruction attacks are launched. To begin with, the central server generates an initial model W^0 and

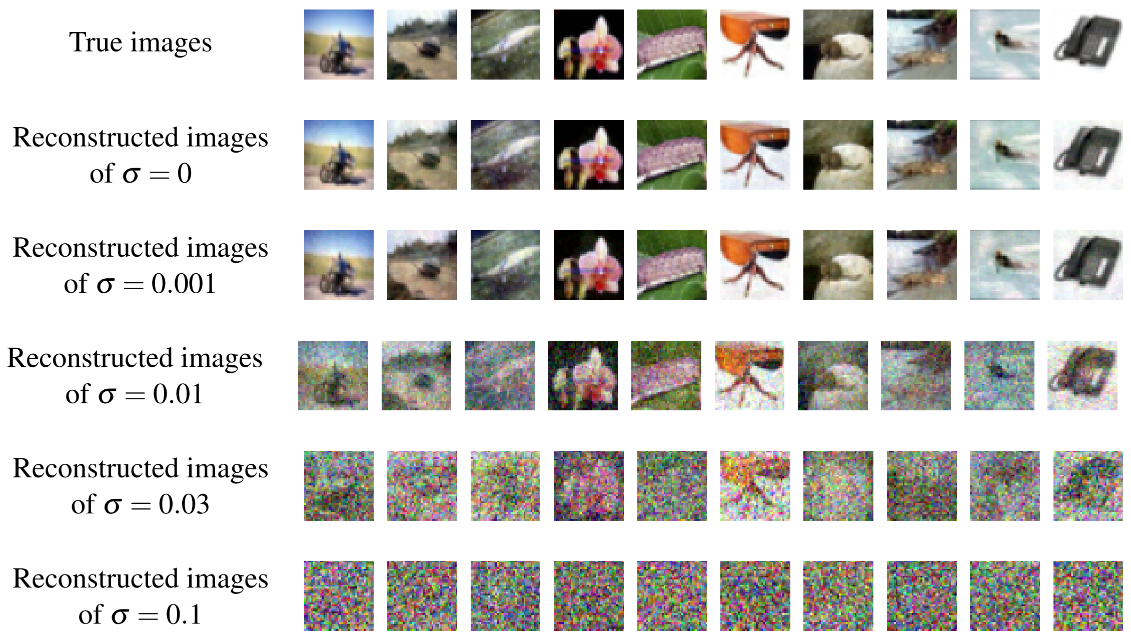

True images
Reconstructed images of $\sigma = 0$
Reconstructed images of $\sigma = 0.001$
Reconstructed images of $\sigma = 0.01$
Reconstructed images of $\sigma = 0.03$
Reconstructed images of $\sigma = 0.1$

Figure 4.1: Reconstructed images with different noise scales compared with the true images.

broadcasts it to m chosen clients from M total clients. Then, each client optimizes the received model with their local data and computes the gradients. In order to meet the UDP guarantee of privacy protection, the local gradients will be clipped as follows:

$$\widehat{g(x^*)} = g(x^*)/max(1, \frac{||g(x^*)||_2}{C}), \tag{4.1}$$

where $\widehat{g(x^*)}$ is the clipped gradients, x^* is the true images, C is the clip bound of the DP mechanism. Two methods of choosing clipping bound are mostly used: the median value of the norm of the gradients over all rounds or a fixed constant. The noise is added to the clipped gradients as follows:

$$\overline{g(x^*)} = \widehat{g(x^*)} + N(0, \sigma^2(\frac{C}{n})^2), \tag{4.2}$$

where n is the total number of data in every client, σ is a preset base noise scale. Next, the clipped noisy gradients $\overline{g(x^*)}$ are sent to the server for aggregation. The server averages all the received gradients to compute the new global model. In this DPFL framework, each local client will compute their own privacy loss with RDP [75], where they will abort training when they run out of their privacy budget (ε is larger than the preset threshold).

The FL training is stopped by the server when a certain amount of clients abort training.

4.2.3 Privacy Leakage Evaluating Main Procedure

This subsection focuses on designing the main procedure for evaluating the privacy protection level of the DP mechanism in FL via reconstruction attack, where the work [48] is used to reconstruct the initial images.

An MLP is first considered for local training, which has three hidden layers, each having 1024 units. The hidden layer uses ReLU activation and the model is optimized by Adam optimizer [129]. Each client has been assigned 100 random images. In this chapter, the CIFAR100 is mostly used to decrease the probability of a reconstruction attack to output the correct label by guessing. The size of batches of each local client in one round is 25.

Table 4.1: The average PSNR, cosine similarity (Cos-sim) and Euclidean distance (Euc-dis) between the reconstructed images and the real images with different base noise scales (σ) and the fixed clipping bound (C).

Parameters	PSNR	Cos-sim	Euc-dis
σ=0.001, C=0.1	21.47	0.97	18.14
σ=0.001, C=0.4	28.52	0.99	8.09
σ=0.001, C=1	28.61	0.99	7.99
σ=0.01, C=0.1	21.63	0.97	17.80
σ=0.01, C=0.4	23.66	0.95	13.89
σ=0.01, C=1	17.84	0.81	26.77
σ=0.1, C=0.1	15.92	0.70	33.51
σ=0.1, C=0.4	10.22	0.14	64.54
σ=0.1, C=1	9.29	0.03	71.96
σ=1, C=0.1	9.12	0.02	73.35
σ=1, C=0.4	8.97	0.01	74.64
σ=1, C=1	8.95	0.00	74.81

One round of local training is first investigated by performing gradient reconstruction attacks with different base noise scales and a fixed clipped bound of one, and the first ten images are presented in Fig. 4.1, which shows that the images are more blurry when the base noise scale increases. To measure the quality of the recovered images, peak signal-to-noise ratio (PSNR), cosine similarity and Euclidean distance are used, which are presented in Table 4.1. From the results, the following observations can be obtained: a) when the total noise scale $\sigma * C$ increases from 0.01 to 0.04, it can be observed that all the metrics change greatly, leading to the quality of reconstructed images drops sharply; b) the value of the clipping bound is not monotonous related to the privacy loss under same noise variance, while most studies claim increasing clipping bound can enhance privacy protection [61, 93]. Specifically, larger clipping bound has a weaker privacy protection level when the base noise variance is smaller since more useful information can be recovered. On the other hand, when the base noise variance increases, larger clipping bounds provides a better privacy protection. These observations first motivate the idea that the choice of C should be further studied to enhance privacy protection, which will be investigated in the remainder of this section. In addition, since a certain scale of privacy can prevent attackers from reconstructing great images, whether larger noise is needed should be studied, which motivates that the trade-off between privacy and accuracy should be more precisely quantified in order to achieve a good overall performance of DPFL.

4.2.4 Privacy Loss Measurement Metric

Even though images are undistinguished by sight when the base noise scale is large, it is worth investigating whether some useful information can still be extracted. Thus, this subsection investigates the privacy leakage in a white-box manner and assumes the reconstruction attacks are launched on the local side. The recovered and original images are fed into five pre-trained models including Resnet56, VGG, RepVGG, Shufflenet and Mobilenet [50, 130–133] and the average cosine similarity and Euclidean distance (normalized to [0,1]) of the average outputs of the five models between the recovered and original images are computed. This subsection proposes to measure a practical privacy loss pl_j of the j-th image based on its weighted deviation of the similarities as follows:

$$\begin{aligned} pl_j = \alpha_2 * (CS_j - avg(CS) - \rho_{cs}) \\ - (1-\alpha_2) * (ED_j - avg(ED) + \rho_{ed})), \end{aligned} \tag{4.3}$$

where α_2 is the weighting coefficient of cosine similarity and Euclidean distance, the CS_j is the cosine similarity of the j-th image, CS is the set of the cosine similarity of the local dataset, ED_j is the Euclidean distance of the j-th image, ED is the set of the Euclidean distance of local dataset, ρ_{cs}, ρ_{ed} are bias factors related to the base noise scale σ and avg is the average operation. ρ_{cs}, ρ_{ed} are set to $\sigma * C$ when the result is smaller than 0.1 and set to 0.1 when it is larger. In this case, when pl_j is larger than zero, it suggests that the j-th reconstructed image is very likely to infer similar results compared with the true image. Based on this result, a practical privacy leakage measuring metric $P(pl)$ for the local user is quantified by counting the number of images with $pl_j > 0$.

4.2.5 Two Reconstruction Manners

To study the privacy loss of multiple rounds of DP-based learning, the reconstruction attack is also performed along with local client training. The local client will train for multiple epochs and reconstruct the gradient in every epoch, where the reconstruction attack on the gradients of every round respectively (one-round-manner) and on the sum of the gradients of all rounds (all-round-manner) is studied.

The privacy leakage is also evaluated in DPFL in both two manners, where the FL will follow the procedure described in subsection 4.2.2. By conducting the privacy loss evaluation in DPFL, it is aimed to study whether the aggregation and average of the gradients will lead to a different outcome for the privacy leakage and also study the trade-off between the accuracy and privacy leakage through the evaluation method.

4.2.6 Clipping Impacts on Privacy Leakage

In addition, the gradient clipping is essential to ensure DP guarantee [128], and the value of the clipping bound needs to be chosen properly since a large clipping bound increases the noise magnitude and a small clipping bound causes the gradients to have less useful information. In some works, the clipping bound is set to a constant value [61, 93], and the noise addition mechanism follows the (4.2). However, some works choose the median l_2 norm values of the unclipped gradients to be its clipping bound in each local client [58, 134]. By implementing two different clipping methods into one local client training, the privacy leakage under the reconstruction attack in two manners is studied in order to provide some insights into choosing clipping bound for a better overall performance in terms of privacy and accuracy.

4.3 Reconstruction Attack Convergence Analysis

In this section, the convergence of the reconstruction attacks under DP mechanisms is analyzed. Before derivation of the convergence, this section provides an assumption as follows:

Assumption 2 *$J(x)$ is $L-$Lipschitz smooth, implying that $||\nabla J(x) - \nabla J(x')|| \leq L||x-x'||$, where L is a positive number.*

It is first derived, for one round of reconstruction attack, how much change on the gradient update is created by the noise addition. To achieve that, the difference between noisy (and clipped) and original gradients is computed to obtain the following lemma.

Lemma 7

$$||\overline{g(x^*)}|| \leq k_1 * ||g(x^*)||, \tag{4.4}$$

where,

$$k_1 = min(1, \frac{C}{||g(x^*)||}) + \frac{\sigma}{\sqrt{n}} * \frac{C}{||g(x^*)||}. \tag{4.5}$$

The proof of Lemma 7 is presented in Appendix G.

Based on Lemma 7, the attack loss on noisy gradients can be expressed as follows:

$$\bar{J} = \arg\min_{x \in (0,1)^n} 1 - \frac{< g(x), g(x^*) >}{k_1 * ||g(x)|| ||g(x^*)||} + \alpha_1 TV(x), \tag{4.6}$$

where $<>$ is the inner product of two matrices, α_1 is a preset constant and TV is the total variation introduced in Section 2.1.3.

If the maximum effect of the noise is considered, which leads to a smaller cosine similarity of the gradients, the maximum value of $||\overline{g(x^*)}||$ is taken. The derivative of noisy reconstruction attacks can be expressed as follows:

$$\overline{\nabla J} = -\frac{1}{k_1}\nabla(\frac{< g(x), g(x^*) >}{||g(x)|| ||g(x^*)||}) + \alpha_1 \nabla TV(x). \tag{4.7}$$

Theorem 6 *It is considered an approximate expected difference in the updates of the attack with noise. As this attack uses signed-Adam to optimize, whether the noise on the transmitted gradients will flip the sign of the gradients of the attack model needs to be considered. However, since learning is a black box, a proper value cannot be directly achieved. Therefore, it is considered that the probability of $-\nabla(\frac{<g(x),g(x^*)>}{||g(x)|| ||g(x^*)||})$ and $\alpha_1 \nabla TV(x)$ have same sign to be s. Besides, if they have different signs, it is considered that the probability that the noise on the gradients can flip the sign is related to the noise. Theoretically, the larger the noise is, the larger the probability of the gradients' sign flipping. Meanwhile, if σ increased, $\frac{1}{k_1}$ decrease. Therefore, it approximates the probability of the sign of the gradients flipping to be $(1-\frac{1}{k_1})$, while an assumption that $\frac{1}{k_1}$ is smaller than one is made. Finally, by taking the expectation of the updates with probability s and $(1-\frac{1}{k_1})$, it can have:*

After passing signed-Adam, the expected difference in the updates of the attack in each training round is estimated as follows:

$$\begin{aligned}\mathbb{E}\{\overline{U}\} &= (s*1+(1-s)*(-1)*(1-\frac{1}{k_1})+(1-s)*1*(\frac{1}{k_1}))U \\ &= (1-2k_2+2k_2 s)U,\end{aligned} \tag{4.8}$$

where,

$$k_2 = \frac{k_1-1}{k_1}, \tag{4.9}$$

and U is the original update of one round of reconstruction attack.

Then, it is derived that after T rounds, the changed update can be bounded by recurrent using Theorem 6 as follows.

Corollary 2 *Each round of updates of the attacks is affected not only by the added noise but also by the changes in the updates of the previous round. Therefore, by considering both factors and recurrent computation, it can achieve the following:*

$$\mathbb{E}\{||\nabla J||^2\} = \frac{L(J(x^0)-J(x^*))}{2Tk_3}, \tag{4.10}$$

where,

$$k_3 = (1-2k_2+2k_2 s). \tag{4.11}$$

The proof of Corollary 2 is shown in Appendix H.

Meanwhile, this section also considers other Euclidean distance-based reconstruction attacks, whose local objective function is expressed as follows:

$$J = \arg\min_{x\in(0,1)^n} ||g(x)-g(x^*)||+Re, \tag{4.12}$$

where *Re* is the regularization term, varying for different algorithms [47,89]. For this local objective function, the attack loss on noisy gradients can be expressed as follows:

Lemma 8

$$\overline{J} = J + \sqrt{2(1-k_1)*g(x)g(x^*)}. \tag{4.13}$$

The proof of Lemma 8 is shown in Appendix I.

While it is assumed that the FL model has a second-order derivative, the derivative of noisy reconstruction attacks can be expressed as follows:

$$\overline{\nabla J} = \nabla J + k_4 g'(x), \tag{4.14}$$

where,

$$k_4 = \sqrt{\frac{2g(x^*)(1-k_1)}{g(x)}}. \tag{4.15}$$

Finally, Theorem 7 is proposed.

Theorem 7 *The convergence of the l_2 distance-based reconstruction attack on noisy gradients is computed as follows:*

$$\mathbb{E}\{||\nabla J||^2\} = \frac{L(J(x^0) - J(x^*))}{2Tk_5^2}, \tag{4.16}$$

where,

$$k_5 = 1 + \frac{k_4}{(\beta_3 \pm 1)}, \tag{4.17}$$

$$\beta_3 g'(x) = \nabla Re. \tag{4.18}$$

The proof of Theorem 7 is presented in Appendix J.

As can be seen in Corollary 2 and Theorem 7, increasing the noise scale and clipping bound may lead to a higher bound on the convergence, and when there is no DP, the convergence bound is the same as that of the noise-free reconstruction attack. The above results can be utilized to compute a theoretical privacy leakage in a black-box manner by comparing it with the original convergence bound (without noise) and to compute a multi-task loss along with the DPFL convergence bound, which can further guide for selecting DP settings and support personalized privacy protection level choosing. Additionally, different orders and multipliers of the noise scale and clipping bound imply that two reconstruction strategies may perform better than each other under different DP and clipping settings.

4.4 Empirical Results

In this section, a series of simulations are performed on the proposed privacy leakage evaluation framework. This section investigates it in two scenarios: a single-end (SE) DL to mainly study how clipping and DP mechanisms perform against reconstruction attacks, and an FL scenario to show the trade-off between accuracy and DP effect. The SE follows the learning procedure similar to a single-user DPFL introduced in Section 4.2.2. The reconstruction attacks algorithm in [47, 48] is adopted.

In this section, simulations are conducted on CIFAR10 and CIFAR100, where the first one contains 60000 32x32 colour images in ten classes, and the second one contains 60000 32x32 colour images in 100 classes, including different animals, tools, etc. Two learning models are used in this section. The first one is the MLP introduced in Section 4.2.3. And the second model is a CNN with six convolutional layers, represented as CNN6, where each convolution layer is followed by a leaky-ReLU with a negative slope of 0.2. The detailed architecture is shown in Table 4.2.

Table 4.2: The CNN6 architecture.

layer index	**Type**	**Input Channels**	**Kernel Size**	**Stride**	**Padding**
layer 1	Conv2d	*channels*	4x4	2	2
layer 2	Conv2d	12	3x3	2	1
layer 3	Conv2d	36	3x3	1	1
layer 4	Conv2d	36	3x3	1	1
layer 5	Conv2d	36	3x3	2	1
layer 6	Conv2d	64	3x3	1	1
layer 7	Flatten	-	-	-	-
layer 8	Linear	3200	-	-	-

In the SE scenario, the client will have 100 data (one of each kind), and the batch size is 25. The client first trains the model for one epoch and computes the noisy gradients. After that, the attacker conducts the reconstruction attack on each epoch's gradients for the two manners (one-round and all-round). Meanwhile, during the simulations, this section uses MLP since the attacker can recover the most visible images from its gradients. In the FL scenario, this section uses the CNN model and CIFAR10 to show the accuracy under DP. The FL has 40 training clients, where each has 100 random local data (4000 training data in total) and a batch size of 20. The size of the test dataset is 1000. Every local client trains the local model for five local epochs before they add DP noise and send it to the server. In every FL round, the attacker conducts an attack on one fixed client's gradients.

As for the learning and attack settings for both scenarios, the attack learning rate, the local learning rate and the TV regularization are all 0.01, and the attacker performs 2500 epochs of the reconstruction attack on the SE/FL's gradients of every epoch for the two manners. Since the trend and comparison of the FL accuracy is mostly studied, this section only performs 15 rounds of FL training and reconstruction for efficiency. In the figures, A_r is the all-round-manner reconstruction, and O_r is the one-round-manner reconstruction. cp is clipping bound and nv is the base noise scale. The uneven curves in all the figures are caused by one-time simulation and random noise. Since this chapter only focuses on the comparison of the trend of the results, the simulations of every setting are conducted only once.

4.4.1 Evaluation of Privacy Leakage in Single-End Learning

This subsection studies the privacy loss under reconstruction attack under two different clipping settings, where the MLP and CIFAR100 are used for the training.

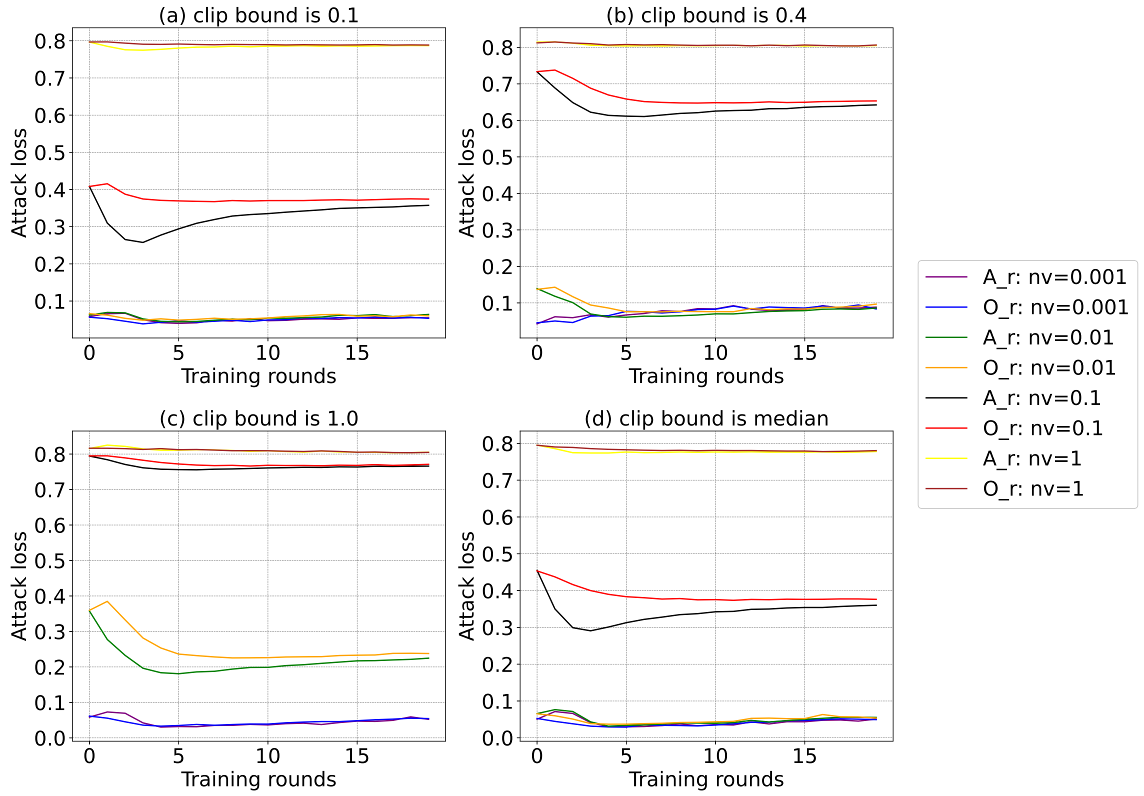


Figure 4.2: Attack loss on SE learning with different base noise scales and the same clipping bound.

It is first shown that, under the same clipping bound, the attack loss of different base noise scales in Fig. 4.2. It is shown that the increase in the base noise scale also increases the attack loss, which generally fits the principle of DP. However, there is an interesting finding that all-round-manner attacks have a smaller attack loss than that of the one-round manner, which means that combining all the received gradients in FL can increase the probability of privacy leakage. Therefore, methods that ensure the receiver cannot tell whom the gradients are coming from can increase privacy protection levels.

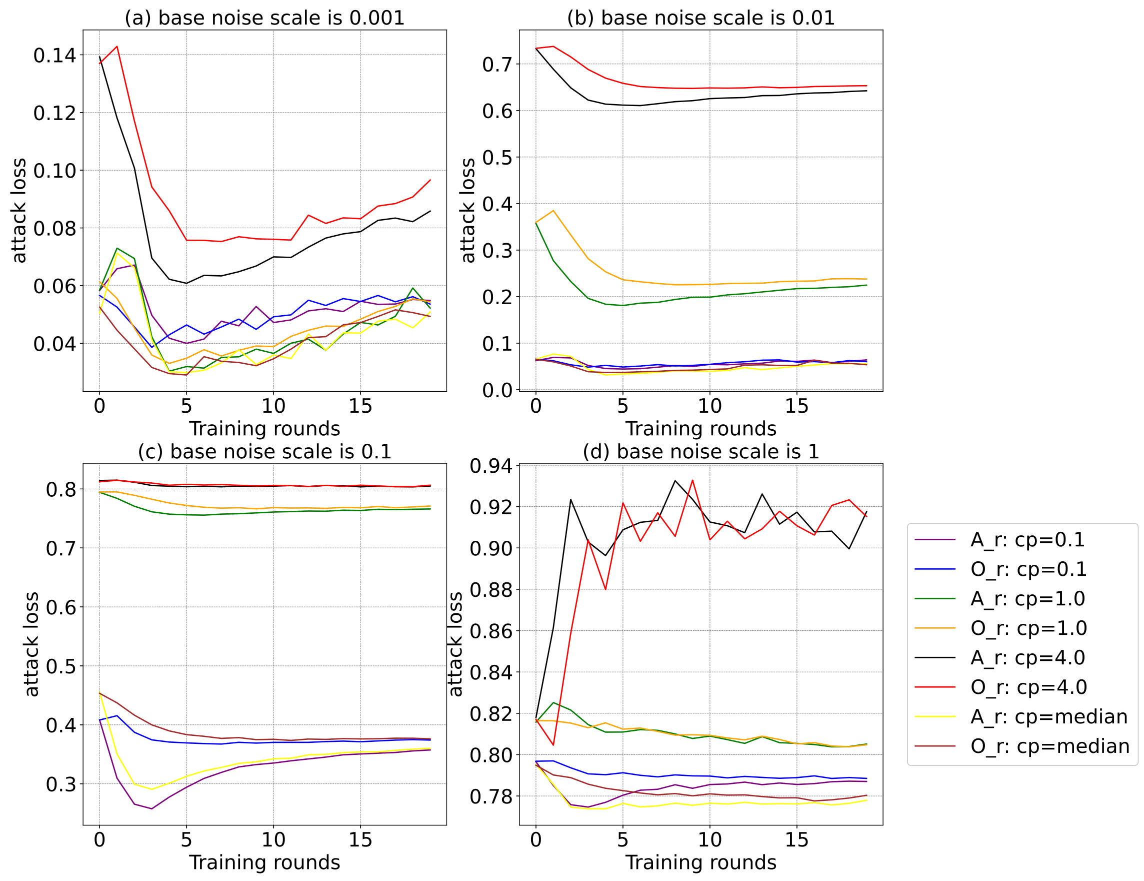


Figure 4.3: Attack loss on SE learning with different clipping bounds and the same base noise scale.

Then, it is present that, under the same base noise scale, the attack loss of different clipping bounds in Fig. 4.3. It shows that as the clipping bound increases, the attack loss decreases since the total noise scale increases, which is similar to the existing literature. Nevertheless, comparing different clipping bounds under the same noise scale cannot fully illustrate the effect of clipping on privacy protection. Meanwhile, the attack loss of the median clipping bound method is similar to a smaller fixed clipping bound. Hence, the median clipping bound may lead to weaker privacy protection under a smaller base noise scale since the total noise scale is not large enough. However, when the base noise scale is large, which can bring a larger total noise scale, the median clipping bound method can be chosen in order to save time when choosing the proper fixed clipping bound.

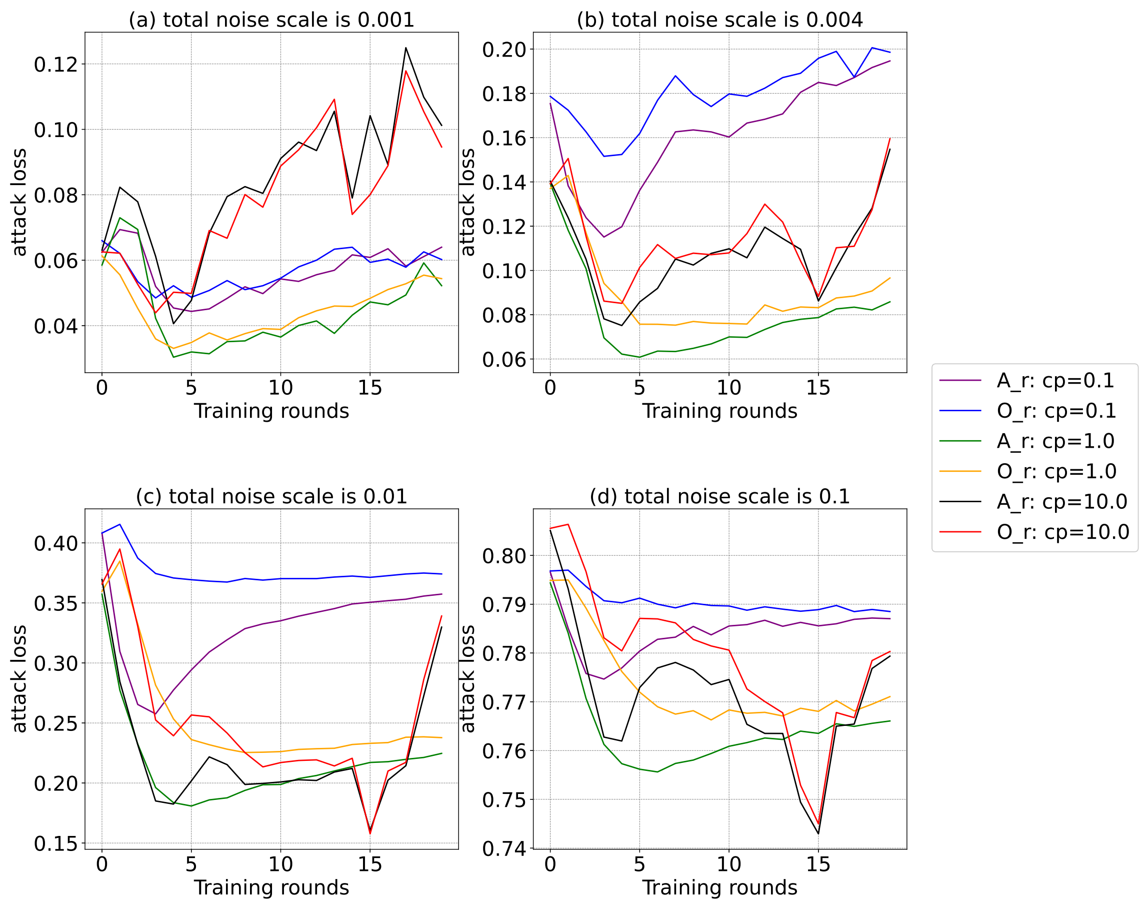


Figure 4.4: Attack loss on SE learning with different clipping bounds and the same total noise scale.

Furthermore, to further study the effect of clipping bound, this subsection further shows the attack loss of different fixed clipping bounds under the same total noise scale, whose results are shown in Fig. 4.4. It is shown that under a relatively small base noise scale, when increasing the clipping bound, the loss first decreases and increases. This is due to that larger clipping allows the gradients to bring more information so that the attacker can recover more information. However, increasing the clipping bound scale will bring a large total noise scale, which is proportional to the base noise scale times clipping bound, and will increase the attack loss as expected.

In addition, other metrics for measuring the effect of reconstruct attack are provided in Table 4.3, which also characterizes how different clipping bounds and noise scales affect privacy protection levels. Specifically, for most settings, when the total noise scale

Table 4.3: Different metrics of attack loss of SE (on CIFAR100 and MLP).

Clipping bound	Noise scale	PSNR	Prob leak	LPIPS
0.1	0.001	20.8497	0.59	0.0322
0.1	0.01	20.7342	0.58	0.0329
0.1	0.04	19.6833	0.54	0.0498
0.1	0.1	16.2349	0.53	0.1052
0.1	1	9.0808	0.24	0.4034
0.4	0.001	25.4005	0.56	0.0253
0.4	0.01	23.6943	0.55	0.0140
0.4	0.1	11.4474	0.4	0.2770
0.4	1	8.8907	0.23	0.4105
1	0.001	26.9651	0.57	0.0079
1	0.01	19.3725	0.52	0.0436
1	0.1	9.2355	0.23	0.3874
1	1	9.1147	0.23	0.4104
4	0.001	23.6943	0.55	0.0143
4	0.01	11.4474	0.4	0.2777
4	0.1	8.8907	0.23	0.4105
4	1	8.5518	0.24	0.3995
10	0.0001	25.4504	0.57	0.0113
10	0.0004	22.9730	0.56	0.0246
10	0.001	20.1703	0.54	0.0439
10	0.01	10.4285	0.26	0.3385
median	0.001	28.3672	0.59	0.0078
median	0.01	26.7709	0.57	0.0080
median	0.1	17.1509	0.53	0.0856
median	1	8.9232	0.23	0.4054

increases, the attack has a degraded performance as it has a smaller probability of leak, larger LPIPS and smaller PSNR. Meanwhile, when the total noise scale is the same, a larger clipping bound leads to a better-reconstructed image quality. For example, when the total noise scale is 0.01, the PSNR is 16.23, 19.37 and 20.17 for the clipping bound of $0.1, 1, 10$, respectively. Besides, two sets of reconstructed images under the same total noise scale of 0.01 with different clipping bounds are shown in Fig. 4.5. They also shows that larger clipping bound under the same total noise scale can weaken privacy protection since the images with the smaller clipping bound are more blurry. Based on the results, this chapter can show insight to choose the optimal value of clipping bound and noise scale for privacy protection.

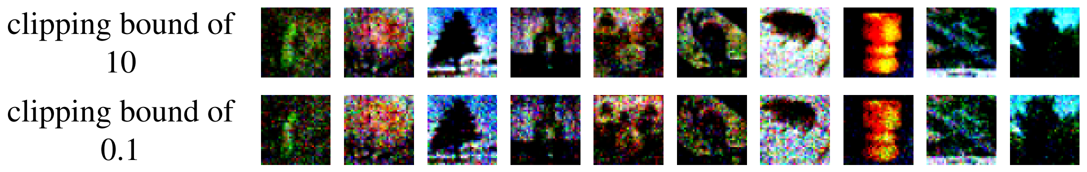


Figure 4.5: Reconstructed images with the same total noise scale.

4.4.2 Evaluation of the Performance of Federated Learning

In this subsection, the trade-off between the FL accuracy and DP settings is studied, where CIFAR10 and CNN6 are used for the local training to show the accuracy and privacy loss characteristics. Besides, the pattern for privacy leakage under two manners of reconstruction attacks is similar to the one in SE learning. Therefore, this subsection only focuses on the all-round-manner.

To begin with, the privacy loss under reconstructed attacks for SE learning and FL are compared. Some metrics are presented to compare the reconstructed images and original images in Fig. 4.6 and Table 4.4, where most metrics show that FL may have a higher probability of privacy leakage than SE learning. Meanwhile, for SE learning, the attack loss has a great increase after a few rounds, while FL has a mild one. The possible reason is that under FL training, the global model may be more generalized, so local training may still generate a relatively large gradient to force the model closer to its personal requirements, while the noise of SE may be too larger to its gradients, making it containing less information. Therefore, as FL training continues, the probability of privacy leakage may still be higher compared with that of SE learning. As multiple DP usage also leads to the increased probability of privacy leakage, reducing the probability of privacy leakage as the training rounds increase should be considered.

Table 4.4: Different metrics of attack loss of SE and FL (on CIFAR100 and MLP).

Model	clipping bound	PSNR	Prob leak	LPIPS
FL	0.1	12.56	0.57	0.1916
	1	12.76	0.54	0.1991
	10	11.00	0.21	0.2332
SE	0.1	12.42	0.55	0.2202
	1	12.40	0.51	0.2224
	10	10.69	0.25	0.2330

This subsection also presents the accuracy under different clipping settings for FL in Fig. 4.7. Since the total noise scale is proportional to the multiplication of the base noise

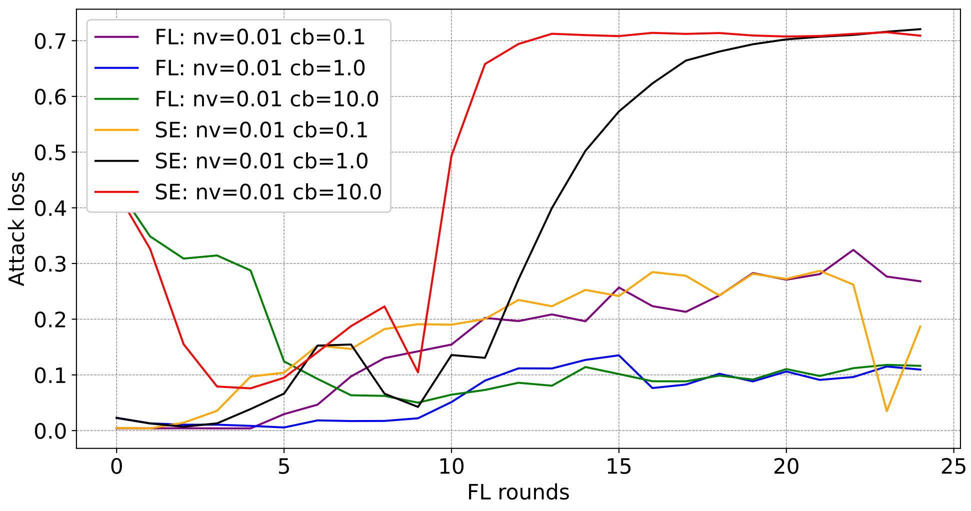


Figure 4.6: Attack loss of SE and FL.

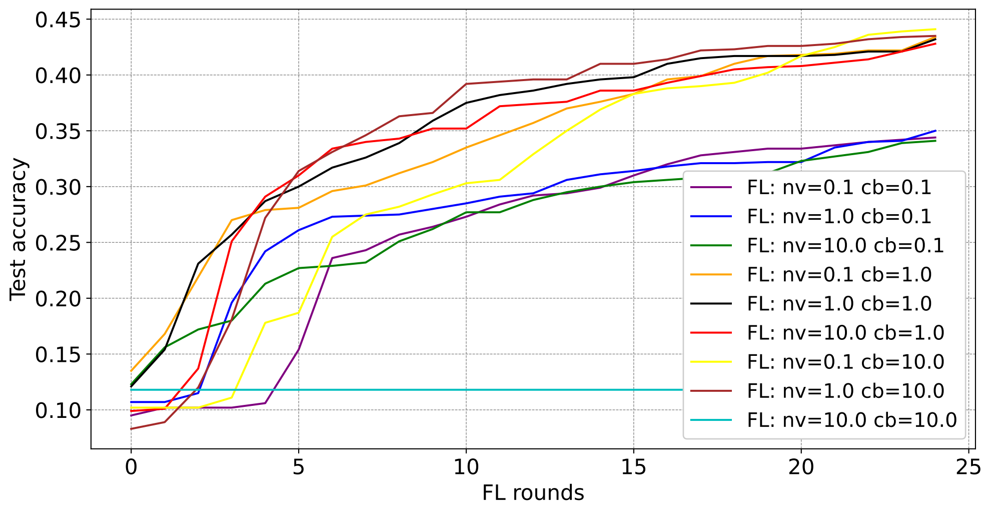


Figure 4.7: FL test accuracy with different DP settings.

scale and clipping bound, the accuracy under the same total noise scale is studied. It is shown in Fig. 4.7 that when the total noise scale is small, increasing the clipping bound can improve the accuracy. For example, when the total noise scale is one, the accuracy increases from 34% to 44% as the clipping bound increases from 0.1 to 10. Meanwhile, when increasing the clipping bound after the noise scale is larger than 10, the FL accuracy can be destroyed due to the large total noise scale. The results may suggest that when comparing different clipping bounds for FL, the total noise scale cannot be neglected, while most existing studies compare the learning and attack results under different clipping bounds and different total noise scales [61], which may be inappropriate.

Finally, the corresponding privacy budget for DPFL is calculated by using RDP accounting with the noise variance and $\delta = 1e-5$. It is observed that when the base noise scale is one, the ε is derived to be eight. Besides, when the base noise scale is smaller than one, the ε is much larger than 100. However, it is shown in Table 4.4 that the noise scale of 0.1 can lead to bad attack results. Additionally, most existing DPFL frameworks choose a ε smaller than ten. The results in this subsection suggest that a smaller privacy budget (larger ε) for DPFL may be needed to provide the required privacy protection level, which can also improve accuracy.

4.4.3 Comparison on Reconstruction Strategies

Table 4.5: Different metrics of attack loss with cosine similarity and Euclidean distance-based reconstruction (on CIFAR100 and MLP).

Reconstruction	C	nvar	PSNR	Prob leak	LPIPS
Euclidean distance	0.1	0.01	17.1171	0.54	0.1146
Cosine similarity	0.1	0.01	20.7342	0.58	0.0329
Euclidean distance	0.1	0.1	16.6834	0.54	0.1164
Cosine similarity	0.1	0.1	16.2349	0.53	0.1052
Euclidean distance	1.0	0.01	20.0328	0.5	0.0829
Cosine similarity	1.0	0.01	19.3725	0.52	0.0436
Euclidean distance	1.0	0.1	12.1396	0.21	0.2525
Cosine similarity	1.0	0.1	9.2355	0.23	0.3874
Euclidean distance	10.0	0.01	12.1396	0.21	0.2525
Cosine similarity	10.0	0.01	10.4285	0.26	0.3385
Euclidean distance	10.0	0.1	7.8497	0.21	0.4113
Cosine similarity	10.0	0.1	7.2322	0.20	0.4675

In this subsection, the comparison of the two most used reconstruction strategies, cosine similarity and Euclidean distance with the same regularized term, is introduced. As shown in Table 4.5, when the total noise scale is smaller than 0.01, the cosine similarity-based reconstruction is able to generate images with better quality than the Euclidean distance-based one. For example, the LPIPS of Euclidean distance-based attack is 0.1146, which is larger than 0.0329, the one of cosine similarity. However, when the total noise increases, the Euclidean distance generates better quality images as the PSNR and probability of leakage is larger and the LPIPS is smaller. In summary, the Euclidean distance-based reconstruction attacks can generate better images when the total noise scale is large, and the cosine similarity-based reconstruction attacks generate better images when it is small. Therefore, combined with methods of approximating the noise scale of the received gradients, this chapter shows how to choose more effective reconstruction attack strategies from the attacker's perspective, which can also promote the selection of more targeted defenses.

4.5 Summary

In this chapter, reconstruction attacks are conducted on DPFL and then based on the reconstructed images, the relations between DP settings, including noise scale and clipping bound choosing, are characterized. To be specific, a privacy loss evaluation method is proposed by using different pre-trained models to infer the output on reconstructed images and true images and compare the distance between them to measure the privacy leakage. To evaluate the privacy loss, extensive simulations are conducted on reconstructed attacks on DP-based learning under different DP settings. Based on the results, this chapter empirically studies the relations among clipping bound, noise scale and privacy protection level for FL and SE learning. Meanwhile, several findings are summarized, which can be utilized to choose better DP settings for FL, including that the anonymous clients mechanism can improve DP protection level by studying two manners of gradients reconstruction (reconstruct from the sum of all gradients and every round gradients) and how clipping bound actually affects privacy protection. Besides, the difference between reconstruction attacks under FL and SE learning is studied. Finally, the difference between the two reconstruction strategies, cosine similarity and Euclidean distance, is compared empirically on privacy protection and their corresponding theoretical convergence bound under noise. With the calculation of the privacy budget by using RDP, the findings in this chapter can be utilized to guide the selection of the proper privacy protection level for DP-based FL.

Chapter 5

Anonymous Differential Privacy-based Decentralized Federated Learning with Performance Analysis

5.1 Introduction

Traditional ML typically requires the centralized storage of all data to train a global model. However, uploading a massive amount of data to a central server not only imposes a huge communication burden but also raises an increasing awareness of privacy protection. Additionally, data collection and processing have been regulated and restricted by data protection regulations such as the EU/UK GDPR [10]. To protect data privacy in ML, an emerging ML training technique called FL has been proposed, where the training tasks are assigned to local devices, and the trained local models are then sent to a central server to form a global model [14]. Since all the data is trained locally at clients' devices and never leaves the local storage, it is expected that FL can protect data privacy. However, traditional FL still suffers from significant challenges, including huge communication, SPOF, and privacy leakage.

Traditional FL relies on a central server to perform model aggregating and broadcasting, which introduces the problem of SPOF and requires the server to be completely trusted. To address these issues, the concept of DFL has been proposed, in which the local clients directly transmit gradients with each other in a peer-to-peer manner, and the gradients aggregation is performed at local clients [135, 136]. For example, the authors in [31,137] propose a gossip-based method for the DFL framework to exchange the model. In addition, the work [135] proposes a voting consensus-based DFL framework by applying a leader-candidate-follower method, while the work [30] proposes a committee-based DFL framework by letting a committee select clients and assign weights based on their models. Except for different topologies, blockchain technology can be adopted in DFL as an effective coordination tool [138, 139]. A blockchain-based DFL can supervise the DFL procedure to ensure that all participants perform honestly and apply a reward mechanism to local clients to encourage them to attend FL [140].

Although DFL and FL are claimed to protect data privacy, studies have demonstrated that reconstruction attacks can be carried out on transmitted local training gradients to recover sensitive information [47, 92, 122]. To achieve the recovery, these reconstruction attacks typically involve synthesizing data to compute gradients, followed by optimization based on either the cosine similarity [92] or Euclidean distance [47] between the synthesized gradients and the transmitted local gradients. To guarantee privacy, some research has implemented an anonymous mechanism in (D)FL in order to protect the identity of local clients [141]. However, the reconstructed data may still reveal private information that shows the identity. Another way to protect privacy is by adopting HE to encrypt gradients before aggregation, which can be computationally expensive due to the large size of the gradients [142].

As a result, DP, a rigorous mathematical tool, is widely employed in FL by adding noise to local gradients before they are transmitted [58]. However, the introduction of noise inevitably degrades accuracy. Therefore, some studies have investigated the trade-off between accuracy and privacy protection to improve overall performance [61, 101]. Although some work has proposed different relaxed bounds for DP in centralized FL in order to increase the accuracy, there is a gap in the literature on DPDFL with a relaxed bound for DP privacy loss or a relaxed noise scale while maintaining the privacy protection, leading to that the accuracy for DPDFL is much worse than that of non-private DFL due to unnecessary noise strength. Meanwhile, the work in [117, 118] suggests that an anonymous mechanism designed for gradients in FL training can reduce privacy leakage. Several works have implemented an anonymous mechanism into centralized DPFL by shuffling the gradients at a trusted third party before sending them to the central server [117, 118], which increases the communication bandwidth requirement and risks of privacy leakage in decentralized settings.

To address these challenges, this chapter proposes a novel anonymous DPDFL, namely ADPDFL, to improve privacy protection and accuracy. To be specific, every client will train the local model and add noise to the computed gradients locally, where the noisy gradients will be exchanged peer-to-peer anonymously. Furthermore, it investigates the privacy guarantee and the convergence bound for ADPDFL. The main contributions of this chapter are summarized as follows:

- This chapter proposes a DPDFL framework under two decentralized methods, the gossip and pseudo-central. In the gossip settings, every local clients broadcasts its local gradients to a subset of the local clients and aggregates the received the gradients to be its new local model. In the pseudo-central settings, a temporary leader is randomly chosen from the training clients in every round to collect all the local gradients, aggregate them into a new global model and broadcast the new model.
- This chapter introduces an anonymous communication protocol via hashed addresses which are changed every round, for the DPDFL to enhance the privacy protection. The anonymous mechanism ensures that no local client can know who they are exchanging models with, and they can not tell whether every two models come from the same client.
- Based on the proposed ADPDFL, a relaxed DP bound is proposed, where under the same privacy budget, every local client can have a smaller noise scale, leading to better accuracy. Furthermore, the corresponding base noise variance is derived for both two decentralized methods.
- The non-convex convergence bound for the proposed ADPDFL under two decentralized methods is derived in terms of the DP budget, TCSR and MER. The theoretical results show that there is an optimal number of TCSR and MER to balance convergence improvements and loss created by noise.
- Extensive simulations are conducted to present the effectiveness of the proposed framework. The results demonstrate how to select appropriate client settings for decentralized DPFL to achieve higher utility and also validate the theoretical results.

5.2 The Proposed Anonymous Differential Privacy-based Decentralized Federated Learning Framework

This section first presents the problem formulation and threat model for this chapter. Next, the proposed framework is introduced along with the decentralized topology, the DP mechanism, and the anonymous method. An overview of the ADPDFL is illustrated in Fig. 5.1.

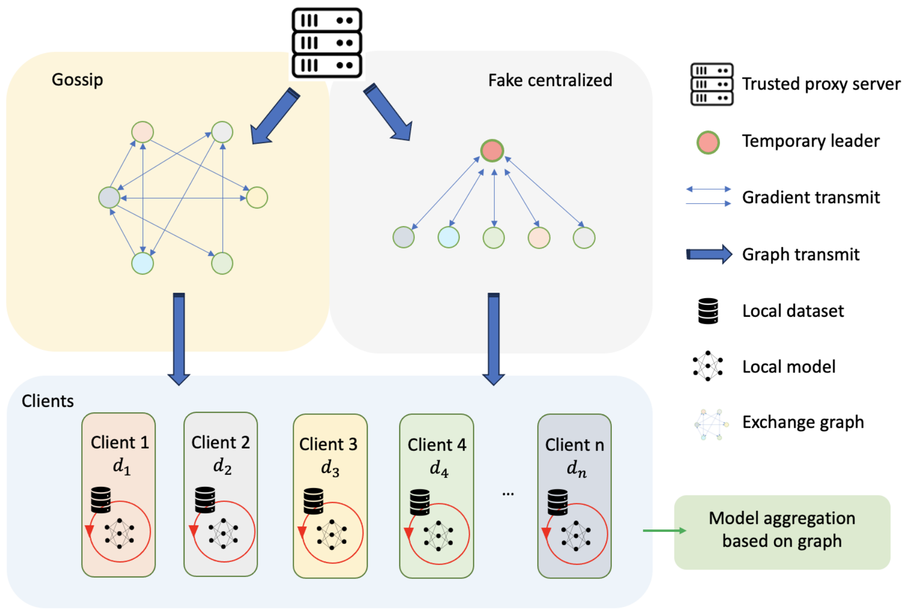


Figure 5.1: The illustration of the proposed ADPDFL framework.

5.2.1 System Model

In the DFL framework, a synchronized initial model is broadcast to distributed clients (DC) by a central server, such as a smart home manufacturer, and the central server will not participate in the future training. Then, DCs perform local training and exchange models in a peer-to-peer manner. After that, new models are aggregated at DCs, which are used for the next round of training. The DFL procedure can be formalized to solve an empirical risk minimization problem as follows:

$$\arg\min_{W} F(W) = \sum_{i=1}^{m} \frac{1}{m} F_i(W_i, D_i), \tag{5.1}$$

where W is the model parameters, $F(W)$ is the global loss, $F_i(W)$ is the loss function for the i-th DC, D_i is the data in the i-th DC, and m is the number of the aggregated local gradients in each round.

5.2.2 Threat Model and Assumption

This chapter considers a two-fold threat model. First, it examines the threat model in the context of semi-honest adversaries, which means that the local clients and the central server will perform the FL protocol correctly, but they may try to infer the original private data from the gradients via privacy-related attacks, such as reconstruction or membership inference attacks [47, 48]. To overcome this threat, DP is employed in FL to protect privacy. Second, combining multiple-round gradients from one user for a reconstruction attack can increase the risk of privacy leakage from DP. Ensuring the unlinkability of gradients across different rounds for DPDFL requires further research.

In this respect, ADPDFL is proposed to solve the aforementioned challenges effectively.

5.2.3 Differential Privacy-based Updating and Anonymous Decentralized Communication Protocol

In this subsection, the proposed ADPDFL framework is introduced, which incorporates a DP-based updating mechanism and an anonymous communication protocol, under two different decentralized methods.

Since the gossip algorithm provides strong robustness of clients drop-out and scalability for distributed clients, the first decentralized method for DCs follows the gossip method, based on which two factors, TCSR r_{tcs} and MER r_{mer}, are implemented to control the FL training procedure and derive the privacy loss. In every communication round, the number of DCs joining the FL training is m_{tcs}, where $m_{tcs} = r_{tcs} * M$. They train their model with their local data and compute the gradients with SGD. The gradients are transmitted to $m_e = r_{mer} * M$, DCs who have joined the local training in this round. Each DC aggregates their received gradients with their own, which becomes their new local model for the next round of training. Finally, the server can randomly collect one local aggregated model or aggregate all the local models to be the global model in the final round. This chapter will investigate how m_{tcs} and m_e affect the convergence performance of gossip DFL and privacy loss.

The second one follows a pseudo-central procedure, in which one DC is randomly chosen as a temporary leader in each round and each training DC transmits its gradients to the chosen DC. The chosen DC aggregates the received gradients and broadcasts them to the training DCs. Finally, the DC chosen in the final round uploads the final aggregated model to the central server after the training is aborted. This chapter will investigate how m_{tcs} affects the convergence performance and privacy loss of pseudo-central-based DFL.

To protect data privacy, DP is adopted by adding Gaussian noise to local gradients before sending them to others to achieve UDP. In every round, each training DC i first bounds the l_2 norm of its local gradients g_i^t to a threshold C as follows:

$$\widehat{g_i^t} = g_i^t / max(1, \frac{||g_i^t||_2}{C}). \tag{5.2}$$

After clipping, Gaussian noise is added to the clipped gradients as follows:

$$\overline{g_i^t} = \widehat{g_i^t} + N(0, \Delta f_i^2 \sigma^2), \tag{5.3}$$

where t is the index of the current round, σ is a preset noise scale and Δf_i is the sensitivity of the i-th client. The required noise scale for the proposed framework is derived in Section 5.3.1. Given DP settings, each local DC has a maximum number of training rounds without privacy leakage. When the FL has an inadequate number of DCs with remaining training rounds, the training is stopped.

To enhance the privacy protection of DPDFL, this chapter designs an anonymous communication protocol between DCs. To formalize the gradient transmission paths, a trusted proxy server (TPS) is adopted. At the beginning of FL, the TPS collects the initial addresses of all DCs, which will be used as the destination for transmitting gradients in the FL system for each DC. In every round, the TPS will request DCs to join the training, and DCs who want to join the training will notify the TPS. After reaching m_{tcs} DCs joining, the TPS performs the same random computation on the current addresses, performs a hashing function on the new addresses of training DCs and sends the corresponding hashed address to each training DC.

The protocol of anonymous gradient transmission is introduced. For the gossip-based framework, it is assumed that every DC receives the same amount of noisy gradients. TPS randomly creates a graph with the hashed address and links between the DCs based on r_{mer}. The graph is broadcast to all the training DCs in each round. Then, DCs can transmit their noisy gradients to corresponding receivers anonymously. In the proposed anonymous protocol, who is exchanging gradients and whether they have received gradients from the same DC are unknown for every local DC. For the fake central-based framework, the role of the TPS is to randomly choose a temporary leader in each round from training DCs and broadcast the hashed address of the temporary leader to every joining DC. The temporary leader finishes the task of aggregating and broadcasting the gradients to the hashed addresses of other training DCs. Since the proposed frameworks only require the transmission of gradients from one DC to another DC without sending them to a third party, and the size of transmitting addresses and graphs are negligible compared to the gradients, the proposed framework can mitigate the communication burden of trusted third-party server and reduce privacy risks compared to the shuffling anonymous mechanisms.

Algorithm 5.1 ADPDFL: Anonymous differential privacy-based decentralized federated learning (1)

```
 1: procedure FL-TRAINING
 2:     Generate a global model $W^0$, broadcast $W^0$, $r_{mer}$, $r_{tcs}$, and $M$ to all the DCs and TPS
 3:     TPS collects initial addresses for all DCs
 4:     Each DC calculates its maximum training rounds for given DP settings.
 5:     for t=0 , 1 , 2,..., T do
 6:         TPS broadcasts the training requests to ask for participants
 7:         DCs that have remaining training rounds send joining requests to TPS
 8:         TPS counts the number of joining DCs and asks DCs to train after having $m_{tcs}$ DCs, where the set of joining DCs is denoted as $S^t$
 9:         for every DC $i$ in $S^t$ in parallel do
10:             Asks TPS for its new hashed ID
11:             $g_i^t \leftarrow$ DC-COMPUTE($W_i^t$)
12:             if $distr$ is $gossip$ then
13:                 $W_i^{t+1} \leftarrow$ GOSSIPCOMM
14:             end if
15:             if $distr$ is $fake\_central$ then
16:                 $W_i^{t+1} \leftarrow$ FAKECENCOMM
17:             end if
18:         end for
19:     end for
20:     Abort FL training when inadequate DCs have remaining training round
21:     if $distr$ is $gossip$ then
22:         The central server randomly selects one model or aggregates noisy models to be the final model
23:     end if
24:     if $distr$ is $fake\_central$ then
25:         The server asks the temporary leader for the current noisy model to be the final model
26:     end if
27: end procedure
28: procedure DC-COMPUTE($W_i^t$)
29:     Train $W_i^t$ with SGD for $E$ epochs
30:     Compute the gradients $g_i^t$
31:     Clip the gradients as $\widehat{g_i^t} = g_i^t / max(1, \frac{||g_i^t||_2}{C})$
32:     Add DP noise as $\overline{g_i^t} = \widehat{g_i^t} + N(0, \sigma^2 \Delta f_i^2)$
33: return $\overline{g_i^t}$
34: end procedure
```

Algorithm 5.2 ADPDFL: Anonymous differential privacy-based decentralized federated learning (2)

```
1: procedure GOSSIPCOMM($W_i^{t+1}$)
2:     TPS generates a random graph with DCs in $S^t$ and $r_{mer}$ for gradient transmission
       while ensuring every DC can receive the same number of gradients
3:     TPS broadcasts the graph to $S^t$
4:     for every DC $i$ in $S^t$ do
5:         Transmits $\overline{g_i^t}$ to its communicating DCs via the given hashed addresses
6:         Computes the average of the received gradients $\overline{g^t}$
7:         Computes the new local model $W_i^{t+1} = W_i^t - \overline{g^t}$
8:     end for
9: return $W_i^{t+1}$
10: end procedure
11: procedure FAKECENCOMM($W_i^{t+1}$)
12:     TPS randomly chooses a $DC_j$ from $S^t$ to be the pseudo-central DC and broadcasts
        the hashed address of $DC_j$ to every training DC in this round
13:     Every DC in $S^t$ except $DC_j$ transmits its gradients to $DC_j$ via the hashed ID
14:     $DC_j$ computes the averaged received gradients $\overline{g^t}$
15:     Compute the new local model $W_i^{t+1} = W_i^t - \overline{g^t}$
16:     $DC_j$ broadcasts $W_i^{t+1}$ to DCs who send gradients to it
17: return $W_i^{t+1}$
18: end procedure
```

In this chapter, the implementation of TPS in the gossip-based framework is to ensure that every DC can receive the same number of gradients for fairness. For the pseudo-central-based framework, the implementation of TPS is to ensure that the temporary leader in each round is not the same in an anonymous way. Therefore, a failing TPS can be easily replaced, given that it performs lightweight computation and communication tasks. For example, numerous trusted edge servers could effectively serve as TPS. Since communicating between hashed addresses is not the focus of this chapter, and it can be achieved by existing techniques, such as proxy networks and trusted routers, this chapter will not elaborate more details of the communication between the hashed addresses further.

The proposed ADPDFL framework is illustrated in Algorithms 5.1 and 5.2. The procedure *FL-TRAINING* introduces the overall FL training and the procedure *DC-COMPUTE* introduces how each DC computes its noisy gradients. Meanwhile, the proposed anonymous decentralized transmission of gradients is provided in the procedures *GOSSIPCOMM* and *FAKECENCOMM*.

5.3 Theoretical Results

This section provides a detailed derivation of the privacy guarantee analysis and the convergence bound for the proposed framework.

5.3.1 Privacy Analysis

In this subsection, the privacy analysis for the proposed ADPDFL framework is presented, and the required noise scale for the proposed framework is computed based on RDP. Since all gradients are clipped as (5.2), the l_2-sensitivity for the i-th local client is computed as follows:

$$\Delta f_i = \frac{sup_{g_i, g_i'} ||g_i - g_i'||_2}{n_i} = \frac{2C}{n_i}, \tag{5.4}$$

where n_i is the number of data that the i-th client uses for training. In this chapter, it is assumed all the clients have the same amount of training data. n will be used to denote n_i in the rest of this chapter.

The privacy guarantee and required noise can be formalized in Theorem 8.

Theorem 8 *Suppose that the probability of clients joining FL training in each round follows a uniform distribution, after T communication rounds, the proposed ADPDFL satisfies a global (ε, δ)-DP for any $\varepsilon < 2log(1/\delta)$ and $\delta > 0$, if σ^2 is chosen as follows:*

$$\sigma^2 \geq \frac{7 s_1 T(\varepsilon + 2log(1/\delta))}{m^2 \varepsilon^2}, \tag{5.5}$$

where $s_1 = r_{tcs}^2 m_e$ for gossip-based framework and $s_1 = r_{tcs}^2$ for pseudo-central framework. When the added noise to each local client is at least $N(0, 4C^2\sigma^2/m^2)$ where σ^2 needs to satisfy (5.5), the proposed ADPDFL can satisfy the guarantee of (ε, δ)-DP for the entire system.

The proof of Theorem 8 is presented in Appendix K.

5.3.2 Convergence Analysis

In this subsection, the convergence analysis for the proposed algorithm is proposed, which studies the convergence bound for the expectation of the global loss function $F(w)$ after T FL communication rounds.

First, $F(w) = \sum_{i \in S^t} \frac{1}{m_{tcs}} F_i(w)$ is defined, and some common assumptions for the analysis of FL are introduced.

Assumption 3 *(Smoothness) Each local objective function $\nabla F_i(W)$ is L-Lipschitz smooth, i.e, for any w and $w' \in \mathbb{R}$, the following can be obtained:*

$$||\nabla F(w) - \nabla F(w')|| \leq L||w - w'||. \tag{5.6}$$

Assumption 4 *(Bounded local divergence) The local divergence of the stochastic gradient is bounded as follows:*

$$\mathbb{E}||\nabla F_i(w) - \nabla F(w)||^2 \leq \zeta^2. \tag{5.7}$$

When the $F_i(W)$ is non-convex, the convergence results of the proposed framework are formalized as follows:

Theorem 9 *(Convergence bound of ADPDFL in non-convex settings) Under Assumption 3 and 4, if the proposed ADPDFL can satisfy (ε, δ)-DP with the required noise shown in Theorem 8, the proposed gossip-based ADPDFL framework can have the convergence upper bound after T communication rounds as follows:*

$$\begin{aligned} \mathbb{E}||\nabla F(w)||^2 \leq & \underbrace{\frac{2(F(\overline{W^0}) - F(\overline{W^*}))}{T(\rho_2 + \rho_2 L - L + 2)} - \frac{\zeta^2}{(\rho_2 + \rho_2 L - L + 2)}}_{Decentralized \quad FL} \\ & + \underbrace{\frac{\zeta^2 (L-1)(1-\rho_2)(m_{tcs} - m_e)}{m_e(\rho_2 + \rho_2 L - L + 2)(m_{tcs} - 1)}}_{client \quad sampling \quad error} + \underbrace{\frac{4LC^2\sigma^2(1-\rho_2)}{n^2 m_e(\rho_2 + \rho_2 L - L + 2)}}_{privacy \quad error}, \end{aligned} \tag{5.8}$$

where ρ_2 is the second largest eigenvalue of the decentralized communication graph matrix, and W^* is the optimal model.

Similarly, the pseudo-central-based ADPDFL can have the convergence upper bound after T communication rounds as follows:

$$\begin{aligned} \mathbb{E}||\nabla F(w)||^2 \leq & \underbrace{\frac{2(F(\overline{W^0}) - F(\overline{W^*}))}{T(2-L)} - \frac{\zeta^2}{(2-L)}}_{Decentralized \quad FL} \\ & + \underbrace{\frac{\zeta^2 (L-1)(m - m_{tcs})}{m_{tcs}(2-L)(m-1)}}_{client \quad sampling \quad error} + \underbrace{\frac{4LC^2\sigma^2}{n^2 m_{tcs}(2-L)}}_{privacy \quad error}, \end{aligned} \tag{5.9}$$

where,

$$\sigma^2 \geq \frac{7s_1T(\varepsilon + 2log(1/\delta))}{m^2\varepsilon^2}, \tag{5.10}$$

The proof of Theorem 9 is presented in Appendix L.

The convergence bound in (5.8) and (5.9) can be divided into three parts. The first two terms are the general convergence error bound for decentralized FL. The third term is the client sampling error from only aggregating part of the clients' gradients, also referred to as local drift. The final term is the privacy error caused by adding DP noise to the gradients.

From the above results, a relationship among the client choosing rates r_{tcs}, r_{mer}, privacy error, and convergence bound can be observed. For the gossip-based framework, when increasing either r_{tcs} or r_{mer}, the final term also increases. In addition, if increasing either r_{tcs} or r_{mer}, the connectivity of the transmitting graph of each round also increases (a smaller ρ_2), causing the second term and the third term to decrease. Therefore, there is also an optimal r_{tcs} and r_{mer}, which can balance the trade-off between convergence and privacy to obtain a smaller convergence bound. For the pseudo-central framework, when increasing the r_{tcs}, it is found that the privacy error increases since more clients send out their gradients, and the third term decreases since more sufficient aggregations are performed. Therefore, there is an optimal choice for r_{tcs} to have the minimum convergence bound.

5.4 Empirical Results

In this section, extensive simulations are performed to evaluate the performance of the proposed ADPDFL and to study the impact of r_{tcs} and r_{mer} on the models in terms of the corresponding number of clients, m_{tcs} and m_e.

5.4.1 Experimental Setup

Two datasets, both of which have ten classes in total, are used for FL training and testing. The first dataset is MNIST, which contains handwritten grayscale images, including 60000 training data and 10000 testing data [38]. The second is CIFAR10, which is a dataset of different colorful animals and tools, including 50000 training data and 10000 testing data [129]. The proposed framework is performed with 100 clients in total. When training MNIST, an MLP with two hidden layers, where each layer has 200 hidden units, is used as the local model. When training CIFAR10, a CNN is used as the local model. The

CNN model has two 5×5 convolution layers, both followed by a 2×2 max-pooling layer, where the first convolution layer has 16 channels and the second has 32 channels. After convolution layers, there is an additional fully connected layer with 512 hidden units before the output layer. For both models, ReLU is used as the activation for the hidden layers.

In this section, simulations are first performed with IID data to evaluate the impact of the parameters. For the IID setting, the data is randomly shuffled and assigned to each client in a balanced way. In addition, simulations are also conducted to show the performance on non-IID data. For the non-IID settings, each class of data is split into 20 shards (200 shards in total), and each client is assigned two shards of different classes. The simulation data settings are denoted to IID-MNIST, IID-CIFAR10, non-IID-MNIST and non-IID-CIFAR10, respectively.

The learning rate is fine-tuned to 0.001, where a higher value brings too much noise, and a lower value slows the convergence. With regard to the settings of the DP mechanism, $\varepsilon = 8$ and $\delta = 1e-5$ are chosen for most of the simulations in this section. Since different m_{tcs} and m_e may have different optimal clipping bound values, simulations with m_{tcs} of 70 on pseudo-central-based framework are first performed with different clipping bounds for 20 rounds to choose the clipping bounds for other m_{tcs} and m_e. As shown in Table 5.1, the clipping bound of one produces the highest accuracy performance, while the smaller ones preserve less information and the larger ones bring too much noise. Hence, a clipping bound of one is chosen for all the other simulations in this section.

Table 5.1: Accuracy of the pseudo-central-based framework on IID-MNIST and IID-CIFAR10 with different clipping bounds.

clipping bound / dataset	0.5	1	2	4
IID-MNIST	0.3855	0.4123	0.1849	0.1253
IID-CIFAR10	0.3776	0.3840	0.3836	0.3724

In all simulation figures and tables, m_{tcs} is denoted as mt, and m_e is denoted as me. Most of the results in the figures and tables are the averaged results from multiple simulations under the same settings due to the randomness of the training and DP noise. To evaluate the performance of the decentralized FL without global model synchronization, the local test accuracy at each round is measured based on the global test dataset before and after local aggregation. The largest value in all training clients' local test accuracy in each round is recorded as the test accuracy for that round, which is used for all tables and figures in this section.

Although the simulations aim to study the best client selection rates for the proposed ADPDFL, the results of DP-FedAvg and a Non-Private version framework, which is the proposed ADPDFL without DP noise and clipping, are provided for comparison, where the base noise variance of DP-FedAVG is set to two with other unchanged. The implementation of the anonymous mechanism is not considered in the simulation since this chapter focuses on the accuracy performance under DP, and the implementation of it does not affect the accuracy performance. However, the calculation of the required noise base variance still involves the anonymous mechanism.

5.4.2 Evaluation of IID-MNIST

In this subsection, the performance of the proposed framework on IID-MNIST is presented.

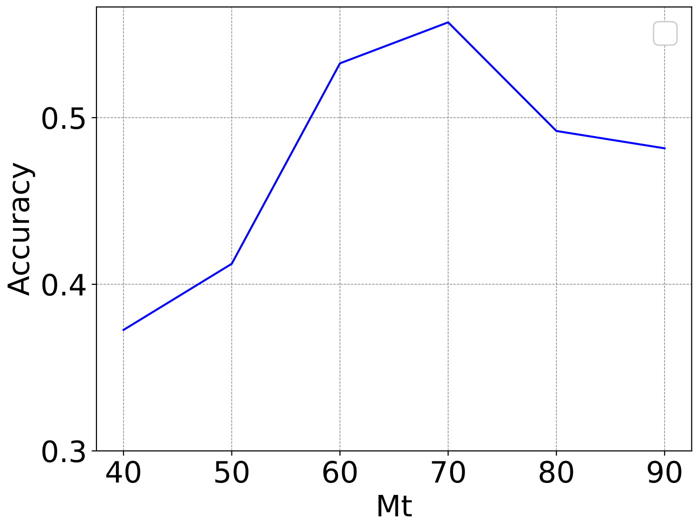


Figure 5.2: Accuracy of the pseudo-central-based framework on IID-MNIST with different m_t for 20 rounds.

The pseudo-central-based framework with different parameter settings is studied, including m_{tcs}, ε and T. The performance under different numbers of training clients for 20 communication rounds is presented in Fig. 5.2, where it is observed that accuracy increases as the number of training clients increases to 80. When m_{tcs} is greater than 80, the accuracy performance becomes worse since the noise is greatly increased. This observation shows that there is an optimal number of training clients to achieve the best performance, which is consistent with the theoretical results. The performance under different ε is also evaluated. As shown in Fig. 5.3, when ε increases, accuracy performance

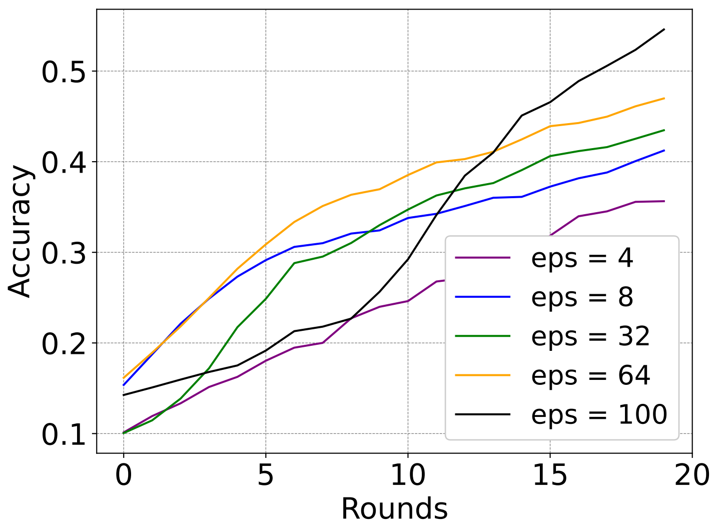


Figure 5.3: Accuracy of the pseudo-central-based framework on IID-MNIST with different ε.

is improved as the noise variance decreases.

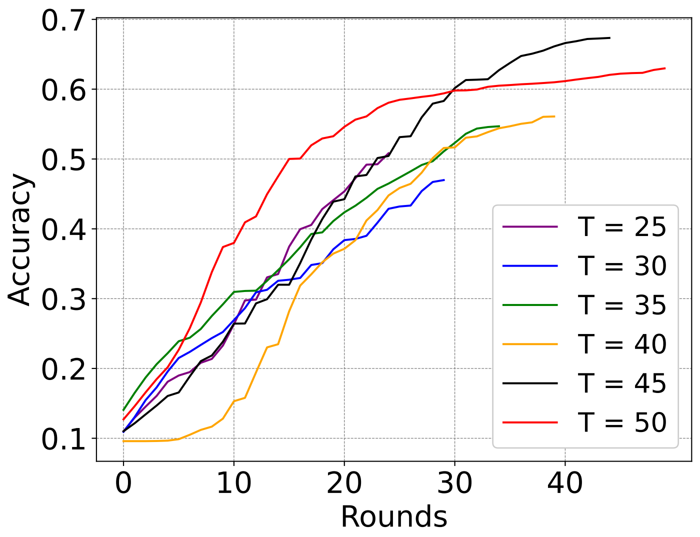


Figure 5.4: Accuracy of the pseudo-central-based framework on IID-MNIST with different total FL communication rounds.

Furthermore, the performance of different total FL communication rounds is studied in Fig. 5.4. It is shown that as the number of total communication rounds increases from 20 to 30, the accuracy performance decreases, while increasing T after 30 brings a better accuracy performance. This finding is caused by increasing the total communication round, which can bring a larger noise variance, and more total training rounds can also improve the accuracy performance, which demonstrates that there is a balance between the number of training rounds and the accuracy performance for the proposed ADPDFL.

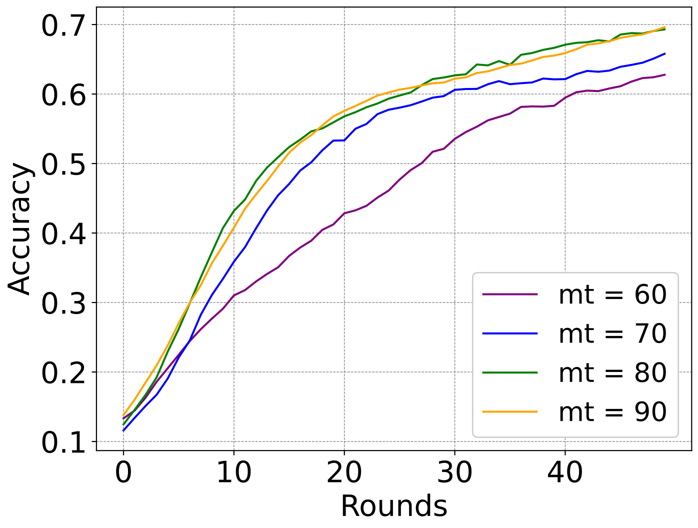


Figure 5.5: Accuracy of the gossip-based framework on IID-MNIST with different m_t.

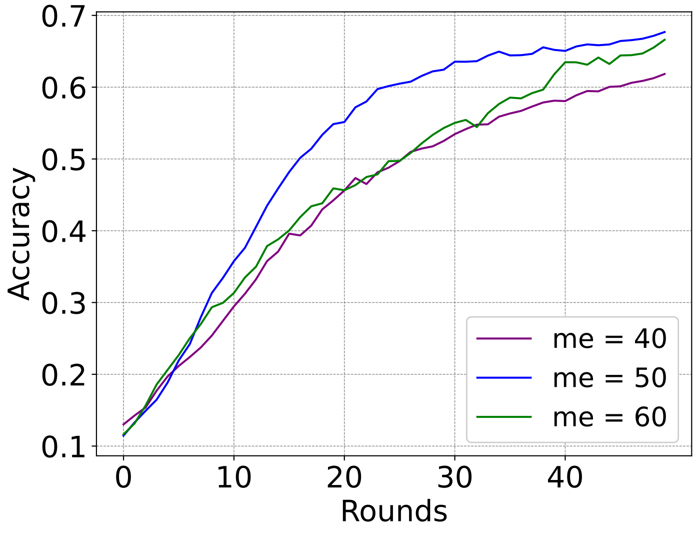


Figure 5.6: Accuracy of the gossip-based framework on IID-MNIST with $m_t = 70$ and different m_e.

The performance of the gossip-based framework is studied to evaluate the impact of m_e and m_{tcs}, where the framework is carried out with 50 communication rounds. As shown in Fig. 5.5, the accuracy performance increases when m_{tcs} increases from 60 to 80 and m_e is 50. Although the accuracy performance of $m_{tcs} = 90$ is almost the same as that of 80, it generates additional computation and communication overheads, which means that there is also an optimal number of m_{tcs} in the proposed gossip-based framework.

The performance of different m_e is also studied as presented in Figs. 5.6, 5.7 and 5.8. As shown in Figs. 5.6 and 5.7, the optimal number of m_e is 50 while other values produce worse accuracy performance. However, Fig. 5.8 shows that when m_e is 50, the accuracy performance is worse than other values. Therefore, there are different optimal numbers of m_e under different m_{tcs} for the proposed ADPDFL.

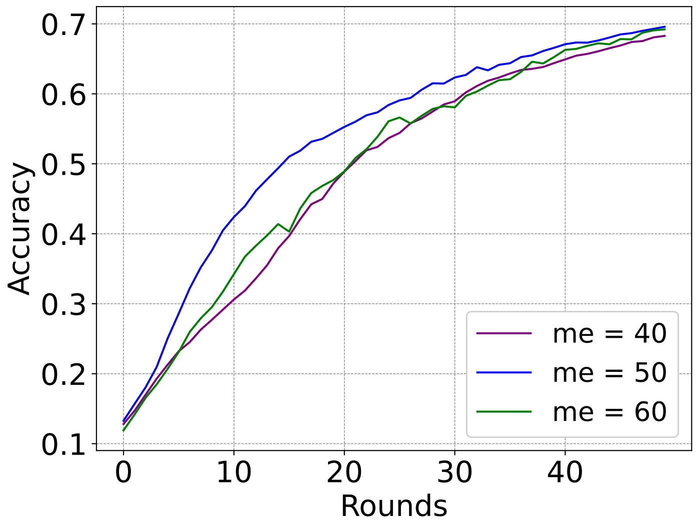


Figure 5.7: Accuracy of the gossip-based framework on IID-MNIST with $m_t = 80$ and different m_e.

5.4.3 Evaluation of IID-CIFAR10

In this subsection, the performance of the proposed framework with IID-CIFAR10 is evaluated. The pseudo-central-based framework is first studied, where the number of communication rounds is 50. The impact of different m_{tcs} is evaluated in Fig. 5.9. The proposed framework is shown to have an increase in accuracy performance when m_{tcs} increases from 20 to 60 except 30, which brings a worse accuracy performance than the one with 20 and 40.

In addition, the impact of m_e and m_{tcs} on the performance of the proposed gossip-based framework for 50 communication rounds is evaluated. As shown in Figs. 5.10, 5.11 and 5.12, different m_e have different optimal numbers of m_{tcs} to achieve the best accuracy. Specifically, when m_{tcs} is 70, $m_e = 40$ provides the best accuracy performance. For $m_{tcs} =$ 80, the accuracy performance of $m_{tcs} = 30$ is the largest. Finally, when m_{tcs} is 90, even

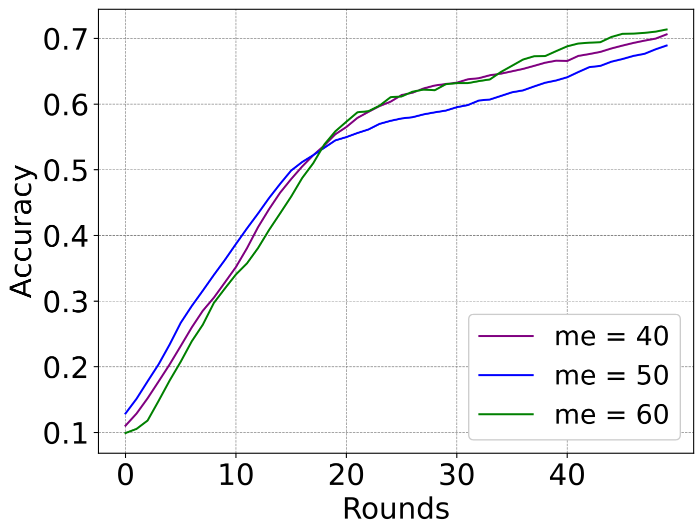


Figure 5.8: Accuracy of the gossip-based framework on IID-MNIST with $m_t = 90$ and different m_e.

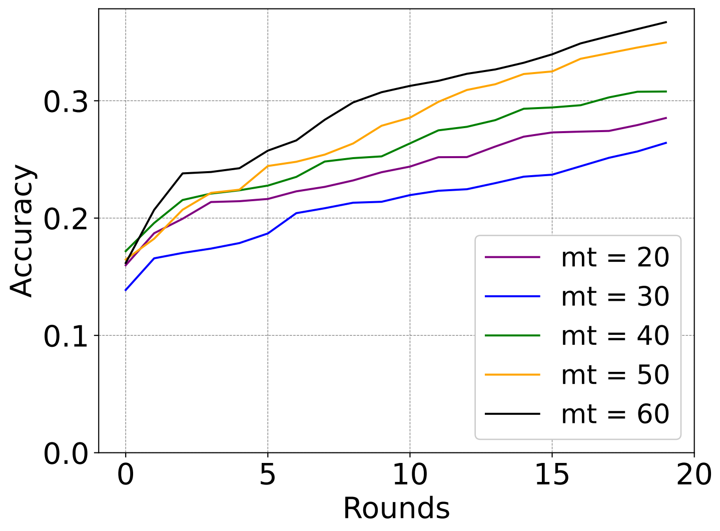


Figure 5.9: Accuracy of the pseudo-central-based framework on IID-CIFAR10 with different m_t.

though $m_e = 40$ and $m_e = 50$ produce similar results, $m_e = 50$ has more communication costs, making $m_e = 40$ the optimal setting.

5.4.4 Evaluation of Non-IID-MNIST

This subsection evaluates the impact of m_{tcs} and m_e on the performance of the proposed framework under non-IID-MNIST, where the number of communication rounds is 50. The

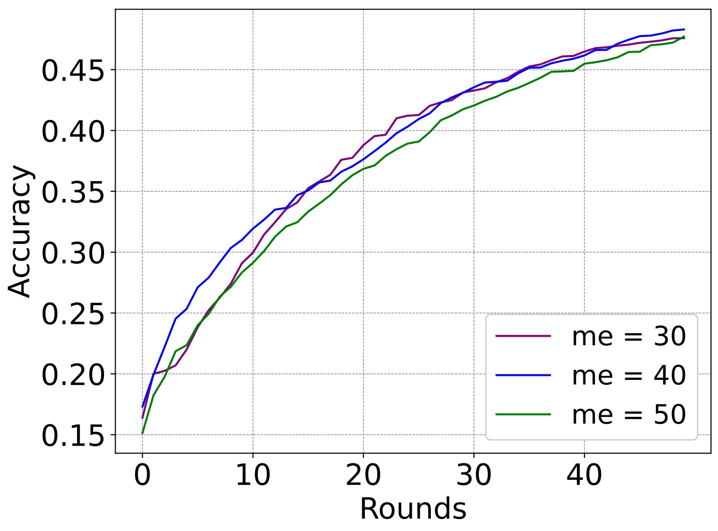


Figure 5.10: Accuracy of the gossip-based framework on IID-CIFAR10 with $m_t = 70$ and different m_e.

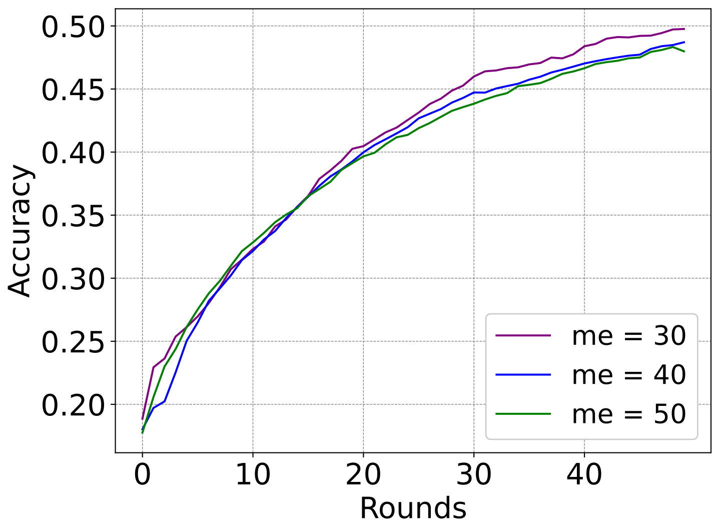


Figure 5.11: Accuracy of the gossip-based framework on IID-CIFAR10 with $m_t = 80$ and different m_e.

proposed pseudo-central-based framework is first performed. In Fig. 5.13, it is shown that there is an optimal number of m_{tcs}, which is 80, to achieve the best accuracy performance in these settings. Then, the impact of m_{tcs} and m_e on the proposed gossip-based framework is studied. Table 5.2 shows that for every m_{tcs}, there is an optimal number of m_e to achieve the best accuracy performance and vice versa. Overall, when m_{tcs} is 90 and m_e is 50, the accuracy performance is the largest.

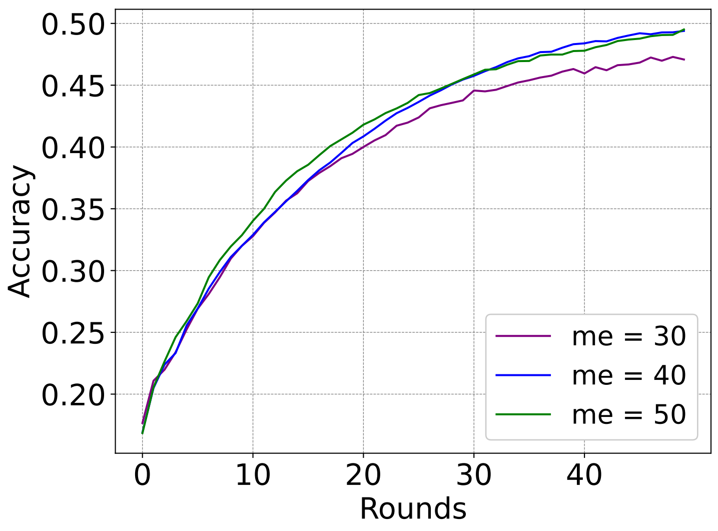


Figure 5.12: Accuracy of the gossip-based framework on IID-CIFAR10 with $m_t = 90$ and different m_e.

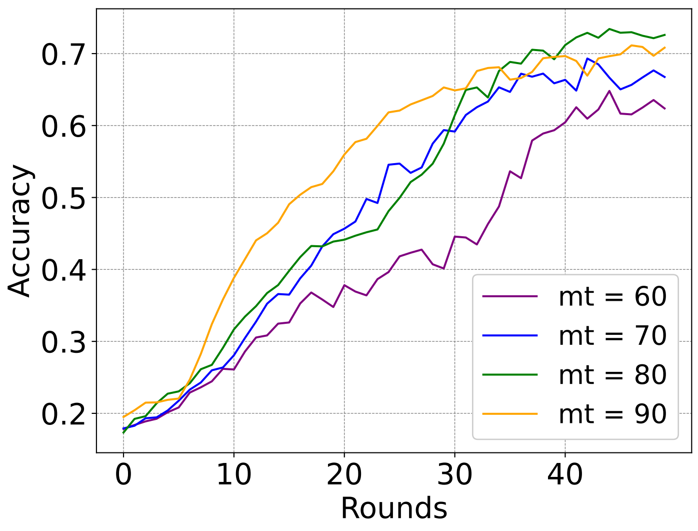


Figure 5.13: Accuracy of the pseudo-central-based framework on non-IID-MNIST with different m_t.

5.4.5 Evaluation of Non-IID-CIFAR10

In this subsection, the impact of m_{tcs} and m_e is evaluated on the performance of the proposed framework under non-IID-CIFAR10, where the number of communication rounds is 50. The proposed pseudo-central-based framework is performed and shown in Fig. 5.14. The accuracy performance increases as m_{tcs} increases from 60 to 90. In addition, the performance of the gossip-based framework is explored. As shown in Table 5.3, increasing

Table 5.2: Accuracy of the gossip-based framework on non-IID-MNIST with different m_t and m_e.

m_t \ m_e	40	50	60
70	0.6261	0.6103	0.6105
80	0.5821	0.6240	0.6081
90	0.5714	0.6297	0.6266

m_e leads to a higher accuracy performance most of the time. However, for different m_e, there is a different relationship between m_{tcs} and the accuracy performance. For example, when m_e is 40 and 50, $m_{tcs} = 80$ has the worst accuracy performance. On the other hand, when m_e is 30, $m_{tcs} = 70$ has a poorer accuracy performance than the others.

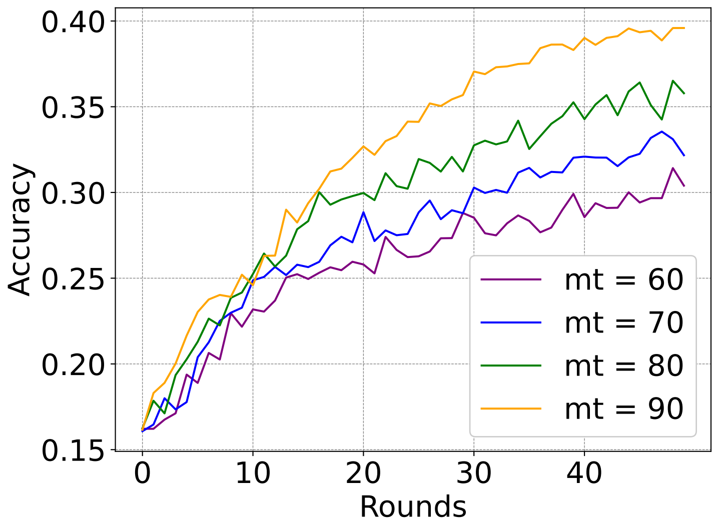


Figure 5.14: Accuracy of the pseudo-central-based framework on non-IID-CIFAR10 with different m_t.

Table 5.3: Accuracy of the gossip-based framework on non-IID-CIFAR10 with different m_t and m_e.

m_t \ m_e	30	40	50
60	0.2572	0.2842	0.3058
70	0.2564	0.2917	0.3017
80	0.2661	0.2830	0.3003
90	0.2717	0.2974	0.3042

5.4.6 Discussions and Comparisons

This subsection first summarizes the impact of m_{tcs} and m_e on the proposed framework. For most settings, there is an optimal number of training clients or exchange clients to achieve the best accuracy, which is consistent with the theoretical results that the optimal number of m_{tcs} and m_e has a larger effect on reducing convergence loss than the corresponding noise loss. On the other hand, there are some cases in which certain values of m_{tcs} and m_e bring the worst performance while increasing or decreasing the number of corresponding clients can improve the accuracy performance. Meanwhile, in most cases, it is found that pseudo-central-based frameworks outperform the gossip-based frameworks with lower communication costs, which is caused by gossip-based frameworks requiring larger noise since each client needs to send out multiple gradients in every round. However, in some large-scale real-world scenarios, the local devices may have limited resources to be the pseudo-central leader to aggregate and broadcast a large amount of gradients, necessitating the gossip-based framework. Meanwhile, in a gossip-based framework, even though different m_e and m_{tcs} can deliver a similar accuracy performance, they may have very different communication costs, and smaller values have a smaller probability of privacy leakage, which means that for a gossip-based framework, it needs to choose the smallest $m_e \times m_{tcs}$ of the ones that bring the highest accuracy. Therefore, more effective solutions for improving the accuracy with minimum communication and computation costs for gossip-based framework needs further optimization.

In addition, the performance of the centralized DP-FedAvg and the proposed DFL framework of a Non-Private version is presented in Table 5.4, where the number of training clients is 70 for all frameworks, and the number of exchange clients is 30 for gossip-based frameworks. As shown in Table 5.4, the proposed ADPDFL only has a small accuracy degradation compared to centralized DPFL and Non-Private DFL with greater privacy protection.

Table 5.4: Accuracy of different frameworks on different datasets, where Non-Private is the proposed DFL framework without implementing DP and Fake-cen denotes the pseudo-central-based framework.

Framework / Dataset	Gossip ADPDFL	Fake-cen ADPDFL	Gossip Non-Private	Fake-cen Non-Private	DP-FedAvg
IID MNIST	0.6701	0.6814	0.6812	0.7005	0.7237
IID CIFAR10	0.4849	0.4971	0.4872	0.5012	0.5369
non-IID MNIST	0.5375	0.6723	0.5624	0.6998	0.7213
non-IID CIFAR10	0.2634	0.3365	0.2705	0.3477	0.3872

5.5 Summary

This chapter first proposes a DPDFL framework based on pseudo-central and gossip topology. Second, to enhance privacy protection, a novel anonymous mechanism for the DPDFL is designed by implementing a TPS to randomly generate and broadcast the gradients exchange graph for the decentralized clients. The TPS only sends the corresponding address to every training client for their gradients transmission, where the address is changed every round. Therefore, no client can know who they are communicating with or whether every two gradients from two rounds are coming from the same clients. The proposed anonymous mechanism can reduce the bandwidth burden, computation overhead and privacy risks compared to the shuffling mechanism. Third, based on the proposed framework, the required base noise variance for both the proposed gossip-based and pseudo-central-based frameworks is derived to satisfy the RDP guarantee. Fourth, the non-convex convergence bound for the proposed frameworks is derived in terms of the DP budget and the number of training and exchanging clients, which shows that there is an optimal number of training and exchanging clients to achieve the best accuracy performance. Finally, extensive simulations are conducted to evaluate the performance of the proposed frameworks on MNIST and CIFAR10 with both IID and non-IID settings, which validate the theoretical results and also show that there is an optimal number of communication rounds.

Chapter 6

Conclusion and Future Work

This chapter outlines the main contributions of this thesis, which focuses on enhancing the utility and privacy protection level of DPFL and provides correspondingly detailed framework designs, performance and privacy analysis, and simulations. In addition, an introduction to challenges and possible future work is provided.

6.1 Conclusion

This thesis first reviews the background and literature to identify the existing challenges in DPFL, including degraded accuracy and expensive communication costs. To solve these challenges, three DPFL frameworks are proposed to improve accuracy and reduce communication costs while enhancing privacy protection. Additionally, a privacy loss evaluation methodology is proposed to guide optimal noise scale selection for DPFL. The previous chapters are concluded in the following of this section.

In Chapter 1, an introduction to FL is presented, covering its fundamental principles and classification. This chapter highlights the challenges in existing FL frameworks, with a particular focus on privacy leakage issues, which motivates the research conducted in this thesis.

In Chapter 2, the foundational concepts and formulas of FL, DP, DP-based learning and reconstruction attacks are presented. In addition, a literature review of DPFL and related fields is introduced.

In Chapter 3, two DPFL frameworks are proposed to improve the convergence performance under DP noise in two sections. Section 3.2 introduces a new FL framework by applying DP both locally and centrally in order to strengthen the protection of participants' privacy. To improve accuracy of the model, this framework also applies sparse gradients before adding noise and MGD on both the server side and the client side. Moreover, using sparse gradients can reduce the total communication costs. This section provides the experiments to evaluate the proposed framework, and the results show that the framework not only outperforms other DP-based FL frameworks in terms of model accuracy but also provides a stronger privacy guarantee. Besides, the framework can save up to 90% of communication costs while achieving the best accuracy.

Section 3.3 proposes two strategies to improve the convergence performance of the LDP-FL. Both methods are achieved by modifying the local objective function to limit the effect of LDP noise on convergence without compromising the privacy protection level. The first one is to minimize the difference between the noisy gradients and original gradients, and the second one is to minimize the expected loss due to the noise by incorporating the corresponding values to the objective function. A detailed framework that adopts the LDP scheme and two strategies is then introduced. This section also presents the theoretical convex and non-convex convergence guarantees of the proposed framework. Finally, simulations on the framework show that the framework can accelerate the convergence up to 40% faster and improve accuracy under different noise compared with other DPFL frameworks.

In Chapter 4, a privacy leakage evaluation methodology consisting of four key aspects is proposed. The first aspect is the basic evaluation procedure, which utilizes reconstruction attacks. To evaluate the privacy leakage, the second aspect is to propose a new metric by using different pre-trained models to infer the outputs of reconstructed and true images. The distance between these outputs is explored to measure the privacy leakage. Third, two different attack manners are designed: one targeting the sum of all rounds' gradients and the other targeting gradients from each individual round. By studying them, it is empirically demonstrated that anonymous mechanisms can enhance privacy protection. Finally, two common clipping methods for DPFL are studied to evaluate their impacts on privacy protection. The simulation results show the optimal settings for noise and clipping to guarantee privacy. All the findings in this chapter are summarized to guide choosing DP settings for personalization, enhancing privacy protection and improving accuracy.

In Chapter 5, an anonymous decentralized DPFL framework, called ADPDFL, is proposed along with an insightful analysis of privacy protection and the convergence bound. Specifically, this chapter first considers two decentralized DPFL methods based on gossip and pseudo-central topologies. In addition, this chapter investigates the impact of the TCSR in each round for both methods and MER in the gossip-based method on the convergence performance. To enhance privacy protection, an anonymous mechanism, in which clients do not know who they are communicating with and cannot determine whether they are communicating with the same client across training, is introduced. Next, this chapter proposes a relaxed DP mechanism for ADPDFL satisfying RDP guarantee and derives the required noise in terms of DP settings, TCSR and MER. After that, the theoretical convergence bound for the proposed ADPDFL under non-convex settings is derived, which suggests that there is an optimal number of TCSR and MER to achieve the best accuracy performance. Finally, a series of simulations are conducted to evaluate the performance of the proposed ADPDFL and validate the theoretical results.

6.2 Future Work

Even though this thesis has proposed several DPFL frameworks to tackle some existing challenges, there are still problems for DPFL that need to be further researched. In this section, the open issues and future research topics for DPFL are introduced.

6.2.1 Industrial Applications

While this thesis focuses on developing and analyzing theoretical frameworks of DPFL, future work should integrate real-world industrial applications. For example, the proposed DPFL frameworks can be implemented in smart homes or healthcare monitoring systems. After collecting IoT data, local clients can train local models using their phones or computers, which are then sent to the central server for aggregation. During training, the proposed frameworks in this thesis can be implemented to prevent central servers from revealing clients' living patterns or health conditions. Meanwhile, a comparative analysis with existing frameworks on industrial IoT data will be conducted to highlight the improvements.

Beyond integrating the proposed work into Industrial IoT applications, further extensions to the work should be explored. To achieve practical industrial applications of DPFL, the scalability of DPFL should be explored. In industrial settings, a massive number of devices may join the training, which may introduce a significant communication and computation burden for central servers. Therefore, an effective method to manage those clients for training and aggregation is a crucial research direction.

Additionally, many recent industrial scenarios focus on large-scale models. Although the DP noise is well-designed, the accuracy becomes more degraded due to the high-dimension embedding, which is common in large-scale models. Therefore, further research should investigate techniques to improve accuracy in large-scale DPFL systems

[55].

These extensions and implementation can enhance the significance and practical utility of the research, effectively connecting theoretical developments with practical needs.

6.2.2 Privacy Budget Allocation

Even though extensive literature and this thesis have proposed various frameworks to improve the performance degradation due to the DP noise, the accuracy performance of DPFL is still worse than non-private FL [55, 143]. The majority solution to this problem is to design different privacy budget allocation mechanisms to achieve a more relaxed privacy loss calculation, which leads to lower required noise [55, 143]. Therefore, a tighter bound than the most popular solution, RDP, in k-fold use of the DP mechanism is worth researching to further improve accuracy. Additionally, while most existing privacy loss derivations provide a lower bound, an upper bound on privacy loss may also be necessary [144]. Since current approaches often assign a fixed privacy budget to each local client during training, a more adaptive privacy budget allocation method is needed throughout the training process to better balance the trade-off between privacy and accuracy [145].

6.2.3 The Trade-off between Fairness and Privacy

Fairness in FL can be defined as two aspects [145, 146]. The first aspect is to eliminate bias among different groups of data, as FL systems might show bias toward certain population groups based on sensitive attributes such as gender, race, or age. The second aspect is to mitigate the unfairness due to weights assignment. Traditional FL normally assigns greater weights to the clients with larger datasets [14]. However, this may drive the global model to contain more knowledge for those clients, which not only brings unfairness for the clients with smaller datasets but also leads to a less generalized global model. In FL, fairness and privacy protection are two conflicting objectives [145, 146]. For example, the unfairness between groups due to the accuracy difference may be enlarged, as local clients requiring stronger privacy have larger accuracy degradation than others, which leads to being treated unfairly for poor performance. On the other hand, ensuring fairness may require the server to access more sensitive information to mitigate the bias in the clients' data distribution, raising more privacy concerns. Therefore, more effective methods for providing required privacy protection for each local client without bringing unfairness need to be further researched [145, 146].

6.2.4 Heterogeneous Privacy Protection Level

Most DPFL frameworks have not accounted for privacy heterogeneity, often assuming that all local clients require the same level of privacy protection [147]. However, in real-world scenarios, different local clients may have varying privacy requirements [55, 147]. For example, patients in a hospital database may be more concerned about the privacy of their medical records compared to others, necessitating stronger privacy protections for them. Furthermore, regulations and laws often impose stricter protections on certain types of data [10, 11]. Therefore, designing an effective heterogeneous privacy protection scheme for DPFL is an important research direction that better aligns with diverse privacy needs.

6.2.5 Unified Privacy Auditing Frameworks

Due to the various applications in the IoT environment, there may be different requirements for privacy protection levels and targets, such as LDP, CDP, approximate DP and RDP [57, 59]. Many recent studies use different DP mechanisms and composition theorems, which may have very different bounds for privacy loss. The most commonly used method for evaluating privacy is the privacy budget ε, which is a theoretical guarantee and varies based on the DP mechanisms. In addition, some regulations may require auditing the privacy during FL [55]. Therefore, a unified privacy auditing framework should be explored to provide a more practical and intuitive privacy measurement in order to ensure compliance with the regulation and to help researchers improve DPFL [55].

6.2.6 Adaptive Defense of Privacy

The current PPFL literature focuses on the passive defense of privacy [143]. To be specific, they normally first assume that clients or servers are curious about the sensitive data and implement corresponding defenses for local clients. Moreover, the clients and servers adopt the privacy protection techniques on the gradients over the training, which may cause unnecessary utility degradation since attackers may not always be active. Another challenge for privacy protection is that privacy attacks also keep evolving, necessitating stronger privacy protection techniques. To tackle the aforementioned problems, a more comprehensive and proactive privacy protection framework is essential to identify privacy attacks or risks across various scenarios, enabling better implementations of appropriate DP settings and timing [143].

While both DP and HE have been proven to ensure privacy protection in FL, DP can lead to accuracy degradation, and HE may introduce heavy computation overhead. There are a few studies that have already combined DP and HE to improve privacy protection and overall utilities, but these methods may still perform worse than the non-private FL [121]. The combination of DP and HE can be further optimized to reduce computational costs and improve accuracy while maintaining strong privacy guarantees [51]. In addition, the existing works assume that HE and DP will be applied together at the same time across the FL training, which may be unnecessary. A novel technique to automatically choose the proper privacy-preserving technique, either alone or in combination, based on the current environment and models, can be designed.

6.2.7 Security and Robustness

Most existing DPFL frameworks assume that the local clients are semi-honest. However, FL is vulnerable to Byzantine faults and malicious clients, which can interfere with the training process. Detecting anomalous clients often requires the collection of additional information, potentially increasing the risk of privacy leakage [122]. The accuracy degradation due to the noise may cause the system to identify the local users as malicious clients. Although a few works have proposed Byzantine-robust DPFL, these works sacrifice the privacy protection level, and they only work against a smaller proportion of Byzantine clients than that in non-private FL [148]. Therefore, robust aggregation methods in DPFL that ensure resilience against adversarial behaviors while maintaining privacy protection need to be further researched.

Appendices

A Proof of Lemma 5

To obtain the expected change on dz_k, dz_k is first obtained by reversing the backpropagation. Then, the expected change dz_k generated by dw_k and db_k are computed correspondingly, which is noted as $d_{dw_k}z_k$ and $d_{db_k}z_k$. Based on (2.8), the following can be obtained:

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

$$dw_k \cdot a_{k-1} = \frac{1}{n'} dz_k \cdot (a_{k-1}^T) \cdot a_{k-1}, \tag{A.2}$$

$$d_{dw_k}z_k = dw_k \cdot a_{k-1}. \tag{A.3}$$

Then, as (2.9) is a vectorization implementation for the DNN training and $d_{db_k}z_k$ cannot be directly obtained, an approximate value as $d_{db_k}z_k = db_k$ is computed. Therefore, for the first layer, its dz_k through reverse backpropagation is computed as

$$\begin{aligned} dz_k &= d_{dw_k}z_k + d_{db_k}z_k \\ &= dw_k \cdot a_{k-1} + db_k. \end{aligned} \tag{A.4}$$

After that, for all the following layers, to compute the reversed dz_k, it also needs to consider the derivative part of the dz_{k-1}. Based on (2.7)-(2.10), the following can be obtained:

$$\begin{aligned} dz_{k-1} &= da_{k-1} \times g'_{act_{k-1}}(z_{k-1}) \\ &= (w_k)^T \cdot dz_k \times g'_{act_{k-1}}(z_{k-1}), \end{aligned} \tag{A.5}$$

$$w_k \cdot dz_{k-1} = dz_k \times g'_{act_{k-1}}(z_{k-1}). \tag{A.6}$$

Next, the multiplication term $g'_{act_{k-1}}(z_{k-1})$ needs to be simplified, where $g_{act_{k-1}}(z_{k-1})$ is the activation function of the $(k-1)$-th layer. For all the hidden layers (except the first one), ReLU activation is used in this thesis. Therefore $g'_{act}(z)$ is a function of following:

$$g'_{act}(z) = \begin{cases} 0 & z < 0, \\ 1 & others. \end{cases} \tag{A.7}$$

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 follows:

$$d_{dz_{k-1}}z_k = w_k \cdot dz_{k-1} \times g'_{act_{k-1}}(z_{k-1}). \tag{A.8}$$

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

$$\begin{aligned} dz_k &= d_{dz_{k-1}}z_k + d_{w_k}z_k + d_{b_k}z_k \\ &= dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g'_{act_{k-1}}(z_{k-1}). \end{aligned} \tag{A.9}$$

Finally, to obtain the expected change in the noisy gradients on the loss, it needs to substitute the expected changes of dw_k, db_k and dz_{k-1} into (A.9), 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. This completes the proof.

B Proof of Theorem 3

To find the expected change in the final loss function, the relation between (3.14) and (3.17) is computed as follows:

$$\frac{dz_k}{d\overline{z_k}} = \frac{dw_k \cdot a_{k-1} + db_k + w_k \cdot dz_{k-1} \times g'_{act_{k-1}}(z_{k-1})}{N(dw_k) \cdot a_{k-1} + N(db_k) + w_k \cdot d\overline{z_{k-1}} \times g'_k(z_{k-1})}. \tag{A.10}$$

Then, the noise is generated as follows:

$$N(\nabla W) = N(0, S_i^2\sigma^2), \tag{A.11}$$

where S_i is calculated as $\frac{||\nabla W||}{n_i}$ for every layer. Therefore, the following can be obtained:

$$\frac{N(dw_k)}{dw_k} = \frac{||N(dw_k)||}{||dw_k||} = \frac{\frac{||dw_k||*\sigma}{\sqrt{n_i}}}{||dw_k||} = \frac{\sigma}{\sqrt{n_i}}. \tag{A.12}$$

With the (2.6) and the constant label value Y, the expected change is computed as $da_K = \frac{\sigma}{\sqrt{n_i}} * (a_K - Y)$, which completes the proof.

C Proof of Corollary 1

Similar to Lemma 5 and Theorem 3, the expected change, $d\overline{z_k}$, is also generated with the expected changes on dz_{k-1}, db_k and dw_k. The properties of the convolution process are used, $a * b \times c = a * (b \times c) = (a * b) \times c$, to obtain the corresponding expected change $d_{\overline{z_{k-1}}} z_k$ from the expected changes on $d\overline{z_{k-1}}$ as follows:

$$\begin{aligned} d\overline{z_{k-1}} &= dz_{k-1} \cdot \frac{d\overline{z_{k-1}}}{dz_{k-1}} \\ &= \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dz_k^{conv} * rot180(a_k) \times g_k'(z_{k-1}), \end{aligned} \tag{A.13}$$

$$d_{\overline{z_{k-1}}} z_k^{conv} = \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dz_k^{conv}. \tag{A.14}$$

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

$$d_{\overline{w_k}}^{conv} z_k = \frac{d\overline{z_{k-1}}}{dz_{k-1}} \cdot dw_k. \tag{A.15}$$

Finally, by combining (A.10), (A.12) and (A.15), the following can be obtained:

$$d\overline{z_k} = \frac{\sigma}{\sqrt{n_i}} \cdot dz_k. \tag{A.16}$$

This completes the proof.

D Proof of Lemma 6

As shown in Algorithm 3.2, in one round of the training, the following can be obtained:

$$h(w_i^t; w^t) = F_i(W_i) + \lambda * S_i \sigma, \tag{A.17}$$

$$\begin{aligned}\nabla h(w_i^t; w^t) &= \nabla F_i(W_i) + \lambda * \sigma \nabla S_i, &(A.18)\\ W_i^t &= W^t - \nabla h(w_i^t; w^t), &(A.19)\\ \nabla W_i^t &= W^t - W_i^t, &(A.20)\\ \overline{W^{t+1}} &= W^t - \sum \frac{1}{m}(\nabla W_i^t + N_i). &(A.21)\end{aligned}$$

By substituting (3.9) into (A.18), the following can be obtained:

$$\nabla h(w_i^t; w^t) = \nabla F_i(W_i) + \lambda * \sigma \frac{W^t - W_i^t}{S_i}. \tag{A.22}$$

Since $F_i(W)$ is ρ-Lipschitz smooth as in Assumption 1, the following can be obtained:

$$F_i(\overline{W^{t+1}}) \leq F_i(\overline{W^t}) + (\nabla \overline{W^t})^T (\overline{W^{t+1}} - \overline{W^t}) + \frac{\rho}{2} ||\overline{W^{t+1}} - \overline{W^t}||^2, \tag{A.23}$$

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, it has $F(\overline{W^t}) = \mathbb{E}\{F_i(\overline{W^t})\}$ and $\nabla F(\overline{W^t}) = \mathbb{E}\{\nabla F(\overline{W_i^t})\}$.

Then, the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}}) - F(\overline{W^t})\} \leq \mathbb{E}\{\nabla(F(\overline{W^t}))^T (\overline{W^{t+1}} - \overline{W^t})\} + \mathbb{E}\{\frac{\rho}{2} ||\overline{W^{t+1}} - \overline{W^t}||^2\}. \tag{A.24}$$

Based on (A.18), (A.19) and (A.21), the following can be obtained:

$$\begin{aligned} W^{t+1} - \overline{W^t} &= \mathbb{E}\{-(\nabla F_i(W) + \lambda * S\sigma\sqrt{n})\}\\ &= -((\nabla F(\overline{W^t}) - \mathbb{E}\{\lambda * \sigma \sqrt{n} \frac{W^t - W_i^t}{S}\})\\ &= -\frac{\nabla F}{1 + \frac{\lambda\sigma\sqrt{n}}{S}}, \end{aligned} \tag{A.25}$$

where $S = \mathbb{E}\{S_i\}$ and $n = \mathbb{E}\{n_i\}$.

By substituting (A.21) and (A.25) into (A.24), the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}}) - F(\overline{W^t})\} \leq \mathbb{E}\{\nabla F^T (-\frac{\nabla F}{\eta + \frac{\lambda\sigma\sqrt{n}}{S}} + N)\} + \frac{\rho}{2}\mathbb{E}\{|| - \frac{\nabla F}{1 + \frac{\lambda\sigma\sqrt{n}}{S}} + N||^2\}. \tag{A.26}$$

Then, using triangle inequation, the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}})-F(\overline{W^t})\}\leq(\frac{\rho l_1^2}{2}-l_1)||\nabla F||^2+(1-l_1\rho)||\nabla F||\mathbb{E}\{||N||\}+\frac{\rho}{2}\mathbb{E}\{||N||^2\}, \tag{A.27}$$

where

$$l_1=\frac{1}{1+\frac{\lambda\sigma\sqrt{n}}{S}}, \tag{A.28}$$

which completes the proof.

E Proof of Theorem 4

By subtracting $\mathbb{E}\{F(W^*)\}$ on both sides of (A.27), the following can be obtained:

$$\begin{aligned}\mathbb{E}\{F(\overline{W^{t+1}})-F(W^*)\}\leq\mathbb{E}\{F(\overline{W^t})-F(W^*)\}+(\frac{\rho l_1^2}{2}-l_1)||\nabla F||^2\\+(1-l_1\rho)||\nabla F||\mathbb{E}\{||N||\}+\frac{\rho}{2}\mathbb{E}\{||N||^2\}.\end{aligned} \tag{A.29}$$

It is known that $||\nabla F(W)||\leq\beta$ and with (3.9), $||W^{t+1}-\overline{W^t}||$ can be bounded as

$$\begin{aligned}||W^{t+1}-\overline{W^t}||&=||-\nabla F(\overline{W^t})-\mathbb{E}\{\lambda*\sigma\sqrt{n}\frac{W^t-W_i^t}{S}\}||\\&\leq-\beta+\lambda*\sigma\sqrt{n}\frac{||W^{t+1}-W^t||}{S}\\&\leq(\lambda*\sigma n^{\frac{3}{2}}-\beta).\end{aligned} \tag{A.30}$$

Then, with the noise generating method in Algorithm 3.2, the following can be obtained:

$$\mathbb{E}\{||N||\}\leq\frac{\sigma}{\sqrt{n}}(\lambda*\sigma n^{\frac{3}{2}}-\beta). \tag{A.31}$$

Then, by substituting (A.25), (A.26), (A.30) and (A.31) into (A.29) and with $F(W)-F(W^*)\leq\frac{1}{2\mu}||\nabla F(W)||^2$, the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}})-F(W^*)\}\leq l_3*\mathbb{E}\{F(\overline{W^t})-F(W^*)\}+\beta q_1(1-l_1\rho)+\frac{\rho}{2}q_1^2, \tag{A.32}$$

where

$$l_3 = (\mu\rho l_1^2 - 2l_1\mu + 1), \tag{A.33}$$

$$q_1 = \frac{\sigma}{\sqrt{n}}(\lambda * \sigma n^{\frac{3}{2}} - \beta). \tag{A.34}$$

Finally, since the noise is generated in the same and independent way, it is assumed the noise for all the communication rounds shares the same expected bound value. By repeating (A.32) for T communication rounds, the convergence can be upper bounded as

$$\mathbb{E}\{F(\overline{W^{t+1}}) - F(W^*)\} \leq l_3^T \mathbb{E}\{F(W^0) - F(W^*)\} + (\beta q_1(1 - l_1\rho) + \frac{\rho}{2}q_1^2) * \frac{(1 - l_3^T)}{1 - l_3}, \tag{A.35}$$

which completes the proof.

F Proof of Theorem 5

Based on (A.27) and (A.31), the following can be obtained:

$$F(\overline{W^{t+1}}) \leq F(\overline{W^t}) + (\frac{\rho l_1^2}{2} - l_1)||\nabla F||^2 + \beta q_1(1 - l_1\rho) + \frac{\rho}{2}q_1^2. \tag{A.36}$$

By taking T iterations of (A.36), the following can be obtained:

$$(l_1 - \frac{\rho l_1^2}{2})\sum||\nabla F||^2 \leq F(W^0) - F(W^*) + T\beta q_1(1 - l_1\rho) + T\frac{\rho}{2}q_1^2, \tag{A.37}$$

which implies:

$$\mathbb{E}\{||\nabla F||^2\} \leq \frac{F(W^0) - F(W^*)}{(l_1 - \frac{\rho l_1^2}{2}) * T} + \frac{\beta q_1(1 - l_1\rho) + \frac{\rho q_1^2}{2}}{(l_1 - \frac{\rho l_1^2}{2})}. \tag{A.38}$$

This completes the proof.

G Proof of Lemma 7

By considering the gradients clipping, noise addition in Section 4.2 and triangle inequality, the noisy gradients can be expressed as follows:

$$\begin{aligned} ||\overline{g(x^*)}|| &= ||g(x^*) * min(1, \frac{C}{||g(x^*)||}) + N(0, \sigma^2(\frac{C}{n})^2)|| \\ &\leq ||g(x^*) * min(1, \frac{C}{||g(x^*)||})|| + ||\sqrt{n} * \sigma \frac{C}{n}|| \\ &\leq ||g(x^*) * min(1, \frac{C}{||g(x^*)||})|| + ||g(x^*)|| * \frac{\sigma}{\sqrt{n}} * \frac{C}{||g(x^*)||}. \end{aligned} \tag{A.39}$$

Then, the following can be obtained:

$$||\overline{g(x^*)}|| \leq k1 * ||g(x^*)||, \tag{A.40}$$

where,

$$k_1 = min(1, \frac{C}{||g(x^*)||}) + \frac{\sigma}{\sqrt{n}} * \frac{C}{||g(x^*)||}. \tag{A.41}$$

This completes the proof.

H Proof of Corollary 2

Since $F(W)$ is L-Lipschitz smooth as in Assumption 2, and if the learning rate is set as $1/L$, the following can be obtained:

$$\begin{aligned} J(x^{t+1}) - J(x^t) &\leq \nabla(J(x^t))^T(W^{t+1} - W^t) + \frac{L}{2}||W^{t+1} - W^t||^2 \\ &\leq \nabla(J(x^t))^T(-\frac{1}{L} * k3 * \nabla J) + \frac{L}{2}|| - \frac{1}{L} * k3 * \nabla J||^2 \\ &\leq -\frac{k_3}{2L}||\nabla J||^2. \end{aligned} \tag{A.42}$$

Then, by computing T times iteratively and rearranging the formulas, the following can be obtained:

$$\sum \frac{k_3}{2L}||\nabla J||^2 \leq J(x^0) - J(x^*), \tag{A.43}$$

$$\mathbb{E}\{||\nabla J||^2\} \leq \frac{L(J(x^0) - J(x^*))}{2Tk3}, \tag{A.44}$$

which completes the proof.

I Proof of Lemma 8

By expanding the l_2-norm, the following can be obtained:

$$\begin{aligned}\bar{J} - Re &= \arg\min_{x\in(0,1)^n} ||g(x) - \overline{g(x^*)}|| \\ &= \arg\min_{x\in(0,1)^n} \sqrt{(g(x) - \overline{g(x^*)})^2} \\ &= \arg\min_{x\in(0,1)^n} \sqrt{g(x)^2 - 2 * \overline{g(x^*)} + \overline{g(x^*)}^2} \\ &= \arg\min_{x\in(0,1)^n} \sqrt{(g(x) - g(x^*))^2 - 2g(x)\overline{g(x^*)} + 2g(x)g(x^*) + \overline{g(x^*)}^2 - g(x^*)^2}.\end{aligned} \tag{A.45}$$

Then, by taking triangle inequalities, the following can be obtained:

$$\bar{J} - Re \leq \sqrt{(g(x) - g(x^*))^2} + \sqrt{\overline{g(x^*)}^2 - g(x^*)^2} + \sqrt{-2g(x)\overline{g(x^*)} + 2g(x)g(x^*)} \tag{A.46}$$

$$\bar{J} \leq J + \sqrt{2g(x)(g(x^*) - \overline{g(x^*)})} + \sqrt{\overline{g(x^*)}^2 - g(x^*)^2}. \tag{A.47}$$

Next, it is considered that if the derivatives of x are computed in $\bar{J}$, the term $g(x^*)$ is discarded. Therefore, for the purpose of simplifying the presentation, the final item of (A.47) is discarded. With Lemma 7, the following can be obtained:

$$\bar{J} = J + 2\sqrt{(1-k1) * g(x)g(x^*)}, \tag{A.48}$$

which completes the proof.

J Proof of Theorem 7

The following can be obtained:

$$\nabla J = g'(x)\frac{g(x) - g(x^*)}{||g(x) - g(x^*)||} + \nabla Re = (\beta_3 \pm 1)g'(x), \tag{A.49}$$

where

$$\beta_3 g'(x) = \nabla Re. \tag{A.50}$$

Then, the following can be obtained:

$$\nabla \bar{J} = (\beta_3 \pm 1)g'(x) + k_4 g'(x) = (1 + \frac{k_4}{(\beta_3 \pm 1)})\nabla J. \tag{A.51}$$

Since $J(x)$ is L-Lipschitz smooth as in Assumption 2, and the learning rate is set as $1/L$, the following can be obtained:

$$\begin{aligned} J(x^{t+1}) - J(x^t) &\leq \nabla(J(x^t))^T (W^{t+1} - W^t) + \frac{L}{2}||W^{t+1} - W^t||^2 \\ &\leq \nabla(J(x^t))^T (-\frac{1}{L} * (\nabla \bar{J})) + \frac{L}{2}|| - \frac{1}{L} * (\nabla \bar{J})||^2 \\ &\leq -\frac{||\nabla \bar{J}||^2}{2L} \\ &\leq -\frac{k_5^2 ||\nabla J||^2}{2L}, \end{aligned} \tag{A.52}$$

where,

$$k_5 = 1 + \frac{k_4}{(\beta_3 \pm 1)}. \tag{A.53}$$

Then, by taking T iterations of (A.52), the following can be obtained:

$$\sum \frac{k_5^2 ||\nabla J||^2}{2L} \leq J(x^0) - J(x^*), \tag{A.54}$$

$$\mathbb{E}\{||\nabla F||^2\} = \frac{L(F(x^0) - F(x^*))}{2Tk_5^2}, \tag{A.55}$$

which completes the proof.

K Proof of Theorem 8

To compute the DP bound of the proposed ADPDFL, it is considered the privacy amplification for sampling and anonymous mechanisms. Collecting training clients is considered a sampling procedure with a sampling rate r_{tcs} for two proposed frameworks in one aggregation. Then, the proposed framework without anonymous mechanism satisfies $(\alpha, 3.5 r_{tcs}^2 \alpha / \sigma^2)$-RDP based on Lemma 3.

Combining the results in [117, 118, 149], shuffling all the gradients before aggregation can reduce the ε by a factor of the number of total clients in an RDP-based centralized FL framework. In the proposed anonymous framework, it can seen that all the gradients are shuffled. Therefore, privacy amplification with a factor of m can be achieved. Meanwhile, since the framework is decentralized and the gradient receiver is also anonymous, the proposed framework can achieve an extra privacy amplification factor of m. Therefore, the proposed framework satisfies $(\alpha, 3.5r_{tcs}^2\alpha/m^2\sigma^2)$-RDP.

Then, by Lemma 4, the proposed fake-central framework satisfies $(\alpha, 3.5Tr_{tcs}^2\alpha/m^2\sigma^2)$. However, for the gossip-based framework, each client needs to send out m_e gradients. Therefore, it satisfies $(\alpha, 3.5Tm_e r_{tcs}^2\alpha/m^2\sigma^2)$-RDP. Then, it sets $s_1 = r_{tcs}^2 m_e$ for gossip-based and $s_1 = r_{tcs}^2$ for fake-central.

Next, in order to guarantee (ε, δ)-DP, the following can be obtained

$$\frac{3.5Ts_1\alpha}{m^2\sigma^2} + \frac{log(1/\delta)}{\alpha - 1} = \varepsilon, \tag{A.56}$$

where $\alpha = 1 + 2log(1/\delta)/\varepsilon$, and the following can be obtained:

$$\frac{3.5Ts_1}{m^2\sigma^2}(1 + \frac{2log(\frac{1}{\delta})}{\varepsilon}) + \frac{log(1/\delta)}{\frac{2log(\frac{1}{\delta})}{\varepsilon}} = \varepsilon, \tag{A.57}$$

$$\frac{3.5Ts_1}{m^2\sigma^2}(1 + \frac{2log(\frac{1}{\delta})}{\varepsilon}) = \frac{\varepsilon}{2}, \tag{A.58}$$

$$\sigma^2 = \frac{7s_1T(\varepsilon + 2log(1/\delta))}{m^2\varepsilon^2}, \tag{A.59}$$

where $s_1 = r_{tcs}^2 m_e$ for gossip-based framework, and $s_1 = r_{tcs}^2$ for fake-central framework.

Meanwhile, if $\alpha = 1 + 2log(1/\delta)/\varepsilon$ and $\alpha \geq 2$, the following can be obtained:

$$\varepsilon \leq 2log(\frac{1}{\delta}). \tag{A.60}$$

This completes the proof.

L Proof of Theorem 9

Considering at tth round FL of gossip-based in Algorithm 5.1 and 5.2 and one client's model is randomly chosen to be the global model, the following can be obtained:

$$\overline{W_i^{t+1}} = \overline{W_i^t} - \sum_{j \in S_i^t} p_i(\nabla F_i(W_j^t) + N_j^t), \tag{A.61}$$

where S is the the set of received gradients at client i at tth round, p_i is the weight of the aggregation.

Since $F(w)$ is L-Lipschitz smooth, the following can be obtained:

$$F(\overline{W^{t+1}}) \leq F(\overline{W^t}) + < \nabla F(\overline{W^t}), (\overline{W^{t+1}} - \overline{W^t}) > + \frac{L}{2} ||\overline{W^{t+1}} - \overline{W^t}||^2. \tag{A.62}$$

Meanwhile, the following lemma is needed.

Lemma 9 *For gossip-based learning, the aggregated model parameters W_i^{t+1} in client i have the following properties [150, 151],*

$$\mathbb{E}||W_i^{t+1} - W_i^t||^2 \leq \frac{1}{1-\rho_2} ||W^{t+1} - W^t||^2, \tag{A.63}$$

where W^{t+1} is the assumed aggregation model parameters of all clients and ρ_2 is the second largest eigenvalue of the graph matrix.

Based on (A.61), Lemma 9 and $2 * < a, b >= ||a||^2 + ||b||^2 - ||a-b||^2$, the following can be obtained:

$$\begin{aligned}
&F(\overline{W^{t+1}}) \leq F(\overline{W^t}) + < \nabla F(\overline{W^t}), -(\sum_{j=m} p_j(\nabla F(W_j^t) + N_j^t)) > + \frac{L}{2} || \sum_{j=m} p_j(\nabla F(W_j^t) + N_j^t)||^2 \\
&F(\overline{W^{t+1}}) \leq F(\overline{W^t}) + < \nabla F(\overline{W^t}), - \sum_{j=m} p_j(\nabla F(W_j^t) > + \frac{L}{2} || \sum_{j=m} p_j(\nabla F(W_j^t) + N_j^t)||^2 \\
&F(\overline{W^{t+1}}) \leq F(\overline{W^t}) - \frac{1}{2} ||\nabla F(\overline{W^t})||^2 - \frac{1}{2} ||(1-\rho) \sum_{j \in S_i^t} p_j(\nabla F(W_j^t)||^2 \\
&\qquad + \frac{1}{2} ||\nabla F(\overline{W^t}) - \sum_{j=m} p_j(\nabla F(W_j^t)||^2 \\
&\qquad + \frac{L(1-\rho)}{2} (|| \sum_{j \in S_i^t} p_j(\nabla F(W_j^t))||^2 + || \sum_{j \in S_i^t} p_j(N_j^t)||^2),
\end{aligned} \tag{A.64}$$

where N_i^t is the added noise on the client i at round t, S_i^t is the aggregation set in client i.

The expectation of both sides of the above formula is taken to have

$$\begin{aligned}\mathbb{E}\{F(\overline{W^{t+1}})-F(\overline{W^{t}})\} \leq -\mathbb{E}\{\frac{1-\rho}{2}\underbrace{||\sum_{j\in S_i^t}p_j(\nabla F(W_j^t)||^2-\frac{1}{2}||\nabla F(\overline{W^t})||^2}_{A1} \\ +\frac{1}{2}||\nabla F(\overline{W^t})-\sum_{j=m}p_j(\nabla F(W_j^t)||^2 \\ +\frac{L(1-\rho)}{2}(||\sum_{j\in S_i^t}p_j(\nabla F(W_j^t))||^2+||\sum_{j\in S_i^t}p_j(N_j^t)||^2)\}.\end{aligned} \tag{A.65}$$

Based on Assumption 4, the term $A1$ is bounded as follows:

$$\begin{aligned}||\sum_{j\in S_i^t}p_j(\nabla F(W_j^t)||^2 &\leq \frac{1}{m_{tcs}m_e}\sum_{j\in S^t}||\nabla F_j(W_j^t)||^2 \\ &+\frac{m_e-1}{m_{tcs}m_e(m_{tcs}-1)}\sum_{\substack{j,j'\in S^t\\ j\neq j'}}(\nabla F_j(W_j^t))^T\nabla F_j'(W_j'^t) \\ &\leq (\frac{1}{m_{tcs}m_e}-\frac{m_e-1}{m_{tcs}m_e(m_{tcs}-1)})\sum_{j\in S^t}||\nabla F_j(W_j^t)||^2 \\ &+\frac{m_e-1}{m_{tcs}m_e(m_{tcs}-1)}(\sum_{j\in S^t}\nabla F_j(W_j^t))^2 \\ &\leq \frac{m_{tcs}(m_e-1)}{m_e(m_{tcs}-1)}||\nabla F(W^t)||^2+\frac{m_{tcs}-m_e}{m_{tcs}m_e(m_{tcs}-1)}\sum_{j\in S^t}||\nabla F_j(W_j^t)||^2 \\ &\leq \frac{m_{tcs}(m_e-1)}{m_e(m_{tcs}-1)}||\nabla F(W^t)||^2+\frac{m_{tcs}-m_e}{m_{tcs}m_e(m_{tcs}-1)}\sum_{j\in S^t}||\nabla F(W^t)||^2 \\ &+\frac{m_{tcs}-m_e}{m_{tcs}m_e(m_{tcs}-1)}\sum_{j\in S^t}||\nabla F_j(W_j^t)-\nabla F(W^t)||^2 \\ &\leq ||\nabla F(W^t)||^2+\frac{m_{tcs}-m_e}{m_{tcs}m_e(m_{tcs}-1)}\sum_{j\in S^t}||\nabla F_j(W_j^t)-\nabla F(W^t)||^2 \\ &\leq ||\nabla F(W^t)||^2+\frac{\zeta^2(m_{tcs}-m_e)}{m_e(m_{tcs}-1)}.\end{aligned} \tag{A.66}$$

Then, with Assumption 4 and Theorem 8 and by substituting (A.66) into (A.65), the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}})-F(\overline{W^t})\} \leq \frac{(L-1)(1-\rho)-1}{2}||\nabla F(W^t)||^2 + \frac{\zeta^2(L-1)(1-\rho)(m_{tcs}-m_e)}{2m_e(m_{tcs}-1)} + \frac{\zeta^2}{2} + \frac{2LC^2\sigma^2(1-\rho)}{n^2 m_e}. \tag{A.67}$$

By rearranging (A.67) and recursive computation, the following can be obtained:

$$-\frac{(L-1)(1-\rho)-1}{2}\sum_T ||\nabla F(W^t)||^2 \leq F(\overline{W^0})-F(\overline{W^*}) - \frac{T\zeta^2}{2} + \frac{2TLC^2\sigma^2(1-\rho)}{n^2 m_e} + \frac{T\zeta^2(L-1)(1-\rho)(m_{tcs}-m_e)}{2m_e(m_{tcs}-1)}. \tag{A.68}$$

Finally, the following can be obtained:

$$\mathbb{E}\{||\nabla F(W^t)||^2\} \leq \frac{2(F(\overline{W^0})-F(\overline{W^*}))}{T(\rho+\rho L-L-2)} - \frac{\zeta^2}{(\rho+\rho L-L+2)} + \frac{\zeta^2(L+1)(1-\rho)(m_{tcs}-m_e)}{m_e(\rho+\rho L-L+2)(m_{tcs}-1)} + \frac{4LC^2\sigma^2(1-\rho)}{n^2 m_e(\rho+\rho L-L-2)}. \tag{A.69}$$

On the other hand, for the fake-central framework, the following can be obtained:

$$\begin{aligned} F(\overline{W^{t+1}}) \leq{}& F(\overline{W^t}) - \frac{1}{2}||\nabla F(\overline{W^t})||^2 - \frac{1}{2}||\sum_{j\in S_i^t} p_j(\nabla F(W_j^t)||^2 \\ &+ \frac{1}{2}||\nabla F(\overline{W^t}) - \sum_{j=m} p_j(\nabla F(W_j^t)||^2 \\ &+ \frac{L}{2}||\sum_{j\in S_i^t} p_j(\nabla F(W_j^t))||^2 + \frac{L}{2}||\sum_{j\in S_i^t} p_j(N_j^t)||^2. \end{aligned} \tag{A.70}$$

By following the similar derivation, it can substitute m_e and m_t to m_t and m in (A.66) to have

$$||\sum_{j\in S_i^t} p_j(\nabla F(W_j^t)||^2 \leq ||\nabla F(W^t)||^2 + \frac{\zeta^2(m-m_{tcs})}{m_{tcs}(m-1)}. \tag{A.71}$$

By substituting (A.71) into (A.70), the following can be obtained:

$$\mathbb{E}\{F(\overline{W^{t+1}}) - F(\overline{W^t})\} \leq \frac{L-2}{2}||\nabla F(W^t)||^2 + \frac{\zeta^2(L-1)(m-m_{tcs})}{2m_{tcs}(m-1)} - \frac{\zeta^2}{2} + \frac{2LC^2\sigma^2}{n^2 m_{tcs}}. \tag{A.72}$$

By rearranging (A.72) and recursive computation, the following can be obtained:

$$-\frac{L-2}{2}\sum_{T}||\nabla F(W^t)||^2 \leq F(\overline{W^0}) - F(\overline{W^*}) - \frac{T\zeta^2}{2} + \frac{T\zeta^2(L-1)(m-m_{tcs})}{2m_{tcs}(m-1)} + \frac{2TLC^2\sigma^2}{n^2 m_{tcs}}. \tag{A.73}$$

Finally, the following can be obtained:

$$\mathbb{E}\{||\nabla F(W^t)||^2\} \leq \frac{2(F(\overline{W^0}) - F(\overline{W^*}))}{T(2-L)} - \frac{\zeta^2}{(2-L)} + \frac{\zeta^2(L+1)(m-m_{tcs})}{m_{tcs}(2-L)(m-1)} + \frac{4LC^2\sigma^2}{n^2 m_{tcs}(2-L)}. \tag{A.74}$$

This completes the proof.

Bibliography

[1] L. Da, W. He, and S. Li, “Internet of things in industries: A survey,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[2] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] Y. B. Zikria, R. Ali, M. K. Afzal, and S. W. Kim, “Next-generation internet of things (iot): Opportunities, challenges, and solutions,” *Sensors*, vol. 21, no. 4, p. 1174, 2021.

[4] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, “Federated learning for internet of things: A comprehensive survey,” *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[5] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein, “Nnu-net: a self-configuring method for deep learning-based biomedical image segmentation,” *Nature Methods*, vol. 18, no. 2, pp. 203–211, 2021.

[6] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[7] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–38, 2019.

[8] D. W. Otter, J. R. Medina, and J. K. Kalita, “A survey of the usages of deep learning for natural language processing,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 604–624, 2020.

[9] 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.

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

[11] C. S. Legislature, “California consumer privacy act (ccpa),” 2018. [Online]. Available: https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5

[12] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, “A survey on distributed machine learning,” *ACM Computing Surveys*, vol. 53, no. 2, pp. 1–33, 2020.

[13] Google, “Federated learning: Collaborative machine learning without centralized training data,” 2017. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[14] 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*, vol. 54, 2017, pp. 1273–1282.

[15] T. K. Dang, X. Lan, J. Weng, and M. Feng, “Federated learning for electronic health records,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 5, pp. 1–17, 2022.

[16] 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, pp. 1–19, 2021.

[17] Y. Li, X. Tao, X. Zhang, J. Liu, and J. Xu, “Privacy-preserved federated learning for autonomous driving,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 8423–8434, 2021.

[18] Z. Zheng, Y. Zhou, Y. Sun, Z. Wang, B. Liu, and K. Li, “Applications of federated learning in smart cities: recent advances, taxonomy, and open challenges,” *Connection Science*, vol. 34, no. 1, pp. 1–28, 2022.

[19] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, “A review of applications in federated learning,” *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020.

[20] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, “A survey on federated learning,” *Knowledge-Based Systems*, vol. 216, p. 106775, 2021.

[21] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, “Distributed federated learning for ultra-reliable low-latency vehicular communications,” *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2019.

[22] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan, T. Van Overveldt, D. Petrou, D. Ramage, and J. Roselander, “Towards federated learning at scale: System design,” in *Proceedings of Machine Learning and Systems*, vol. 1, 2019, pp. 374–388.

[23] D. Liu, L. Bai, T. Yu, and A. Zhang, “Towards method of horizontal federated learning: A survey,” in *2022 8th International Conference on Big Data and Information Analytics*, 2022, pp. 259–266.

[24] Y. Liu, Y. Kang, T. Zou, Y. Pu, Y. He, X. Ye, Y. Ouyang, Y.-Q. Zhang, and Q. Yang, “Vertical federated learning: Concepts, advances, and challenges,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 7, pp. 3615–3634, 2024.

[25] K. Wei, J. Li, C. Ma, M. Ding, S. Wei, F. Wu, G. Chen, and T. Ranbaduge, “Vertical federated learning: Challenges, methodologies and experiments,” *arXiv preprint arXiv:2202.04309*, 2022.

[26] S. Saha and T. Ahmad, “Federated transfer learning: concept and applications,” *Intelligenza Artificiale*, vol. 15, no. 1, pp. 35–44, 2021.

[27] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, “Fedhealth: A federated transfer learning framework for wearable healthcare,” *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[28] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, pp. 1–40, 2016.

[29] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar, “Fully decentralized federated learning,” in *Third Workshop on Bayesian Deep Learning (NeurIPS)*, vol. 2, 2018.

[30] C. Che, X. Li, C. Chen, X. He, and Z. Zheng, "A decentralized federated learning framework via committee mechanism with convergence guarantee," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4783–4800, 2022.

[31] 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.

[32] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[33] J. Wen, Z. Zhang, Y. Lan, Z. Cui, J. Cai, and W. Zhang, "A survey on federated learning: challenges and applications," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 2, pp. 513–535, 2023.

[34] K. Zhang, X. Song, C. Zhang, and S. Yu, "Challenges and future directions of secure federated learning: A survey," *Frontiers of Computer Science*, vol. 16, pp. 1–8, 2022.

[35] D. Zeng, S. Liang, X. Hu, H. Wang, and Z. Xu, "Fedlab: A flexible federated learning framework," *Journal of Machine Learning Research*, vol. 24, no. 100, pp. 1–7, 2023.

[36] H. Zhu, J. Xu, S. Liu, and Y. Jin, "Federated learning on non-iid data: A survey," *Neurocomputing*, vol. 465, pp. 371–390, 2021.

[37] M. Ye, X. Fang, B. Du, P. C. Yuen, and D. Tao, "Heterogeneous federated learning: State-of-the-art and research challenges," *ACM Computing Surveys*, vol. 56, no. 3, pp. 1–44, 2023.

[38] 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.

[39] 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.

[40] 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*, 2020, pp. 5132–5143.

[41] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[42] C. Ma, J. Li, L. Shi, M. Ding, T. Wang, Z. Han, and H. V. Poor, "When federated learning meets blockchain: A new distributed learning paradigm," *IEEE Computational Intelligence Magazine*, vol. 17, no. 3, pp. 26–33, 2022.

[43] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4641–4654, 2020.

[44] E. T. Martínez Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Communications Surveys Tutorials*, vol. 25, no. 4, pp. 2983–3013, 2023.

[45] L. Yuan, Z. Wang, L. Sun, P. S. Yu, and C. G. Brinton, "Decentralized federated learning: A survey and perspective," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[46] H. Yin, P. Molchanov, J. M. Alvarez, Z. Li, A. Mallya, D. Hoiem, N. K. Jha, and J. Kautz, "Dreaming to distill: Data-free knowledge transfer via deepinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8715–8724.

[47] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 337–16 346.

[48] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 937–16 947, 2020.

[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[51] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–36, 2021.

[52] 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.

[53] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[54] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[55] J. Fu, Y. Hong, X. Ling, L. Wang, X. Ran, Z. Sun, W. H. Wang, Z. Chen, and Y. Cao, "Differentially private federated learning: A systematic review," *arXiv preprint arXiv:2405.08299*, 2024.

[56] 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.

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

[58] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3388–3401, 2022.

[59] L. Sun, J. Qian, and X. Chen, "Ldp-fl: Practical private aggregation in federated learning with local differential privacy," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

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

[61] 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.

[62] 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.

[63] S. Weng, L. Zhang, X. Zhang, and M. A. Imran, “Faster convergence on differential privacy-based federated learning,” *IEEE Internet of Things Journal*, vol. 11, no. 12, pp. 22 578–22 589, 2024.

[64] S. Weng, Y. Gou, L. Zhang, and M. A. Imran, “Evaluating privacy loss in differential privacy based federated learning,” *under review in Future Generation Computer Systems*.

[65] S. Weng, L. Zhang, Y. Gou, T. Nguyen, and M. A. Imran, “Anonymous differential privacy-based decentralized federated learning with performance analysis,” *under review in IEEE Transactions on Dependable and Secure Computing*.

[66] J. Zou, Y. Han, and S.-S. So, “Overview of artificial neural networks,” *Artificial Neural Networks: Methods and Spplications*, pp. 14–22, 2009.

[67] T. L. Fine, *Feedforward Neural Network Methodology*. Springer Science & Business Media, 2006.

[68] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[69] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[70] S. Ruder, “An overview of gradient descent optimization algorithms,” *arXiv preprint arXiv:1609.04747*, 2016.

[71] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[72] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[73] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5188–5196.

[74] A. Rényi, "On measures of entropy and information," in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, volume 1: contributions to the theory of statistics*, vol. 4, 1961, pp. 547–562.

[75] I. Mironov, "Rényi differential privacy," in *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, 2017.

[76] R. Hu, Y. Guo, and Y. Gong, "Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy," *IEEE Transactions on Mobile Computing*, vol. 23, no. 8, pp. 8242–8255, 2024.

[77] L. Wang, B. Jayaraman, D. Evans, and Q. Gu, "Efficient privacy-preserving stochastic nonconvex optimization," in *Uncertainty in Artificial Intelligence*, 2023, pp. 2203–2213.

[78] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[79] H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, "Federated learning with matched averaging," in *International Conference on Learning Representations*, 2020.

[80] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 713–10 722.

[81] D. A. E. Acar, Y. Zhao, R. Matas, M. Mattina, P. Whatmough, and V. Saligrama, "Federated learning based on dynamic regularization," in *International Conference on Learning Representations*, 2021.

[82] 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.

[83] 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.

[84] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, “Adaptive federated optimization,” in *International Conference on Learning Representations*, 2021.

[85] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, “Towards personalized federated learning,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 12, pp. 9587–9603, 2022.

[86] A. Fallah, A. Mokhtari, and A. Ozdaglar, “Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 3557–3568, 2020.

[87] Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song, “The secret revealer: Generative model-inversion attacks against deep neural networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 253–261.

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

[89] 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.

[90] L. Zhu, Z. Liu, and S. Han, “Deep leakage from gradients,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[91] B. Zhao, K. R. Mopuri, and H. Bilen, “Idlg: Improved deep leakage from gradients,” *arXiv preprint arXiv:2001.02610*, 2020.

[92] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, “Evaluating gradient inversion attacks and defenses in federated learning,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 7232–7241, 2021.

[93] 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.

[94] X. Yang, Y. Feng, W. Fang, J. Shao, X. Tang, S.-T. Xia, and R. Lu, “An accuracy-lossless perturbation method for defending privacy attacks in federated learning,” in *Proceedings of the ACM Web Conference*, 2022, pp. 732–742.

[95] 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.

[96] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, “Learning differentially private recurrent language models,” in *International Conference on Learning Representations*, 2018.

[97] S. Gade and N. H. Vaidya, “Privacy-preserving distributed learning via obfuscated stochastic gradients,” in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 184–191.

[98] A. Triastcyn and B. Faltings, “Federated learning with bayesian differential privacy,” in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 2587–2596.

[99] H. Zhou, G. Yang, H. Dai, and G. Liu, “Pflf: Privacy-preserving federated learning framework for edge computing,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 1905–1918, 2022.

[100] D. Bernau, J. Robl, P. W. Grassal, S. Schneider, and F. Kerschbaum, “Comparing local and central differential privacy using membership inference attacks,” in *IFIP Annual Conference on Data and Applications Security and Privacy*, 2021, pp. 22–42.

[101] 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.

[102] X. Zhang, H. Gu, L. Fan, K. Chen, and Q. Yang, “No free lunch theorem for security and utility in federated learning,” *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 1, pp. 1–35, 2022.

[103] H. Lu, C. Liu, T. He, S. Wang, and K. S. Chan, “Sharing models or coresets: A study based on membership inference attack,” *arXiv preprint arXiv:2007.02977*, 2020.

[104] W. Wei, L. Liu, M. Loper, K.-H. Chow, M. E. Gursoy, S. Truex, and Y. Wu, “A framework for evaluating client privacy leakages in federated learning,” in *25th European Symposium on Research in Computer Security*, 2020, pp. 545–566.

[105] R. Hu, Y. Gong, and Y. Guo, “Federated learning with sparsification-amplified privacy and adaptive optimization,” in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 2021.

[106] K. Wei, J. Li, C. Ma, M. Ding, W. Chen, J. Wu, M. Tao, and H. V. Poor, “Personalized federated learning with differential privacy and convergence guarantee,” *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 4488–4503, 2023.

[107] A. M. Girgis, D. Data, S. Diggavi, A. T. Suresh, and P. Kairouz, “On the rényi differential privacy of the shuffle model,” in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, p. 2321–2341.

[108] 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.

[109] B. Niu, Y. Chen, B. Wang, Z. Wang, F. Li, and J. Cao, “Adapdp: Adaptive personalized differential privacy,” in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[110] A. Golatkar, A. Achille, Y.-X. Wang, A. Roth, M. Kearns, and S. Soatto, “Mixed differential privacy in computer vision,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 8376–8386.

[111] Y. Hu, Z. Tan, X. Li, J. Wang *et al.*, "Adaptive clipping bound of deep learning with differential privacy," in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2021, pp. 428–435.

[112] L. Han, D. Fan, J. Liu, and W. Du, "Federated learning differential privacy preservation method based on differentiated noise addition," in *2023 8th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, 2023, pp. 285–289.

[113] 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.

[114] X. Yang, W. Huang, and M. Ye, "Dynamic personalized federated learning with adaptive differential privacy," *Advances in Neural Information Processing Systems*, vol. 36, pp. 72 181–72 192, 2023.

[115] Y. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled reacute;nyi differential privacy and analytical moments accountant," in *The 22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 1226–1235.

[116] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[117] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, "Shuffled model of differential privacy in federated learning," in *International Conference on Artificial Intelligence and Statistics*, 2021, pp. 2521–2529.

[118] U. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2019, pp. 2468–2479.

[119] 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.

[120] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 13–23.

[121] 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.

[122] 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.

[123] R. Liu, Y. Cao, M. Yoshikawa, and H. Chen, "Fedsel: Federated sgd under local differential privacy with top-k dimension selection," in *Database Systems for Advanced Applications: 25th International Conference*, 2020, pp. 485–501.

[124] 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.

[125] 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.

[126] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[127] J. Zhang, T. He, S. Sra, and A. Jadbabaie, "Why gradient clipping accelerates training: A theoretical justification for adaptivity," in *International Conference on Learning Representations*, 2019.

[128] X. Zhang, X. Chen, M. Hong, Z. S. Wu, and J. Yi, "Understanding clipping for federated learning: Convergence and client-level differential privacy," in *International Conference on Machine Learning*, 2022.

[129] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009. [Online]. Available: https://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf

[130] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[131] K. Simonyan, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[132] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6848–6856.

[133] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, "Repvgg: Making vgg-style convnets great again," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 733–13 742.

[134] X. Wu, Y. Zhang, M. Shi, P. Li, R. Li, and N. N. Xiong, "An adaptive federated learning scheme with differential privacy preserving," *Future Generation Computer Systems*, vol. 127, pp. 362–372, 2022.

[135] Y. Gou, S. Weng, M. A. Imran, and L. Zhang, "Voting consensus-based decentralized federated learning," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16 267–16 278, 2024.

[136] C. Briggs, Z. Fan, and P. Andras, "Federated learning with hierarchical clustering of local updates to improve training on non-iid data," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–9.

[137] Z. Tang, S. Shi, B. Li, and X. Chu, "Gossipfl: A decentralized federated learning framework with sparsified and adaptive communication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 34, no. 3, pp. 909–922, 2023.

[138] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Network*, vol. 35, no. 1, pp. 234–241, 2021.

[139] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2019.

[140] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "Flchain: A blockchain for auditable federated learning with trust and incentive," in *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, 2019, pp. 151–159.

[141] Y. Chen, Y. Su, M. Zhang, H. Chai, Y. Wei, and S. Yu, "Fedtor: An anonymous framework of federated learning in internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 19, pp. 18 620–18 631, 2022.

[142] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.

[143] K. Pan, Y.-S. Ong, M. Gong, H. Li, A. K. Qin, and Y. Gao, "Differential privacy in deep learning: A literature survey," *Neurocomputing*, p. 127663, 2024.

[144] A. E. Ouadrhiri and A. Abdelhadi, "Differential privacy for deep and federated learning: A survey," *IEEE Access*, vol. 10, pp. 22 359–22 380, 2022.

[145] T. H. Rafi, F. A. Noor, T. Hussain, and D.-K. Chae, "Fairness and privacy preserving in federated learning: A survey," *Information Fusion*, vol. 105, p. 102198, 2024.

[146] H. Chen, T. Zhu, T. Zhang, W. Zhou, and P. S. Yu, "Privacy and fairness in federated learning: on the perspective of tradeoff," *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–37, 2023.

[147] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp. 3347–3366, 2021.

[148] X. Ma, X. Sun, Y. Wu, Z. Liu, X. Chen, and C. Dong, "Differentially private byzantine-robust federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 3690–3701, 2022.

[149] K. Talwar, S. Wang, A. McMillan, V. Jina, V. Feldman, P. Bansal, B. Basile, A. Cahill, Y. S. Chan, M. Chatzidakis *et al.*, "Samplable anonymous aggregation for private federated data analysis," *arXiv preprint arXiv:2307.15017*, 2023.

[150] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[151] W. Liu, L. Chen, and W. Zhang, "Decentralized federated learning: Balancing communication and computing costs," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 8, pp. 131–143, 2022.