Differentially Private Federated Learning on Non-iid Data: Convergence Analysis and Adaptive Optimization

Lin Chen, Xiaofeng Ding, *Member, IEEE*, Zhifeng Bao, Pan Zhou, *Senior Member, IEEE*, and Hai Jin, *Fellow, IEEE*

***Abstract*—Federated learning (FL) has attracted increasing attention in recent years due to its data privacy preservation and great applicability to large-scale user scenarios. However, when FL faces numerous clients, it is inevitable to emerge the non-independent and identically distributed (non-iid) data between clients, which brings an enormous challenge for model training and performance analysis like convergence. Besides, due to the non-iid data, the participating clients of FL tend to be extremely heterogeneous so the number of samplings among clients causes a sampling variance problem, which induces a huge variation in convergence. More importantly, although FL can foster privacy security via locally retaining the training data, if local data is secret and sensitive, FL should have more powerful privacy protection to resist the cloud server or third party to infer private information from shared models or intermediate gradients. Facing the non-iid and privacy challenges, we propose a differential privacy (DP) based non-iid FL algorithm called DPNFL to jointly tackle these two issues. Specifically, motivated by the DP and its variants, we are the first to adopt the truncated concentrated differential privacy technique under the FL scenario to more tightly track end-to-end privacy loss, while requiring less noise injection for the same level of DP. To avoid the sampling variance problem, we enable the server to sample the partial clients uniformly without replacement, which also guarantees unbiased sampling. To further improve the algorithm performance, we also propose an adaptive version of DPNFL named AdDPNFL, which adopts the adaptive optimization on the server-side to simultaneously alleviate the impact of non-iid data and DP noise on model utility. Finally, we perform extensive experiments to validate the effectiveness and superiority of our algorithms.**

Manuscript received 5 July 2022; revised 30 August 2023; accepted 1 March 2024. Date of publication 19 March 2024; date of current version 7 August 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62172179 and in part by the CCF-Huawei Populus euphratica Innovation Research Funding under Grant CCF-HuaweiDB202309. Recommended for acceptance by X. Xiao. *(Corresponding authors: Xiaofeng Ding; Pan Zhou.)*

Lin Chen, Xiaofeng Ding, and Hai Jin are with the National Engineering Research Center for Big Data Technology and System Lab, Services Computing Technology and System Lab, Cluster and Grid Computing Lab, Hubei Engineering Research Center on Big Data Security, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: chenl@hust.edu.cn; xfding@hust.edu.cn; hjin@hust.edu.cn).

Zhifeng Bao is with the School of Computing Technolgies, RMIT University, Melbourne, VIC 3000, Australia (e-mail: zhifeng.bao@rmit.edu.au).

Pan Zhou is with the Hubei Engineering Research Center on Big Data Security, School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: panzhou@hust.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2024.3379001, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2024.3379001

***Index Terms*—Federate learning, non-iid data, differential privacy, convergence analysis.**

I. Introduction

FEDERATED learning (FL) has been an emerging distributed learning prototype that empowers numerous clients to cooperatively learn the joint model with the coordination of the cloud server, free of sharing the local data [1]. By ensuring the localized data, FL has superiority over the conventional centralized learning framework on privacy and communication efficiency. Moreover, different from general distributed learning, federated learning is eminently suitable for the large-scale user scenario, where the number of user equipment like mobile phones and IoT devices is huge. Meanwhile, rather than adopting one-step local update in various distributed learning settings, FL has a unique feature that enables all the clients to execute many local updates and then the server periodically aggregates these updates to reduce the communication cost. In view of these merits, FL has increasingly become a hot topic in the academic community and has been applied to many realistic areas including next-word prediction, financial data mining, and medical record analysis [2], [3].

However, when FL faces numerous clients, it is inevitable to emerge the non-independent and identically distributed (non-iid) data among clients. Specifically, since FL collects the local training data from various sorts of clients, the data size of the participating clients is notably differing and the probability distribution of data labels in the different clients is dissimilar. Then, the local dataset chosen at random might not express the true data distribution from the global perspective, which possibly brings biases to the update of the global model and causes the difference among the local models learned from non-iid data. Therefore, using these diverging models to aggregate will degrade convergence speed and model utility. On the whole, the performance reduction and communication burden of FL could turn unacceptable when the data distribution is non-iid.

Considering that the widely-recognized statistical theories are mostly on the premise of the iid data, there merely exists a handful of theoretical analyses of convergence on the non-iid scene, though the non-iid issue is challengeable for FL. Except for the theoretical value, studying the non-iid problem of FL is also vital for practical applications since non-iid data is ubiquitous in realistic FL environments. As an example, in the next word prediction task on a smartphone keyboard, the diverse typing habit of various users can generate non-iid data during the cross-client training. Besides, the cross-silo medical data sets are inherently non-iid, because of the characteristics like dissimilar data collection rules and diverse local demographic data. Based on the above requirements, we aim to investigate the convergence problem of FL on non-iid data.

Due to the non-iid data, the participating client devices of FL tend to be extremely heterogeneous, which brings a new challenge to the client sampling problem in FL. Since it is unrealistic for the full client sampling to select all clients per communication round, most FL algorithms adopt the partial client sampling where the server simultaneously samples partial clients and uses the present global model to replace the local model in unsampled clients. However, some studies [4], [5] have demonstrated that such a method is biased and converges to a sub-optimal minimum. Besides, this method typically has a slow convergence speed because of the abrasion brought from the unsampled clients. To handle these deficiencies, some works [6], [7], [8] allow the server to select clients in a multinomial distribution (MD) with replacement, which is usually called MD sampling, to achieve unbiased sampling. However, since the server selects clients with replacement, the number of samplings in the different clients causes a variance, which likely produces a huge variation in convergence performance, specifically for the non-iid FL. Furthermore, when the number of sampled clients is large, the MD sampling scheme has a high time complexity which results in complicated computation or even intractability. To avoid these issues in the MD sampling, we adopt the uniform sampling without replacement, where the server collects the first r updated models from the clients and then creates the new global model via these models. Without replacement procedure, uniform sampling has the advantage over time complexity and can sample the client with an equal number of times to avoid the variance problem.

On the other hand, although FL can foster the privacy security via locally retaining the training data, if local data includes susceptible or private information, FL should possess powerful privacy protection to guarantee that the cloud (or malicious third party) can not precisely restore such information based on the model update sharing among the clients. But existing adversary models [9], [10] have shown that privacy implications appear during sharing respective model updates among clients. Moreover, due to the open computation network structure and intensive cooperation between clients, it also unavoidably gives chances to semi-honest servers or clients for inferring the confidential information from inner gradient or shared model in other benignant clients. To eliminate these concerns, some works propose several privacy-preserving frameworks like secure multi-party computation, homomorphic encryption and shuffle with anonymity to preserve the privacy in FL, but these methods largely cost communication resources. Differential privacy (DP) [11] has been regarded as a de-facto standard because it exhibits rigorously privacy-preserving capability and highly effective flexibility while performing data analysis. Common DP techniques including Gaussian and Laplacian mechanisms cautiously inject the well-tuned noise into the algorithm output to obtain DP. However, since the model scale of the deep neural network in FL is usually large and positively related to the scale of auxiliary noise, it is hard to avoid adding overmuch noise to the output, which leads to the severe deterioration in model performance and brings unprecedented challenges for DP-based FL to balance the privacy-utility trade-off.

Facing the above two significant challenges, most research individually considers the non-iid-ness and DP, since the conjoint analysis of these two issues seems intractable. Concretely, non-iid data requires more communication rounds to achieve the desired performance but more rounds mean more noise addition which degrades the utility. To our best knowledge, only a few papers [12], [13], [14] simultaneously investigate the non-iid and DP problem. These works either utilize the full client sampling or MD client sampling that ignores the impact of the heterogeneous client due to non-iid data. Moreover, to achieve DP, they either adopt the conventional moments accountant technique or Renyi differential privacy, which are both less tight than the truncated concentrated differential privacy (tCDP) used in our work.

To sum up, in this paper, we focus on the performance analysis of FL on non-iid data while preserving DP. To be more practical, our work only requires the server to be trust-but-curious rather than fully trusted. In this situation, the central server can potentially observe all client updates and the third-party server can silently observe the local model. For such a FL scenario, we propose a differentially private algorithm named DPNFL to protect the user privacy, where we inject the Gaussian noise into the local gradient and exploit the post-processing property of DP to simultaneously achieve the DP of the intermediate model update and final local model. To cope with non-iid data, we adopt the partial client sampling without replacement under the non-iid FL setting to mitigate the impact of the heterogeneous client. To further improve the algorithm performance, we propose an adaptive version of DPNFL named AdDPNFL which adopts the adaptive optimization on the server-side to simultaneously alleviate the impact of non-iid data and DP noise on model utility. For privacy analysis, we adopt the advanced tCDP to tightly record the end-to-end privacy loss and thus achieve less noise injection under the same DP guarantees. For the convergence analysis, we utilize two kinds of metrics to measure the non-iid degree regarding the strongly convex and non-convex objective functions and give the corresponding convergence bounds. We summarize the main contributions as follows:

1) We propose a novel algorithm called DPNFL to jointly tackle the DP and non-iid-ness in FL. Compared to the few existing works, our method mainly differs in the DP analytical technique and client sampling method. Specifically, our work is the first to adopt tCDP technique under the FL scenario. Different from the traditional DP and more recent RDP, tCDP technique is tighter to bound end-to-end privacy loss so requiring less noise injection for the same level of DP. Besides, unlike the widely adopted full client and MD sampling, our partial client sampling without replacement is both more practical than full client sampling and avoids the sampling variance problem in MD that disadvantages the convergence of non-iid FL. Except

TABLE I
SUMMARY OF MAIN NOTATIONS

T	the number of the global updates
τ	the number of the local updates
$\boldsymbol{w}^*$	the global optimal solution
f^*	the minimum value of global objective function f
f_i^*	the minimum value of local objective function f_i
p_i	the weight of the i-th client
L	the smoothness of the objective function
G	the boundness of the gradient
ϕ_i	the bounded local variance of the local function f_i
Γ_s	the non-iid degree in strongly convex case
Γ_n	the non-iid degree in non-convex case
$\boldsymbol{\gamma}$	the injected noise for differential privacy
σ	the standard variance of the injected noise
$[n]$	the set of integers $\{1, 2, \cdots, n\}$ for total client n
r	the number of sampled clients
$\mathcal{C}_t$	the sampled client subset at round t

for the above methodological innovations, in theoretical analysis, we utilize two kinds of metrics to respectively measure the non-iid degree when the objective function in the FL problem is strongly convex or non-convex and provide the rigorous convergence bounds.

2) To further improve the algorithm performance, we propose an adaptive version of DPNFL named AdDPNFL which adopts the adaptive optimization on the server-side to not only alleviate the impact of non-iid data but also contribute to improving the model accuracy subject to the DP noise. By theoretical analysis, we provide the rigorous convergence bounds when the objective function is strongly convex or non-convex.
3) We perform extensive experiments to validate the effectiveness of our algorithms and demonstrate their superiority over state-of-the-art algorithms.

We arrange the rest of this paper as below. We provide related work and preliminaries in Sections II and III, respectively. We present DPNFL and AdDPNFL algorithms and the corresponding theoretical results in Sections IV and V. We conduct the privacy analysis in Section VI. We perform extensive experiments to showcase the effectiveness and superiority of our algorithms in Section VII. We conclude our work in Section VIII. We give the main notations in Table I.

II. Related Work

Federated Learning on non-iid Data: In the context of FL, for the seminal work and the proposed algorithm FedAvg [1], though the authors claim that FedAvg can cope with non-iid data to a certain degree, a lot of research has indicated that the accuracy decline of FL is almost inevitable on non-iid data [15], [16]. The exceedingly straight strategy to handle such an issue is sharing a public data set among client devices for obtaining approximately iid data. Nevertheless, data sharing possibly induces extra communication costs and even privacy leakage. On the other hand, Karimireddy et al. [17] find that the non-iid data introduces the drift in every client's update, which slows or destabilizes the convergence. Therefore, they correct so-called client drift to tackle the challenge from the non-iid data. Inspired by client drift correction, some works resort to inserting regularization terms into the objective function [18] or bringing additional variate for controlling the clients' local training [19] to tackle the non-iid problem of FL. Nonetheless, the correction term typically utilizes fairly stale control variables, which causes slow convergence. Besides, client-drift correction requires simultaneously communicating the model and gradient of the local client, which doubly consumes communication resources. To avoid the above shortcomings, some researchers attempt to adopt adaptive optimization to handle the non-iid-ness of FL since there exist studies proving that on the non-iid data, this scheme contributes to the training in the circumstance subject to the heavy tailed noises within stochastic gradients [20]. Moreover, some literature proves that an adaptively cooperative update helps better converge in FL [21]. As a dominant case, Reddi et al. [22] are the first to study the adaptive optimization method and show that by mitigating the local learning rate, the adaptive optimization on the server-side can significantly improve the model performance of FL under non-iid data. Therefore, in this paper, we also adopt server-side adaptive optimization to alleviate the impact of the non-iid data on algorithm performance.

Federated Learning With Client Sampling: In general, there are two kinds of client sampling strategies in FL, i.e., full client sampling and partial client sampling. Some early works [23], [24] adopt the full client sampling strategy and require all devices to engage in every communication period, which consumes too many communication resources. In view of this, most works including the seminal FedAvg adopt the partial client sampling to aggregate the client. But existing studies prove that the eventual model of FedAvg is in expectation differing with the determinate aggregations from each client, which implies that the client sampling scheme of FedAvg is biased and thus converges to a sub-optimal minimum [4], [5]. Although some follow-up works [25], [26] make improvements to the sampling strategy of FedAvg, they still encounter bias. To escape from this dilemma, [18] proposes to sample partial clients with replacement from a multinomial distribution, where the sampling probability rests with its own data proportion. The unbiased MD sampling ensures optimality for FL and minimizes the amount of participating clients in each period, so it has been a prevalent sampling strategy in many studies [6], [7], [8]. But since the sample in the MD method is drawn with replacement, each client has a different number of being sampled times, which brings into the sampling variance and thus induces a huge variation in the convergence performance under the non-iid data. To avoid the sampling variance problem, some works [16], [27] choose another unbiased sampling strategy named uniform sampling to sample partial clients uniformly without replacement. Besides, [27] shows that when the number of sampled clients is large, the uniform sampling has a lower time complexity compared to the MD sampling, which is beneficial for computation and analysis. Therefore, in this paper, we choose uniform sampling as the client sampling strategy.

Federated Learning With Differential Privacy: There are many works utilizing DP to secure FL [13], [14], [28], [29], [30], [31], [32], [33], [34], [35], [36], [37]. Multiple papers [30], [31], [32], [33], [34] consider the fully-trusted server so it merely requires resisting the third-party server that eavesdrops on the final model. In this case, the works [30] and [31] enable the cloud server to inject Gaussian noise into the synchronized local model for preserving DP of FedAvg. However, similar to the privacy scenario in our work, numerous studies [13], [14], [35], [36], [37] consider the more practical scenes where the server is honest-but-curious and each client should locally perturb respective update and then send it to the server. This brings more strict privacy surroundings and can not only avert privacy leakage among clients but also between clients and servers. On the other hand, since the first work [38] about the differentially private deep learning emerges where moments accountant technique is designed to analyze the privacy loss, there have been many improved relaxations of classical DP, which mainly include the CDP [39], zCDP [40], RDP [41] and tCDP [42]. As the latest variant, tCDP has been widely adopted as the privacy analysis technique to tightly track the cumulative privacy loss [35], [43]. To our best knowledge, this work is the first to utilize the tCDP to analyze DP in FL.

III. Preliminaries

A. Federate Learning

A general FL framework contains one server and n clients. Each client $i \in [n]$ collectively learns a global optimal model $\boldsymbol{w} \in \mathbb{R}^d$ via solving: $\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \sum_{i=1}^{n} p_i f_i(\boldsymbol{w})$, wherein $f_i(\boldsymbol{w})$ expresses i-th client's objective function and p_i is the weight proportion satisfying $\sum_{i=1}^{n} p_i = 1$. For realizing global optimality, FL performs one T-round procedure. The server first sets the initial global parameter $\boldsymbol{w}_0$. At round t, the server selects r clients at random out of $[n]$ according to sampling rate r/n and shares the global parameter $\boldsymbol{w}_{t-1}$ with selected client devices. Every selected client locally executes τ iterative SGD steps over respective data set and then calculates the difference between local optimum $\boldsymbol{w}_t^i$ and global parameter $\boldsymbol{w}_{t-1}$: $\Delta_t^i = \boldsymbol{w}_t^i - \boldsymbol{w}_{t-1}$, and eventually sends Δ_t^i to the server. The server averages these difference values and updates the global model $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \sum_{i=1}^{r} p_i \Delta_t^i$. The subsequent rounds repeat the identical process.

B. Differential Privacy

Definition 1. ((ϵ,δ)-DP [11]): For the randomized mechanism $\mathcal{M} : \mathbb{X} \to \mathbb{Y}$ with domain $\mathbb{X}$ and range $\mathbb{Y}$, if under arbitrary pairs from the adjacent datasets $X, X' \in \mathbb{X}$ and arbitrary output subset $Y \subseteq \mathbb{Y}$, the following condition is satisfied: $\Pr[\mathcal{M}(X) \in Y] \le e^{\epsilon} \Pr[\mathcal{M}(X') \in Y] + \delta$, $\mathcal{M}$ is regarded as (ϵ,δ)-DP mechanism.

Based on the conventional (ϵ,δ)-DP, Bun et al. [42] proposed truncated concentrated differential privacy (tCDP) to improve the composition analysis, where some valuable definitions and properties are provided.

Definition 2: Given real number $\rho > 0$, $\omega > 1$, arbitrary neighboring pairs $X, X' \in \mathbb{X}$ and η-order Rényi divergence [41] $\mathrm{D}_\eta(\cdot\|\cdot)$, if the following condition is satisfied: $\forall \eta \in (1,\omega)$, $\mathrm{D}_\eta(\mathcal{M}(X)\|\mathcal{M}(X')) \le \rho\eta$, the randomised mechanism $\mathcal{M} : \mathbb{X} \to \mathbb{Y}$ guarantees ω-truncated ρ-concentrated DP (abbreviated as (ρ,ω)-tCDP).

Lemma 1. (Gaussian Mechanism for Achieving tCDP): For mechanism $\mathcal{M}$, the Gaussian mechanism injecting $\mathcal{N}(0,\sigma^2)$ noise into each coordinate of $\mathcal{M}$'s output ensures (ρ,∞)-tCDP, wherein $\sigma \ge S_2(\mathcal{M})/\sqrt{2\rho}$, $S_2(\mathcal{M}) = \sup_{X\sim X'} \|\mathcal{M}(X) - \mathcal{M}(X')\|_2$ is l_2-sensitivity evaluating the largest output change of $\mathcal{M}$ when only one entry changes.

Lemma 2. (tCDP–(ϵ,δ)-DP Conversion): (ρ,ω)-tCDP can be converted into $(\rho + 2\sqrt{\rho\log(1/\delta)}, \delta)$-DP when $\delta \ge e^{-(\omega-1)^2\rho}$.

Lemma 3. (Composition): Given (ρ_1,ω_1)-tCDP mechanism $\mathcal{M}_1 : \mathbb{X} \to \mathbb{Y}_1$ and (ρ_2,ω_2)-tCDP mechanism $\mathcal{M}_2 : \mathbb{X} \times \mathbb{Y}_1 \to \mathbb{Y}_2$ where $\mathcal{M}_2(\cdot, y_1)$ satisfies $y_1 \in \mathbb{Y}_1$, then for mechanism $\mathcal{M}_3 : \mathbb{X} \to \mathbb{Y}_2$ with the definition $\mathcal{M}_3(X) = \mathcal{M}_2(X, \mathcal{M}_1(X))$, it obtains $(\rho_1 + \rho_2, \min(\omega_1,\omega_2))$-tCDP.

Lemma 4. (Privacy Amplification by Subsampling): Suppose real number $\rho, q \in (0, 0.1]$ and $a, A \in \mathbb{N}$ with $q = a/A$ and $\log(1/q) \ge 3\rho(2 + \log_2(1/\rho))$. Given (ρ,ω')-tCDP mechanism $\mathcal{M} : \mathbb{X}^a \to \mathbb{Y}$ with $\omega' \ge \frac{1}{2\rho}\log(1/q) \ge 3$, then for the mechanism $\mathcal{M}_S : \mathbb{X}^A \to \mathbb{Y}$ with the definition $\mathcal{M}_S(x) = \mathcal{M}(x_S)$ where $x_S \in \mathbb{X}^a$ is the restriction of $x \in \mathbb{X}^A$ to the entries specified by a uniformly random subset $S \subseteq [A]$ with $|S| = a$, $\mathcal{M}_S$ obtains $(13q^2\rho, \omega)$-tCDP under $\omega = \frac{\log(1/q)}{4\rho}$.

C. Threat Model

Similar to other privacy issues, before presenting a specific mechanism, a clear formulation should be given about what threat model is being discussed. The attacker in our context is considered as the honest-but-curious cloud server or clients in the FL setting. The cloud side is deemed to reliably obey the predetermined training rule but has a great curiosity towards inferring clients' secret information during model sharing. Moreover, certain clients are able to conspire with the cloud side or each other for inferring the personal data of a particular innocent client. Additionally, the attacker might even be the passive outside adversaries who are capable of eavesdropping on any shared messages while executing training protocol. But they do not spontaneously add fake information to information transference or disturb it.

IV. Differentially Private Federated Learning on Non-iid Data

A. Problem Formulation

We study a non-iid FL framework including n clients, wherein each client i respectively owns local data set $\mathcal{B}_i = \{\zeta_i^1, \ldots, \zeta_i^{n_i}\}$ containing n_i data examples. The clients linked by a center server are aimed at collaboratively seeking the global model $\boldsymbol{w}$ to minimize the optimization goal:

$$f(\boldsymbol{w}) = \sum_{i=1}^{n} p_i f_i(\boldsymbol{w})$$

where p_i represents the proportion of the updates executed by i-th client and it is affected by the data size and distribution among different clients. For iid FL, $p_i = \frac{1}{n}$ but for the non-iid FL, p_i is different among clients and also satisfy $\sum_{i=1}^{n} p_i = 1$. $f_i(\boldsymbol{w}) = \frac{1}{|\mathcal{B}_i|}\sum_{\zeta_i \in \mathcal{B}_i} l(\boldsymbol{w}, \zeta_i)$ indicates client i's local objective function wherein $l(\boldsymbol{w}, \zeta_i)$ expresses the compound cost function regarding example ζ_i and model parameter $\boldsymbol{w}$. Note that for non-iid FL, each dataset $\mathcal{B}_i$ is supposed to have different data distributions, i.e., $\mathbb{E}_{\zeta_i \in \mathcal{B}_i}[l(\boldsymbol{w}, \zeta_i)] \neq \mathbb{E}_{\zeta_i \in \mathcal{B}_{i'}}[l(\boldsymbol{w}, \zeta_i)]$ for any $i \neq i'$. In the context of FL, the parameters $\boldsymbol{w}^*$ and $\boldsymbol{w}_i^*, i \in [n]$ respectively minimizing $f(\boldsymbol{w})$ and $f_i(\boldsymbol{w})$ could be mutually different. We denote $f^* = \min_{\boldsymbol{w}} f(\boldsymbol{w}) = f(\boldsymbol{w}^*)$ and $f_i^* = \min_{\boldsymbol{w}} f_i(\boldsymbol{w}) = f_i(\boldsymbol{w}_i^*)$.

B. Proposed DPNFL Algorithm

In this section, we propose our differential privacy based non-iid federated learning algorithm called DPNFL including three essential ingredients: periodic averaging, partial client sampling and local gradient perturbation.

Periodic Averaging: To alleviate the communication pressure, we adopt the periodic averaging method, i.e., enable the participating clients to perform multiple local updates and then a cloud server executes the periodical aggregation on these updates. More specifically, after the cloud side updates the model and subsequently sends it to the clients, the clients run τ SGD iterative steps to achieve localized model updating and later upload appropriate messages into the cloud side to update the aggregate model. When performing total $\mathbb{T}$ SGD iterative steps on every client, it merely requires $T = \mathbb{T}/\tau$ communication rounds between clients and the server, thus decreasing the overall communication overhead by $1/\tau$ compared to the conventional method that aggregates the local model per training step. As a result, the total communication cost of training the model also decreases.

Partial Client Sampling: As we illustrate in related work, the full client sampling $\boldsymbol{w}_t = \sum_{i=1}^{n} p_i \boldsymbol{w}_i^t$ suffers from some shortcomings such as straggler effect and considerably high communication cost. A common remedy is a kind of partial client sampling method named MD sampling where the server samples a client subset $\mathcal{C}_t(|\mathcal{C}_t| = r)$ with replacement in a Multinomial Distribution, i.e., the server selects each client i depending on respective weight p_i. Then the server performs the aggregation $\boldsymbol{w}_t = \frac{1}{r}\sum_{i \in \mathcal{C}_t} \boldsymbol{w}_i^t$. The unbiasedness of MD sampling improves the optimization performance and communication efficiency, but the with-replacement process degrades the convergence and results in the high time complexity. Therefore, we adopt another kind of partial client sampling method guaranteeing unbiasedness while avoiding the above issues. Specifically, we define a threshold $r(1 \leq r < n)$ and allow the cloud server to gather the output set $\mathcal{C}_t$ composed by the first r reacted clients. Once accumulating r outputs, the cloud side does not wait for the remnant clients anymore and regards the $r + 1$-th to n-th clients to be stragglers at the current iterative step. Then, the aggregate update executes a weighted model average: $\boldsymbol{w}_t \longleftarrow \frac{n}{r}\sum_{i \in \mathcal{C}_t} p_i \boldsymbol{w}_i^t$.

The above sampling process can be generalized as sampling r clients uniformly without replacement and executing the weighted averaging with each client i's weight $\frac{n}{r}p_i$. Thus, we call this sampling method as uniform sampling and the below lemma proves its unbiasedness property.

Lemma 5. (Unbiasedness of Uniform Sampling): For any aggregation time step $t \in \{\tau, 2\tau, \ldots\}$, it has:

$$\mathbb{E}_{\mathcal{C}_t}\left[\frac{n}{r}\sum_{i \in \mathcal{C}_t} p_i \boldsymbol{w}_i^t\right] = \sum_{i=1}^{n} p_i \boldsymbol{w}_i^t$$

Proof: The left side can be expanded as:

$$\sum_{\substack{\mathcal{C} \subseteq [n] \\ |\mathcal{C}| = r}} \Pr[\mathcal{C}_t = \mathcal{C}] \frac{n}{r}\sum_{i \in \mathcal{C}_t} p_i \boldsymbol{w}_i^t = \frac{1}{C_n^r}\frac{n}{r} C_{n-1}^{r-1} \sum_{i \in [n]} p_i \boldsymbol{w}_i^t,$$

which can be simplified as the right side. ■

Therefore, in expectation, uniform sampling is identical to full client sampling and it achieves convergence performance parallel to the local-update SGD approach [44], [45].

Local Gradient Perturbation: From the threat model stated in Section III, the clients and the cloud server in our FL are honest-but-curious and outside adversaries have the capability to sneakingly observe the transferred information. Such adversaries can acquire the most recent global parameter $\boldsymbol{w}_t$ transmitted from the cloud side towards clients and the local model parameters $\{\boldsymbol{w}_i^{t,\tau}\}_{i \in \mathcal{C}_t}$ transmitted from clients towards the cloud, both of which include the secret information in clients' local samples. We hope to utilize DP methods for avoiding privacy exposure within the above two kinds of information. For this aim, we exploit the gradient perturbation based on Gaussian noise [38] for guaranteeing DP and thus the adversary can not infer much privacy regarding a single training example of $\mathcal{B}_i$ according to the shared information. Concretely, for s-th local iterative step in t-th round, client $i \in \mathcal{C}_t$ can update its local model from $\boldsymbol{w}_i^{t,s+1} = \boldsymbol{w}_i^{t,s} - \alpha_l(\boldsymbol{g}_i^{t,s} + \boldsymbol{\gamma}_i^{t,s})$, wherein $\boldsymbol{\gamma}_i^{t,s}$ expresses the Gaussian noise drawn from distribution $\mathcal{N}(0, \sigma^2\mathbf{I}_d)$. Completing τ local iterative steps, the aggregated local parameter $\boldsymbol{w}_i^{t,\tau}$ would achieve DP on client i to some extent, which varies in proportion to noise scale σ. Based on the post-processing feature in DP [11], after summing the local models, the globally shared model $\boldsymbol{w}_{t+1}$ would also achieve the same degree of DP on client i.

Algorithm Update: Next, we utilize the above three essential modules to specifically describe DPNFL. Our presented approach contains T rounds and in each round, all the clients perform τ local updates. At every round t, the cloud server selects $r \leq n$ clients uniformly without replacement which makes up the sampling set $\mathcal{C}_t$. Subsequently, the cloud side transmits its present parameter $\boldsymbol{w}_t$ into each client in $\mathcal{C}_t$ and every client $i \in \mathcal{C}_t$ exploits respective local data set to locally execute τ SGD updates. More specifically, using $\boldsymbol{w}_i^{t,s}$ to express the model of i-th client on s-th iteration at round t, at every local iterative step $s \in [0, \tau - 1]$, client i performs the local model update based on

$$\boldsymbol{w}_i^{t,s+1} = \boldsymbol{w}_i^{t,s} - \alpha_l(\boldsymbol{g}_i^{t,s} + \boldsymbol{\gamma}_i^{t,s})$$

wherein $\boldsymbol{g}_i^{t,s} = \frac{1}{B}\sum_{\zeta_i^{t,s} \in \mathcal{B}_i^{t,s}} \nabla l(\boldsymbol{w}_i^{t,s}, \zeta_i^{t,s})$ expresses the stochastic gradient computed over a mini-batch $\mathcal{B}_i^{t,s}$ containing B samples and $\boldsymbol{g}_i^{t,s}$ unbiasedly estimates $\nabla f_i(\boldsymbol{w}_i^{t,s})$. $\boldsymbol{\gamma}_i^{t,s}$ indicates the noise following Gaussian distribution $\mathcal{N}(0, \sigma^2\mathbf{I}_d)$.

Algorithm 1: DPNFL.

```
Input: Initial model $\boldsymbol{w}_0$, noise scale $\sigma$, number of rounds $T$,
       number of local updates $\tau$, local learning rates $\alpha_l$,
       batch size $B$
Output: $\boldsymbol{w}_{t+1}$
for $t = 0$ to $T - 1$ do
    Server selects the first $r$ responded clients which form a
      sampling subset $\mathcal{C}_t$ with $|\mathcal{C}_t| = r$
    Server sends $\boldsymbol{w}_t$ to each client in $\mathcal{C}_t$
    for each client $i \in \mathcal{C}_t$ in parallel do
        $\boldsymbol{w}_i^{t,0} \leftarrow \boldsymbol{w}_t$
        for $s = 0$ to $\tau - 1$ do
            Compute stochastic gradient $\boldsymbol{g}_i^{t,s}$ on a mini-batch
              $\mathcal{B}_i^{t,s}$ with $B$ samples
            Model update with perturbed gradient:
              $\boldsymbol{w}_i^{t,s+1} \leftarrow \boldsymbol{w}_i^{t,s} - \alpha_l(\boldsymbol{g}_i^{t,s} + \boldsymbol{\gamma}_i^{t,s})$ where $\boldsymbol{\gamma}_i^{t,s} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d)$
        $\Delta_i^t \leftarrow \boldsymbol{w}_i^{t,\tau} - \boldsymbol{w}_t$ and upload $\Delta_i^t$ to the server
    Server computes $\Delta_t = \frac{n}{r}\sum_{i \in \mathcal{C}_t} p_i \Delta_i^t$
    Server updates $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \Delta_t$
```

Notably, each client starts with the same initial parameter $\boldsymbol{w}_i^{t,0} = \boldsymbol{w}_t$. Once updating τ local iterations, the cloud server will aggregate the r collected local model differences and update the next model based on

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t \frac{n}{r} \sum_{i \in \mathcal{C}_t} p_i \Delta_i^t$$

Finally, the process repeats for T rounds. The presented approach is completely described in Algorithm 1.

C. Convergence Analysis

In this part, we give the rigorous convergence analysis of DPNFL. Due to space limitations, we present all the proofs of convergence analysis in supplementary material, available online. Before starting our results, we make the following assumptions:

Assumption 1. (Basic assumption):

a) $f_1, \ldots, f_i$ all satisfy L-smoothness: for any $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$, $f_i(\boldsymbol{a}_1) \leq f_i(\boldsymbol{a}_2) + (\boldsymbol{a}_1 - \boldsymbol{a}_2)^{\mathrm{T}} \nabla f_i(\boldsymbol{a}_2) + \frac{L}{2}\|\boldsymbol{a}_1 - \boldsymbol{a}_2\|_2^2$.

b) Suppose $\boldsymbol{g}_i$ is the stochastic gradient on the mini-batch sampled out of $\mathcal{B}_i$. f_i satisfies the bounded local variance, that is, $\mathbb{E}\|\boldsymbol{g}_i^{t,s} - \nabla f_i(\boldsymbol{w}_i^{t,s})\|^2 \leq \phi_i^2, i \in [n]$.

c) Any stochastic gradient is uniformly bounded, that is, $\mathbb{E}\|\nabla l(\boldsymbol{w}, \zeta)\|^2 \leq G^2$, for arbitrary data sample ζ in $\mathcal{B}_i$ and $\forall \boldsymbol{w} \in \mathbb{R}^d, i \in [n], t \in [0, T-1]$.

d) For any data sample ζ in $\mathcal{B}_i$, the local gradient estimator satisfies unbiasedness, that is, $\mathbb{E}[\nabla f_i(\boldsymbol{w}, \zeta)] = \mathbb{E}[\nabla f_i(\boldsymbol{w})], \forall \boldsymbol{w} \in \mathbb{R}^d$ and $i \in [n]$.

When the objective function f_i is strongly convex, we have the following assumption.

Assumption 2. (Strongly Convex Assumption): $f_1, \ldots, f_n$ all satisfy λ-strongly convex, that is for arbitrary $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$, $f_i(\boldsymbol{a}_1) \geq f_i(\boldsymbol{a}_2) + (\boldsymbol{a}_1 - \boldsymbol{a}_2)^{\mathrm{T}} \nabla f_i(\boldsymbol{a}_2) + \frac{\lambda}{2}\|\boldsymbol{a}_1 - \boldsymbol{a}_2\|_2^2$.

To evaluate the non-iid degree, we adopt the following two definitions which reflect the non-iid degree for strongly convex and non-convex objective functions, respectively.

Definition 3. (Local-Global Objective Gap [16], [25]). Supposing that the objective function f_i satisfies strong convexity, given the global optimal solution $\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} f(\boldsymbol{w})$ and local optimal solution $\boldsymbol{w}_i^* = \arg\min_{\boldsymbol{w}} f_i(\boldsymbol{w})$, the local-global objective gap is defined according to:

$$\Gamma_s \triangleq f^* - \sum_{i=1}^{n} p_i f_i^* = \sum_{i=1}^{n} p_i (f_i(\boldsymbol{w}^*) - f_i(\boldsymbol{w}_i^*)) \geq 0$$

Notably, Γ_s reflects the intrinsic characteristic of the local and global objective functions and it has no relation with the client sampling method. When the non-iid degree is higher, the value of Γ_s becomes larger. When $\Gamma_s = 0$, it means that the local and global optimum are consistent.

Definition 4. (Bounded Gradient Variance): Supposing that the objective function is non-convex, the bounded gradient variance can be used to evaluate the non-iid degree:

$$\sum_{i=1}^{n} p_i \mathbb{E}\|\nabla f_i(\boldsymbol{w}) - \nabla f(\boldsymbol{w})\|^2 \leq \Gamma_n^2$$

Note that $\mathbb{E}\|\nabla f_i(\boldsymbol{w}) - \nabla f(\boldsymbol{w})\|^2$ denoted as Γ_0^2 is a common measure to quantify the non-iid degree when the objective function is non-convex [46], [47]. But in our definition, we also consider the impact of the weight coefficient p_i on the non-iid-ness, which has never been involved by other works.

The following lemma is the key for our analysis, which simplifies our derivation process.

Lemma 6: The gap between the global and local model has the upper bound: for $s = [0, \ldots, \tau - 1]$

- Strongly convex case:

$$\mathbb{E}\|\boldsymbol{w}_i^{t,s} - \boldsymbol{w}_t\|^2 \leq 5\tau\alpha_l^2(G^2 + d\sigma^2 + 4\tau G^2)$$

- Non-convex case:

$$\begin{aligned}&\mathbb{E}\|\boldsymbol{w}_i^{t,s} - \boldsymbol{w}_t\|^2\\ &\leq 5\tau\alpha_l^2(G^2 + d\sigma^2 + 6\tau\Gamma_0^2) + 30\tau^2\alpha_l^2\mathbb{E}\|\nabla f(\boldsymbol{w}_t)\|^2\end{aligned}$$

Now, we give the convergence results for DPNFL w.r.t. strongly convex and non-convex cases.

Theorem 1. (Strongly Convex Case): When Assumptions 1, 2 hold and learning rate $\alpha_l = \frac{4}{\lambda\tau t + 16L\tau}$, then Algorithm 1 satisfies:

$$\mathbb{E}[f(\boldsymbol{w}_T)] - f^* \leq \frac{1}{\xi + T}\left(\frac{2\,L}{\lambda^2}\mathcal{H} + \frac{8\,L^2}{\lambda}\|\boldsymbol{w}_1 - \boldsymbol{w}^*\|^2\right)$$

where $\xi = \frac{16\,L}{\lambda}$, $\mathcal{H} = 40(G^2 + d\sigma^2 + 4\tau G^2) + (8L\frac{n}{r} + 16\,L)\Gamma_s + 4\frac{n}{r}d\sigma^2 + \frac{\lambda}{L}(G^2 + d\sigma^2) + 4\frac{n}{r}\sum_{i=1}^{n} p_i\phi_i^2$ and L, λ, ϕ_i, G are defined in Assumptions and Γ_s defined in Definition 3 is the non-iid degree.

Theorem 2. (Non-Convex Case): When Assumption 1 holds and learning rate guarantees $\alpha_l < \frac{1}{200\tau L}$, then Algorithm 1 satisfies:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{w}_t)\|^2 \leq \frac{1}{T\Upsilon}(f(\boldsymbol{w}_0) - f^*) + \frac{Q}{\Upsilon}$$

where $\Upsilon = \frac{\alpha_l \tau}{2} - 15\alpha_l^3\tau^3 L^2 - 90\frac{n}{r}\alpha_l^4\tau^4 L^3 - \frac{n}{r}\alpha_l^2\tau^2 L$ and $Q = (1 + \frac{n}{r}\tau + \frac{n}{r})\alpha_l(G^2 + d\sigma^2) + (\frac{2n}{r} + 1)\alpha_l\tau\Gamma_n^2$. L, G are defined in Assumption 1 and Γ_n given in Definition 4 is the measure of the non-iid-ness.

Remark 1: When the other parameters are fixed, as the non-iid-ness (Γ_s or Γ_n) increases, the convergence bounds of strongly convex and non-convex cases both grow larger, which implies that non-iid-ness has a negative impact on the convergence performance. Similarly, the noise scale σ also has a negative effect on the convergence performance. In terms of client scheme, different from the convergence result of MD sampling in work [16]'s Theorem 2, which has a weak relation to the amount of sampled clients, our uniform sampling can improve the convergence performance when more clients participate (i.e., increase the client sampling ratio), which is consistent with the findings in [48]. It means that by using uniform sampling, we can in practice increase the participation rate r/n to accelerate the training speed and improve the model utility.

Remark 2: For the strongly convex case, our convergence bound $\mathcal{O}(\frac{1}{T})$ matches the best result of the non-iid FL with uniform sampling [16]. For the non-convex case, our convergence result includes a vanishing term with increasing T and a constant term Q/Υ unconnected with T, the size of which rests with the specific problem variables. Our vanishing term has a convergence speed $\mathcal{O}(\frac{1}{T})$ matching that in classical SGD. Our constant term Q/Υ has a large relation to the non-iid degree. Concretely, when τ is sufficiently big and the sampling ratio has the linear amplification relation to Γ_n^2, for making Q/Υ small, it needs adequately small α_l to counterbalance the difference between two consecutive communications. Thus, to achieve a small Q/Υ, the local learning rate is required to negatively correlate with local steps, which conforms to the findings in strongly convex FL [16], i.e., FL needs to decay the learning rate to ensure convergence on non-iid data.

V. Adaptive Algorithm

The DP technique unavoidably degrades the convergence and deteriorates the model performance since based on the composition characteristic of DP, the privacy loss in each client proportionally boosts as the iteration number grows. As a common remedy, adaptive optimization like Adam can accelerate the training speed and reduce the iteration numbers required for an expected algorithm accuracy, which contributes to the improved convergence speed and model utility. In view of this, an intuitive idea is to use adaptive optimization to replace client-side SGD training. Nevertheless, under FL scenarios, applying adaptive optimization to every client generally exists two specific limitations. First, within the overall training procedure, in general, every client merely participates one time or sporadically participates a few times. Thus, the stale historical contents like the momenta inside Adam bring to the poor algorithm performance if using adaptive optimization to replace SGD on every client in the local update phase. Second, sustaining the historical data on resource-limited clients such as mobile phones commonly costs huge resources used for computing and storing. Given the above, in our adaptive algorithm, the server instead of the clients prepares to perform the adaptive updating, free of extra communication overhead. More appealingly, adaptive server optimization has shown the effectiveness to mitigate the negative impact of non-iid data on FL performance [22], [49]. Therefore, based on the adaptive server optimization, we can effectively tackle the non-iid-ness, privacy-utility trade-off and convergence.

Algorithm 2: AdDPNFL.

```
Input: Initial model $w_0$, initial momenta $[\nu_{-1}]_j \geq \pi^2, \forall j \in [d]$, $\mu_{-1} = \mathbf{0}$, noise scale $\sigma$, number of rounds $T$, number of local updates $\tau$, momentum parameters $\beta_1,\ \beta_2$, learning rates $\alpha_l,\ \alpha_g$, batch size $B$
Output: $w_{t+1}$
for $t = 0$ to $T-1$ do
    Server selects the first $r$ responded clients which form a sampling subset $\mathcal{C}_t$ with $|\mathcal{C}_t| = r$
    Server sends $w_t$ to each client in $\mathcal{C}_t$
    for each client $i \in \mathcal{C}_t$ in parallel do
        $w_i^{t,0} \leftarrow w_t$
        for $s = 0$ to $\tau - 1$ do
            Compute stochastic gradient $g_i^{t,s}$ on a mini-batch $\mathcal{B}_i^{t,s}$ with $B$ samples
            $w_i^{t,s+1} \leftarrow w_i^{t,s} - \alpha_l(g_i^{t,s} + \gamma_i^{t,s})$ where $\gamma_i^{t,s} \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$
        $\Delta_i^t \leftarrow w_i^{t,\tau} - w_t$ and upload $\Delta_i^t$ to the server
    Server computes $\Delta_t = \frac{n}{r}\sum_{i\in\mathcal{C}_t} p_i\Delta_i^t$
    $\mu_t \leftarrow \beta_1\mu_{t-1} + (1-\beta_1)\Delta_t$
    $\nu_t \leftarrow \beta_2\nu_{t-1} + (1-\beta_2)\Delta_t^2$
    $w_{t+1} \leftarrow w_t + \alpha_g\mu_t/(\sqrt{\nu_t} + \pi)$
```

Overall, we proposed an adaptive algorithm named AdDPNFL based on DPNFL. In this algorithm, the server-side keeps two vector-valued momentums $\boldsymbol{\mu}, \boldsymbol{\nu} \in \mathbb{R}^d$ and updates them every round. For any round t, once completing τ local iterative steps, all the clients $i \in \mathcal{C}_t$ transmit their own model updates $\Delta_i^t = \boldsymbol{w}_i^{t,\tau} - \boldsymbol{w}_t$ into the server for improving the global parameter according to the following rules:

$$\begin{cases} \boldsymbol{\mu}_t = \beta_1\boldsymbol{\mu}_{t-1} + (1-\beta_1)\frac{n}{r}\sum_{i\in\mathcal{C}_t} p_i\Delta_i^t \\ \boldsymbol{\nu}_t = \beta_2\boldsymbol{\nu}_{t-1} + (1-\beta_2)\boldsymbol{\mu}_t^2 \\ \boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha_g\boldsymbol{\mu}_t/(\sqrt{\boldsymbol{\nu}_t} + \pi) \end{cases}$$

wherein $\beta_1, \beta_2 \in [0, 1)$ express momentum hyperparameters, α_g is the global step size and π determines adaptivity degree. The complete algorithm is given in Algorithm 2.

A. Convergence Analysis

Theorem 3. (Strongly Convex Case): With Assumptions 1, 2 and Definition 3, $L, G, \phi_i, \lambda, \Gamma_s$ are defined there. Supposing that the learning rate guarantees $\alpha_g \in [\frac{3\sqrt{LG}(\frac{G}{L}+\pi)}{2\lambda}, \frac{2\beta_2 G}{d\tau\lambda}]$, $\alpha_l \in [\sqrt{\frac{d}{\beta_2 LG}}, \frac{1}{4L\tau}]$ and $\alpha_g\alpha_l \leq \frac{r\beta_2\pi^2}{2n\tau(G+L)}$, then Algorithm 2 satisfies:

$$\mathbb{E}[f(\boldsymbol{w}_T)] - f^* \leq \frac{L}{2T}\mathbb{E}\|\boldsymbol{w}_0 - \boldsymbol{w}^*\|^2 + \frac{3L}{2}\mathcal{K}$$

where f^* is the optimal objective value and $\mathcal{K} = 20\frac{G^2+d\sigma^2+4\tau G^2}{L^2} + (\frac{4\frac{n\tau^2}{rL}+4\frac{\tau}{L}}{\beta_2\pi^2} + \frac{4\,d}{\beta_2\,G})\Gamma_s + \frac{2\tau\lambda d(G^2+d\sigma^2)}{\beta_2\,L^2\,G} + \frac{\frac{n\tau^2}{rL^2}(2d\sigma^2+2\sum_{i=1}^n p_i\phi_i^2+G^2+d\sigma^2)}{\beta_2\pi^2}$.

Theorem 4. (Non-Convex Case): With Assumption 1 and Definition 4, L, G, Γ_n are given therein. Supposing that the local learning rate guarantees $\alpha_l \leq \frac{1}{8\tau}\min\{\frac{1}{L}, \frac{\pi r}{10n(G+\alpha_g L)}\}$, then Algorithm 2 satisfies:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\boldsymbol{w}_t)\|^2 = \mathcal{O}\left((\sqrt{\beta_2}\alpha_l\tau G/\sqrt{d}+\pi)(\Theta+\Theta')\right)$$

where f^* is the optimal objective value and

$$\Theta = \frac{f(\boldsymbol{w}_0)-f^*}{\alpha_l\alpha_g\tau T} + \frac{G^2+d\sigma^2+6\tau\Gamma_n^2}{\tau\pi},\ \Theta' = (G+\alpha_g L)\cdot$$
$$\frac{n}{r}\left[\frac{6\tau\alpha_l}{\pi^2}(G^2+d\sigma^2) + \frac{\alpha_l}{\pi^2}(G^2+d\sigma^2+6\tau\Gamma_n^2) + \frac{6\tau\alpha_l\Gamma_n^2}{\pi^2}\right]$$

Remark 3: Except for satisfying similar analyses in Remark 1, the convergence bounds of the strongly convex and non-convex cases for Algorithm 2 are also affected by the adaptivity degree π that has a positive impact on the convergence bounds. More importantly, by carefully choosing the local and global learning rates, we can mitigate the impact of the non-iid-ness but cannot totally eliminate it. Under exceedingly non-iid scenarios, we might need the other methods like client-drift correction [17]. Nevertheless, our experimental analyses show that for modest, spontaneous non-iid-ness, adaptive optimization is fairly effectual, specifically under the cross-device environment. Besides, we can easily combine our algorithms with those methods.

Remark 4: For the strongly convex case, some previous works like [50] also study the convergence bound over the non-iid data and partial client participation. Whereas, they adopt a stricter bounded gradient diversity assumption different from our bounded gradient norm assumption. Specifically, their assumption implies that the global minimizer $\boldsymbol{w}^*$ also minimizes each local loss function, which is unrealistic under non-iid FL scenarios. For the non-convex case, when $T \gg \tau$, we can choose the suitable $\alpha_l = \frac{1}{L\tau\sqrt{T}}$ and $\alpha_g = \sqrt{r\tau}$ to obtain the convergence bound $\mathcal{O}(\frac{1}{\sqrt{r\tau T}})$, which matches the best existing result under the non-convex setting of our interest [22]. Therefore, compared to the general convergence bounds like $\mathcal{O}(\frac{1}{\sqrt{T}})$ of [46], our bound of non-convex case achieves the linear speedup with the help of partial client sampling and local SGD.

Remark 5: To obtain the convergence results of our algorithms, we have made abundant theoretical analyses. To better illustrate our technical contribution, we summarize the crucial points as follows. First, our adopted uniform client sampling without replacement brings technical difficulty compared with the widely-used full or partial client sampling. Specifically, in Proposition 1 of the supplementary material, available online, we meticulously study the squared norm of the adopted sampling method, which is essential for calculating the aggregated model difference Δ_i^t. However, this quantity is very easy to analyze under full/partial client sampling via Jensen's inequality. Besides, unlike some existing works [25], [51] only suitable for the balanced FL setting with $p_i = 1/n$, our convergence analysis applies to any $p_i \in [0,1]$, which makes our analytical techniques more general. Finally, compared to some works [52], [53] only providing the relationship for the learning rate, finding the explicit learning rate condition is technically challenging due to the element-wise operations in our adaptive algorithm.

VI. Privacy Analysis

As discussed in our threat model, we aim to exploit DP mechanisms to resist outside adversaries or the semi-honest cloud server and clients to infer private details from certain clients' local samples. Rather than utilizing the routine (ϵ,δ)-DP method, we first employ tCDP to record the tighter end-to-end privacy loss for our algorithms across multiple iterations and then convert it into (ϵ,δ)-DP. In what follows, under the premise of Assumption 1-c), we first give the sensitivity analysis about gradient $\boldsymbol{g}_i^{t,s}$ in Lemma 7 and thus obtain the tCDP guarantee under one iteration in Algorithms 1 and 2. Significantly, Assumption 1-c) is general under DP-FL settings and could be satisfied via gradient calibration methods [38]. Lastly, in Theorem 5, we prove Algorithms 1 and 2 satisfy (ϵ,δ)-DP in client i across T rounds.

Lemma 7: The sensitivity of the stochastic gradient $\boldsymbol{g}_i^{t,s}$ on client i in every local update has the upper bound $2\,G/B$.

Proof: On client i, for arbitrary adjacent data sets X_i and X_i' which are merely different on data example j and both contain B data samples, then in every local update, the stochastic gradient has the following sensitivity:

$$\|\boldsymbol{g}(\boldsymbol{w}_i^{t,s}, X_i) - \boldsymbol{g}(\boldsymbol{w}_i^{t,s}, X_i')\|_2 = \frac{1}{B}\|\nabla l(\boldsymbol{w}_i^{t,s}, \zeta_j)$$
$$- \nabla l(\boldsymbol{w}_i^{t,s}, \zeta_j')\|_2$$

According to Assumption 1-c), we can estimate the sensitivity of $\boldsymbol{g}_i^{t,s}$ to be $\Delta_2(\boldsymbol{g}_i^{t,s})) \leq 2\,G/B$. ■

Theorem 5: For the mini-batch $\mathcal{B}_i^{t,s}$ selected at random without replacement from $\mathcal{B}_i$ according to the sampling ratio B/n_i and the Gaussian noise $\boldsymbol{\gamma}_i^{t,s}$ following $\mathcal{N}(0,\sigma^2\mathbf{I}_d)$. Assume K_i as the amount of participating rounds for client i over the entire training procedure and then our algorithms achieve (ϵ,δ)-DP on client i, wherein

$$\epsilon = \frac{26K_i\tau G^2}{n_i^2\sigma^2}\left(\frac{B^2\sigma^2\log\frac{n_i}{B}}{4\,G^2} - 1\right)$$
$$\delta = exp\left(-\frac{26K_i\tau G^2}{n_i^2\sigma^2}\left(\frac{B^2\sigma^2\log\frac{n_i}{B}}{8\,G^2} - 1\right)^2\right)$$

Proof: From Lemmas 1 and 7, every local update in our algorithms guarantees $(\frac{2\,G^2}{B^2\sigma^2}, \infty)$-tCDP over client i's sub-sampled mini-batch $\mathcal{B}_i^{t,s}$. Because of client selection, not each client would send its model to the server across t-th round. Supposing that a certain client does not send the model, it will not expose its privacy at the current round. According to the sub-sampling amplification characteristic of tCDP in Lemma 3 and satisfying the conditions therein, every local update in our algorithms obtains $(\frac{26\,G^2}{n_i^2\sigma^2}, \frac{B^2\sigma^2 log\frac{n_i}{B}}{8\,G^2})$-tCDP. Assuming that undergoes T rounds of communication, client i performs $K_i\tau$ SGD iterative step. From Lemma 4, client i achieves a total $(\frac{26K_i\tau G^2}{n_i^2\sigma^2}, \frac{B^2\sigma^2 log\frac{n_i}{B}}{8G^2})$-tCDP after T communication rounds. Finally, we can convert tCDP into (ϵ, δ)-DP via Lemma 2. Significantly, any client in every round participates in the communication according to sampling ratio r/n and hence the expected value of K_i is the same as that of Tr/n. ■

Remark 6: Although existing works like [54], [55] can also achieve tight privacy accountant via RDP, our adopted tCDP possesses some distinct advantages. Specifically, the adjacent datasets of RDP are separately based on the addition/removal relation [55] and replacement relation [54]. Since the work [56] has implied that the replacement relation is approximately twice as strong as the removal/addition relation, our adopted tCDP with the replacement relation under adjacent datasets has a better privacy definition compared to [55]. Moreover, although our adopted tCDP and the RDP in [54] have the same type of the adjacent dataset, the subsampling method in [54] considers the sampling without replacement and our adopted tCDP considers the Poisson sampling method which is leveraged in the widely-used DPSGD algorithm. Finally, although the RPD and our adopted tCDP can both provide a better privacy bound after subsampling, the analytical result of privacy amplification from subsampling in RDP [54] is rather complex and only can be efficiently used by numerical computation. However, in Lemma 4, our adopted tCDP provides simple subsampling results, which can be easily obtained by manual calculation.

VII. Experiments

Model Setting. 1) *Strongly convex case:* We inspect the previously theoretic analyses regarding a strongly convex case (Theorems 1 and 3) via a multinomial logistic regression model. Concretely, if $l(\boldsymbol{w}, \boldsymbol{z}_i)$ expresses the predictive model containing the parameter $\boldsymbol{w} = (\boldsymbol{W}, \boldsymbol{b})$ and the relation $l(\boldsymbol{w}, \boldsymbol{z}_i) = \text{softmax}(\boldsymbol{W}\boldsymbol{z}_i + \boldsymbol{b})$, we can describe the loss function as: $f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n} \text{CrossEntropy}\,(f(\boldsymbol{w}, \boldsymbol{z}_i), \boldsymbol{y}_i) + h\|\boldsymbol{w}\|_2^2$, which forms a convex optimization problem. We can choose the regularization hyperparameter as $h = 10^{-3}$.

2) *Non-convex case:* For exhibiting the superiority of the proposed algorithms subject to non-convex objective functions (corresponding to Theorems 2 and 4), we carry out experiments over different datasets on a deep neural network model, whose structure is similar to that in [38] about DPSGD. We utilize one feed-forward neural network including ReLU units and softmax of 10 categories having cross-entropy loss. The network unites one PCA projection layer with dimension 60 and one hidden layer including 1200 hidden units. From Theorems 2 and 4, the convergence bounds grow as the model parameter dimensionality increases and therefore we need the PCA layer for avoiding the dimension defects affected by the artificial noise. Likewise, the deeper neural networks also undergo this issue. The PCA implementation can be used to preprocess entire non-private samples under our frame.

Dataset. 1) *Logistic regression (w.r.t. strongly convex case):* We conduct the performance evaluation of the previous theoretic analyses regarding strongly convex functions (Theorems 1 and 3) over MNIST[1] and FMNIST[2] dataset. We use the MNIST dataset since it is prevalent in academia. For achieving non-iid data distribution, we distribute the data amongst all the clients guaranteeing that every client includes examples with seven digits/fashions. For investigating the impact of data unbalancedness, we further change the sample size amongst clients. Concretely, under the unbalanced scenarios, the sample size between clients obeys a power law, but under the balanced setting, we enforce an equal number of examples on each client. Similarly, we execute these operations for the FMNIST dataset to validate the generality of our algorithms. (2) DNN (w.r.t. non-convex case): We adopt SVHN[3] and CIFAR10[4] datasets to examine the theoretic findings subject to non-convex objective functions (Theorems 2 and 4). According to the above description, we likewise perform the data allocation on these two datasets for acquiring non-iid data distribution.

To clarify the impact of the non-iid-ness on the algorithm performance in more depth, we also run our algorithms on the following synthetic dataset (w.r.t. the Logistic Regression model) and variant CIFAR10 dataset (w.r.t. the DNN model). Synthetic data helps us control non-iid-ness more exactly. Herein, we adopt the same setting as the work [16]. We use Synthetic(u, v) to represent the synthetic dataset with hyperparameters u and v, where u determines the degree of difference between local models and v determines the degree of difference between the local data on every client and those on other clients. The sample size n_i on every client obeys a power law, namely, we distribute data according to an unbalanced fashion.

For the DNN model, to further investigate the impact of the non-iid-ness on our algorithms, we exert the Dirichlet distribution used in [49] for generating disjointly non-iid client training data and denote it as Dir(ψ). The value of ψ determines the degree of non-iid-ness: $\psi = 100$ mimics identical local data distributions and a lower ψ means that clients have a high probability to possess examples from merely one class.

Parameter Setting: The overall number of clients n is set to 100. Privacy level and failure probability are respectively set as $\epsilon = 0.3$ and $\delta = 10^{-2}$. The default sampling ratio of clients is set as 0.1. We adjust the hyperparameters via grid search and give the optimal values as follows: over MNIST and FMNIST, we choose batch size $B = 10$, local iteration numbers $\tau = 300$ and local learning rate $\alpha_l = 0.01$; over SVHN and CIFAR10, the corresponding values are $B = 50$, $\tau = 50$ and $\alpha_l = 0.1$, respectively. For Algorithm 2, we choose $\beta_1 = 0.9, \beta_2 = 0.99$ and $\pi = 10^{-3}$. We choose the global learning rate over MNIST and FMNIST as $\alpha_g = 0.01$ and over SVHN and CIFAR10, this value is 0.005. Furthermore, we enable local and global learning rates to have the decaying rate $1/\sqrt{t}$.

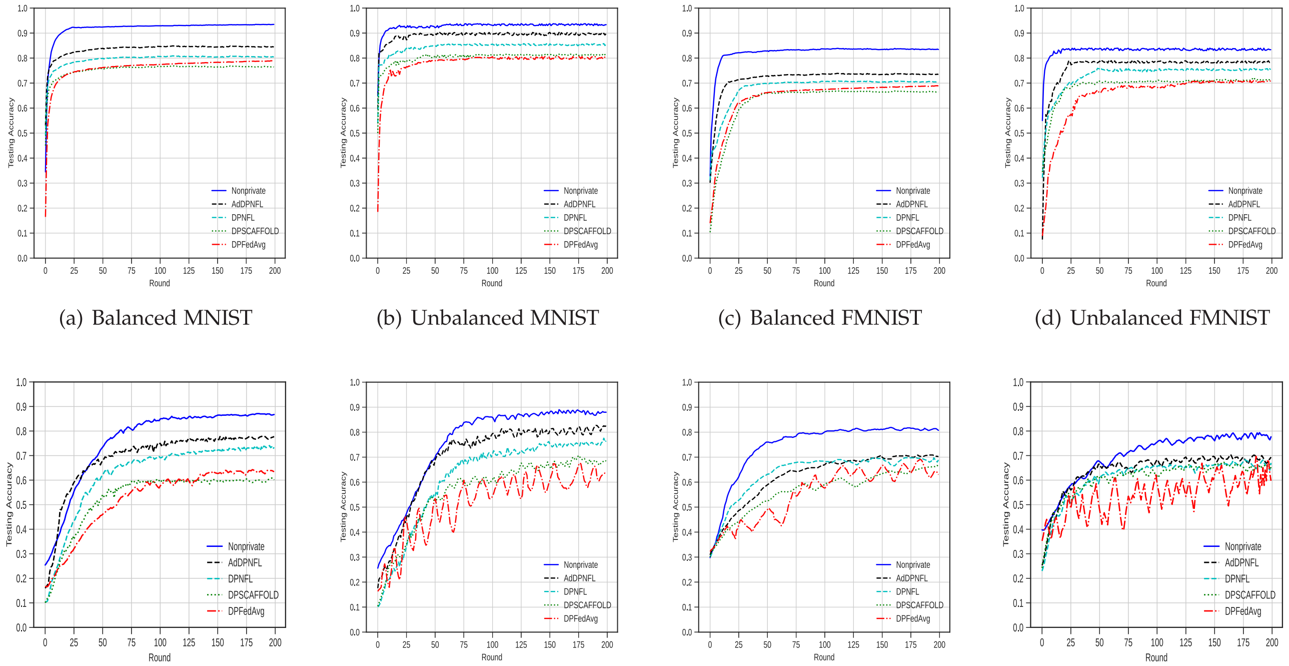
Over the above datasets, we utilize the aforementioned settings about parameters, models and loss functions. Without considering the noise injection, for the logistic regression model, we can achieve 93.48% testing accuracy on MNIST dataset

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

[2] https://github.com/zalandoresearch/fashion-mnist

[3] http://ufldl.stanford.edu/housenumbers/

[4] https://www.cs.toronto.edu/ kriz/cifar.html



(a) Balanced MNIST

(b) Unbalanced MNIST

(c) Balanced FMNIST

(d) Unbalanced FMNIST

(e) Balanced SVHN

(f) Unalanced SVHN

(g) Balanced CIFAR10

(h) Unbalanced CIFAR10

Fig. 1. Comparison of our DPNFL and AdDPNFL algorithms with other differentially private algorithms on Logistic Regression (first row) and DNN (second row).

and 83.44% testing accuracy on FMINST dataset. These results are parallel to those reached under a vanilla neural network. Similarly, the best testing accuracy of non-private version on SVHN and CIFAR10 is 87.74% and 80.35% under our DNN architecture.

Comparative Algorithms:

- DPSCAFFOLD [13] is the state-of-the-art differentially private federated learning algorithm on non-iid data, which is based on the SCAFFOLD federated learning algorithm and combines the control variable with RDP to simultaneously tackle heterogeneity and DP.
- DPFedAvg [31] is the first differentially private federated learning algorithm, which adopts the moments accountant method to achieve the DP for FedAvg algorithm.

A. Experimental Result

Balanced versus Unbalanced Data: For the strongly convex case, we first perform the experiment on logistic regression model over the balanced and unbalanced MNIST datasets. From Fig. 1(a), we can see that on the balanced data, the non-private version of our AdDPNFL algorithm (denoted as Nonprivate) can achieve up to 93.48% testing accuracy which is parallel to the state-of-the-art accuracy of the FL algorithm on MNIST dataset. It implies the advantage of our adaptive algorithm in model utility. Moreover, comparing with four differentially private algorithms, our AdDPNFL achieves the highest accuracy (84.45%) and DPSCAFFOLD achieves the lowest accuracy (76.45%). It means that for the balanced data, our adaptive server aggregation method is helpful for convergence and contribute to less noise addition which results in higher accuracy. Besides, comparing the testing accuracies of two baseline algorithms, i.e., DPSCAFFOLD (76.44%) and DPFedAvg (78.79%), we can find that in the balanced case, the testing accuracy on DPSCAFFOLD improves insignificantly and might degrade, and its model convergence rate decreases. The major reason is that during early model training, the global variate after random initialization misguides the model's optimization direction. For the testing accuracies of two proposed algorithms, namely, AdDPNFL (84.45%) and DPNFL (80.59%), it can be seen that the adaptive version achieves better accuracy, which means that the adaptive optimization is beneficial for the training result under the DP noise. Comparing the proposed algorithms with two baseline algorithms, we can see that they are both superior to the baselines. The reason is that different from the ordinary server aggregation, our DPNFL and AdDPNFL are both based on the partial client selection without replacement strategy which can avoid the negative straggler effect in DPSCAFFOLD and DPFedAvg. Furthermore, in the client-side sampling case, the local control variate in DPSCAFFOLD seldom gets updated and thus exploiting the control variates for estimating the updating direction could be considerably imprecise. As a result, even for the balanced scenario, our algorithms are better than the baseline algorithms.

For the non-iid and unbalanced scenario, from Fig 1(b), we have also observed similar results, but some differences exist. First, we can see that on the unbalanced data, all the algorithms achieve relatively unstable training procedures since the non-iid and unbalanced data will make the direction of the gradient more fluctuated. Second, we can see that under the unbalanced environment, our non-private algorithm changes a little but the other three differentially private algorithms including AdDPNFL, DPNFL and DPSCAFFOLD achieve better accuracies compared with the corresponding accuracies in the balanced case. It means that these algorithms specific for heterogeneous data including the non-iid and unbalanced data make improvements on the model utility. However, for DPFedAvg, the testing accuracy in unbalanced cases declines, which validates the limitation of DPFedAvg under the heterogeneous dataset. Besides, we can see that our proposed algorithms are still better than DPSCAFFOLD. Except for the better client sampling strategy, it should be noted that our DPNFL uses the more suitable non-iid measure which simultaneously includes the heterogeneity of data size and data distribution. More importantly, our improved version of AdDPNFL further adopts the adaptive optimization, as has been illustrated in the remark, the adaptive optimization foster the convergence rate on the non-iid data with the accelerated gradients. Therefore, our algorithms are better than the baseline algorithms in the heterogeneous setting. Last but not least, to generalize the observations on MNIST, we also perform the corresponding experiment on the FMNIST dataset and Fig. 1(c)–(d) demonstrates the similar results to validate the effectiveness of our algorithms.

For the non-convex case, we perform the experiments on the DNN model over the balanced SVHN and unbalanced SVHN datasets. From Fig. 1(e) and (f), we can see that the tendencies of all the algorithms are similar to the results in Fig. 1(a) and (b). Likewise, 86.74% testing accuracy of our non-private AdDPNFL on the SVHN dataset showcases the superiority of the adaptive server aggregation methods. It is easy to find that the observations in Fig. 1(a) and (b) are nearly suitable for the case of the SVHN dataset. But since the format of training data in SVHN is more complex than that in MNIST, we can see that when the data is balanced, the variation in SVHN is more unstable than that in MNIST. Also for this reason, the curves in unbalanced SVHN are considerably fluctuated compared to the curves in balanced SVHN. Additionally, it is deserved to illustrate that on the large model structure and the complex dataset, our proposed algorithms still achieve better algorithm performance than that of DPSCAFFOLD. DPSCAFFOLD usually deteriorates during carrying on large scale deep learning since it merely achieves approximate reduction on gradient drift across every round but cannot remove it. This kind of residual error could cumulatively grow as training proceeds, which has been discussed in [15] and it induces slower convergence and worse performance. But for DPNFL, we do not introduce extra control variables to cope with non-iid-ness and therefore algorithm performance has little reaction from the model and dataset. Finally, we also execute the corresponding experiment on the CIFAR10 dataset, which further validates the reasonability of experimental observations (see Fig. 1(g) and (h)).

To intuitively exhibit the effectiveness of our algorithms, we depict the testing accuracy of different algorithms on various unbalanced datasets in Fig. 2, where the testing accuracies are the average accuracy by running five times. From Fig. 2, we first observe that the non-private AdDPNFL achieves the highest testing accuracy, which is in line with the fact that DP noise degrades the model utility. Second, it is apparent that AdDPNFL excels in all the other differentially private algorithms over all the unbalanced datasets. It means that the adaptive server aggregation and client sampling without replacement in our algorithm design, as well as our measurement of non-iid-ness, can effectively handle the non-iid data distribution. Besides, for the complex datasets, all the testing accuracies decline since the classification task is harder for the more complex dataset.

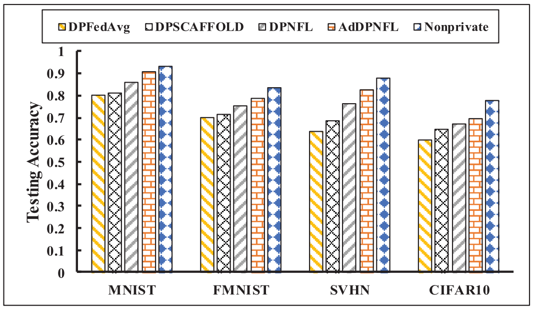
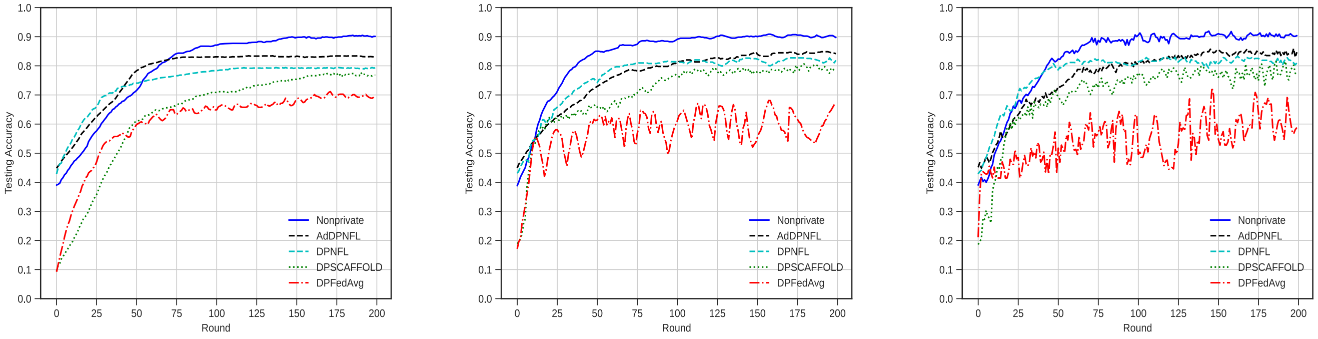


Fig. 2. Testing accuracy of different algorithms under various unbalanced datasets.

Synthetic and Dirichlet Distribution Dataset: To investigate the impact of non-iid-ness on the algorithm performance in more depth, we perform our algorithms on the logistic regression model over the synthetic dataset and split the CIFAR10 dataset according to the Dirichlet distribution as the training dataset for DNN model. We choose three levels of non-iid-ness for synthetic data, that is, Synthetic-IID corresponding to iid-ness, Synthetic(1, 1) corresponding to middle non-iid-ness and Synthetic(5, 5) corresponding to high non-iid-ness. For CIFAR10, we also choose three levels for non-iid-ness where Dir(100), Dir(5) and Dir(0.1) respectively represent the low, middle and high non-iid-ness. The details of how the parameters control the non-iid-ness have been stated in part of parameter settings.

From Figs. 3 and 4, we can easily see that whether in the high or low non-iid datasets, our algorithms can outperform the other baseline algorithms. The observations in synthetic dataset and variant CIFAR10 are partially similar to that in Fig. 1(a) and (b). Herein, we only give the unique observations for these more fine-grained non-iid experiments. First, we can see that as the non-iid-ness increases, the training procedures become much more fluctuated. The reason is that the non-iid data severely influences the update direction of the gradient. When the dataset is highly non-iid, the gradient update becomes quite unstable, which results in unstable training curves. Second, it is evident that under the low or high non-iid-ness, the DPSCAFFOLD algorithm achieves worse testing accuracy than that in the middle non-iid case. The limitation of DPSCAFFOLD in the low non-iid case is that for iid or low non-iid data distribution, the local control variate in DPSCAFFOLD seldom gets updated and thus exploiting the control variates for estimating the updating direction could be considerably imprecise. In a high non-iid case, considering that not all the users update respective local control variates across each round, the high non-iid data distribution

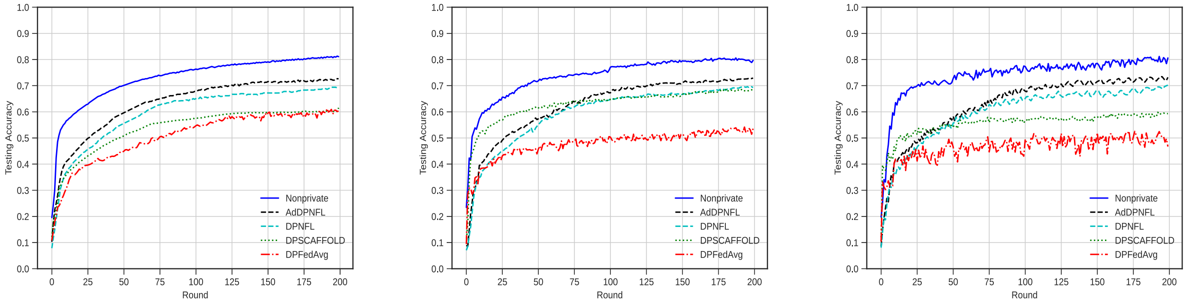


(a) Synthetic-IID

(b) Synthetic(1, 1)

(c) Synthetic(5, 5)

Fig. 3. Comparison of our DPNFL and AdDPNFL algorithms with other differentially private algorithms on Logistic Regression over the synthetic dataset.



(a) Dir(100)

(b) Dir(5)

(c) Dir(0.1)

Fig. 4. Comparison of our DPNFL and AdDPNFL algorithms with other differentially private algorithms on DNN over the CIFAR10 dataset.

TABLE II
TESTING ACCURACY FOR DIFFERENT DEGREES OF NON-IID-NESS ON SYNTHETIC AND CIFAR10 DATASETS

Dataset	Non-iid-ness Settings	Testing Accuracy				
		DPFedAvg	DPSCAFFOLD	DPNFL	AdDPNFL	NonPrivate
Synthetic	Syn-IID	69.84 ± 1.05	76.81 ± 1.05	78.35 ± 0.29	**82.54 ± 0.17**	90.12 ± 0.44
	Syn (1, 1)	67.14 ± 1.21	77.15 ± 1.44	81.90 ± 0.32	**84.18 ± 0.19**	89.72 ± 0.56
	Syn (5, 5)	66.41 ± 2.42	77.08 ± 1.57	81.99 ± 0.45	**84.68 ± 0.34**	90.39 ± 0.77
CIFAR10	Dir(100)	59.38 ± 0.67	61.56 ± 0.64	69.04 ± 0.16	**72.65 ± 0.21**	81.09 ± 0.71
	Dir(5)	54.95 ± 1.57	67.65 ± 0.85	69.90 ± 0.73	**72.90 ± 0.63**	79.67 ± 0.86
	Dir(0.1)	48.46 ± 1.59	59.32 ± 1.67	70.20 ± 1.18	**73.19 ± 1.04**	80.60 ± 1.15

might result in an unacceptable difference between local control variate and latest global control variate, which finally leads to the control-lag problem so the model accuracy gets worse when the non-iid-ness is too high. But our algorithm is relatively less affected by the non-iid-ness. More specifically, as the non-iid-ness increases, the testing accuracies of our algorithms achieve slightly better performance.

Except for depicting the training process in Figs. 3 and 4, we also present the final testing accuracies of different cases in synthetic and CIFAR10 datasets in Table II . From this table, we can easily see that due to the noise addition, the non-private AdDPNFL is at least 4.85% better than all the other differentially private algorithms. Among the differentially private algorithms, AdDPNFL expectedly acquires the best testing result and is at least 3.77% better than DPSCAFFOLD, which validates the superiority of our algorithms.

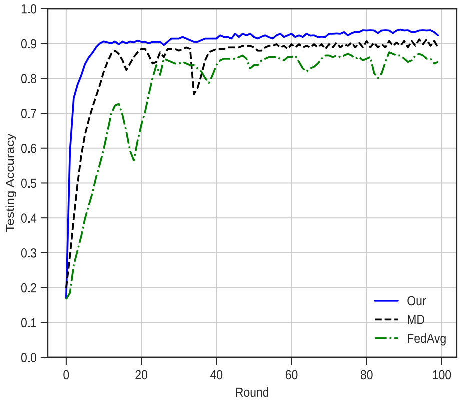
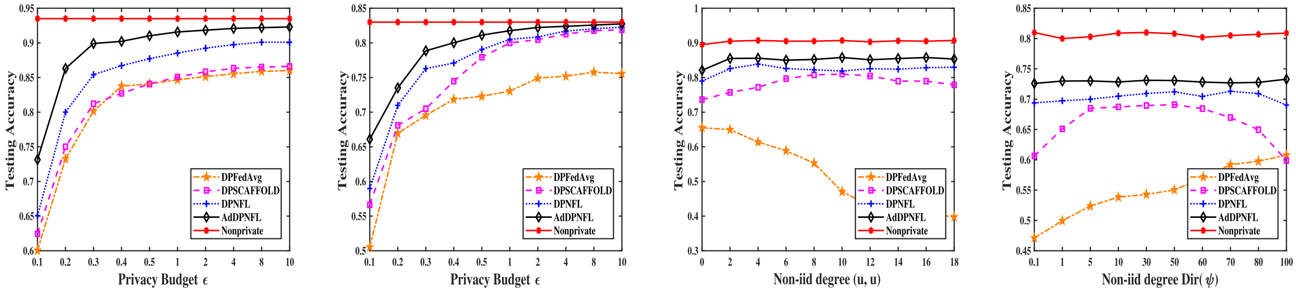


Fig. 5. Comparative results of our sampling method with the existing approaches.



(a) Logistic regression on MNIST

(b) DNN on SVHN

(c) Logistic regression on synthetic Dataset

(d) DNN on CIFAR10

Fig. 6. Impacts of the privacy level ((a) and (b)) and non-iid degree ((c) and (d)) on the testing accuracy under the strongly convex and non-convex settings.

On the other aspect, to validate the effectiveness of our uniform client sampling without replacement, we want to compare it with the existing sampling methods including the widely-used MD sampling [16], [17], [18] and the conventional FedAvg's sampling [1] that separately consider the sampled and unsampled client. For a fair comparison, we choose the non-private Algorithm 1 as our basic algorithm since it does not consider the adaptive optimization and DP, which is a pure client sampling algorithm. Meanwhile, we choose the original algorithms in [16] and [1] as the comparative algorithms since their works only consider client sampling problems in FL. We chose the previous unbalanced MNIST as the training dataset since we want to study the performance of sampling methods under the non-iid/heterogeneous setting. The experimental results are shown in the Fig. 5. From this figure, we can see that our sampling method achieves the best testing accuracy compared with other existing approaches and the training process is more stable. Besides, the convergence speed of our algorithms is the fastest, which validates the superiority of our sampling method analyzed in the introduction part. Specifically, FedAvg's client sampling is biased and can only converge to a sub-optimal minimum so it achieves the lowest accuracy among these three algorithms. Besides, due to the abrasion brought by the unsampled clients, FedAvg's sampling method typically leads to a slow convergence speed. For MD sampling, since the server selects clients with replacement, the number of samplings in the different clients causes a variance, which likely produces a huge variation in convergence performance, specifically for the non-iid FL.

Finally, we adjust three important hyperparameters: privacy budget ϵ, non-iid-ness level (u, v) or ψ, and adaptivity degree π to observe the impact of the hyperparameter on the algorithm performance. Herein, without loss of generality, we only maintain one parameter u in the synthetic dataset setting for brevity. That is, we denote the different levels of non-iid-ness as Synthetic(u, u) on the synthetic dataset.

Impact of the Privacy Budget: In Fig. 6(a), we plot the testing accuracies of different algorithms with various privacy budgets ϵ on the logistic regression model over the MNIST dataset. Unsurprisingly, the non-private AdDPNFL keeps unchangeable when the privacy budget varies. Additionally, it is not hard to find that when the privacy budget increases, the testing accuracies of all the differentially private algorithms go up. This is because the big privacy budget means a low privacy level, which adds less noise to the algorithms. Therefore, the model utility will improve. More importantly, we observe that whether in a small or big privacy budget, the testing accuracies of our algorithms are higher than those of the baseline algorithms. Especially, the advantage in the case of a high privacy level (small privacy budget ϵ) is more distinct. This means that our algorithm is feasible to simultaneously achieve high performance and strong privacy guarantees. For completeness, we also conduct the experiments on the DNN model with the SVHN dataset and exhibit the experimental results in Fig. 6(b), where the results are similar to those in MNIST. But it should be noted that on the more complex dataset SVHN, all the model accuracies will decrease and DPFedAvg has the inadequate capability to tackle the non-iid data.

Impact of the non-iid degree: To elaborate the impact of the non-iid-ness, we execute the previous experiments on synthetic and variant CIFAR10 datasets with more different non-iid-ness parameters. Specifically, for the logistic regression on the synthetic dataset, we vary Synthetic(u, u) from Synthetic(0, 0) to Synthetic(18, 18). It is required to point out that Synthetic(0, 0) is a low non-iid setting rather than an iid setting and Synthetic(18, 18) corresponds to a high non-iid setting. For the DNN model on the CIFAR10 dataset, we vary the non-iid degree from $\psi = 0.1$ to $\psi = 100$ where $\psi = 100$ can be seen as the iid setting. From Fig. 6(c) and (d), we can find that the testing accuracies of our algorithms including non-private AdDPNFL, AdDPNFL and DPNFL, change a little or become slightly higher when the non-iid-ness varies. It demonstrates the robustness of our algorithms for the non-iid FL. Besides, we can see that DPSCAFFOLD has the best testing accuracy at the moderate non-iid degree. It means that too low or high non-iid degrees will be harmful to DPSCAFFOLD for achieving satisfactory model utility. Finally, comparing Fig. 6(c) with (d), we can find that the tendency of the testing accuracies regarding DPFedAvg is the opposite. But it should be noted that for Fig. 6(c), from left to right, the non-iid degree increases and for Fig. 6(d), the non-iid degree is getting lower. Therefore, in Fig. 6(c) and (d), the variations of the testing accuracies on DPFedAvg are still consistent with the previous observation, i.e., when the non-iid degree increases, the model accuracy of DPFedAvg tends to decrease.

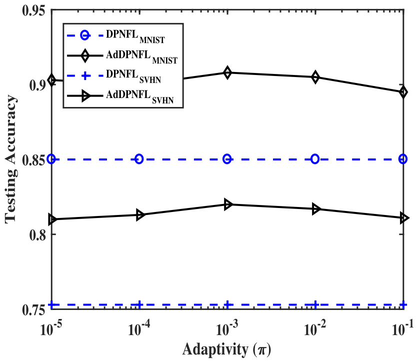


Fig. 7. Impact of adaptivity π on the testing accuracy over unbalanced MNIST and SVHN.

Impact of the Adaptivity Level: For the proposed algorithms, we want to observe the impact of the adaptivity parameter π on the algorithm performance. Fixing the other parameters like ϵ, we run DPNFL and AdDPNFL on the logistic regression model over the unbalanced MNIST dataset and for completeness, we also run these two algorithms on the DNN model over the unbalanced SVHN dataset. We vary the adaptivity parameter from $\pi = 10^{-5}$ to $\pi = 10^{-1}$ and regard DPNFL as the baseline algorithm. From Fig. 7, we can see that our performance is comparatively resilient to π as it varies. Specifically, under different adaptivity parameters, the testing accuracy of AdDPNFL changes a little and keeps around 90% and 81.5% in the MNIST and SVHN dataset, respectively. Without any doubt, the testing accuracies of DPNFL are always constant, which is respectively 85% and 76% in MNIST and SVHN datasets. Over any dataset, $\pi = 10^{-3}$ makes a similar effect compared to other values. It conforms to the analysis in [22], which shows that suitably big π produces the finer results for federated adaptive optimizers. Therefore, using the default value $\pi = 10^{-3}$ in the previous experiments is reasonable.

VIII. Conclusion

In this work, we investigate the differentially private federated learning on non-iid data. To improve the communication efficiency and avoid the sampling variance, we adopt the partial client sampling without replacement method to achieve the aggregation in the non-iid federated learning setting. To better analyze the end-to-end privacy loss, we utilize the latest truncated concentrated DP technique and obtain the tighter privacy loss bounds. Moreover, to further improve the model performance and mitigate the negative effect of the non-iid-ness, we also propose an adaptive algorithm based on server-side adaptive optimization. We provide the rigorous analysis of convergence and privacy for all the algorithms and the extensive experimental results validate their effectiveness.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[2] P. Kairouz et al., “Advances and open problems in federated learning,” *Found. Trends Mach. Learn.*, vol. 14, no. 1/2, pp. 1–210, 2021.

[3] J. Xing, J. Tian, Z. Jiang, J. Cheng, and H. Yin, “Jupiter: A modern federated learning platform for regional medical care,” *Sci. China Inf. Sci.*, vol. 64, no. 10, pp. 1–14, 2021.

[4] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, “Clustered sampling: Low-variance and improved representativity for clients selection in federated learning,” in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 3407–3416.

[5] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 7611–7623.

[6] H. Yang, M. Fang, and J. Liu, “Achieving linear speedup with partial worker participation in non-iid federated learning,” in *Proc. 9th Int. Conf. Learn. Representations*, 2021.

[7] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, “Federated learning with compression: Unified analysis and sharp guarantees,” in *Proc. 24th Int. Conf. Artif. Intell. Statist.*, 2021, pp. 2350–2358.

[8] R. Balakrishnan, T. Li, T. Zhou, N. Himayat, V. Smith, and J. Bilmes, “Diverse client selection for federated learning via submodular maximization,” in *Proc. 10th Int. Conf. Learn. Representations*, 2022.

[9] M. Fredrikson, S. Jha, and T. Ristenpart, “Model inversion attacks that exploit confidence information and basic countermeasures,” in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1322–1333.

[10] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 3–18.

[11] C. Dwork et al., “The algorithmic foundations of differential privacy,” *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3/4, pp. 211–407, 2014.

[12] N. Mohammadi, J. Bai, Q. Fan, Y. Song, Y. Yi, and L. Liu, “Differential privacy meets federated learning under communication constraints,” *IEEE Internet Things J.*, vol. 9, no. 22, pp. 22204–22219, Nov. 2022.

[13] M. Noble, A. Bellet, and A. Dieuleveut, “Differentially private federated learning on heterogeneous data,” in *Proc. 25th Int. Conf. Artif. Intell. Statist.*, 2022, pp. 10110–10145.

[14] Z. Xiong, Z. Cai, D. Takabi, and W. Li, “Privacy threat and defense for federated learning with non-iid data in AIoT,” *IEEE Trans. Ind. Inform.*, vol. 18, no. 2, pp. 1310–1321, Feb. 2022.

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

[16] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of FedAvg on non-iid data,” in *Proc. 8th Int. Conf. Learn. Representations*, 2020.

[17] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “SCAFFOLD: Stochastic controlled averaging for federated learning,” in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[18] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in *Proc. Mach. Learn. Syst.*, 2020, pp. 429–450.

[19] S. P. Karimireddy et al., “Mime: Mimicking centralized stochastic algorithms in federated learning,” 2020, *arXiv: 2008.03606*.

[20] J. Zhang et al., “Why ADAM beats SGD for attention models,” 2019, *arXiv: 1912.03194*.

[21] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, “Federated learning for keyword spotting,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 6341–6345.

[22] S. J. Reddi et al., “Adaptive federated optimization,” in *Proc. 9th Int. Conf. Learn. Representations*, 2021.

[23] S. Wang et al., “Adaptive federated learning in resource constrained edge computing systems,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[24] W. Chen, S. Horvath, and P. Richtarik, “Optimal client sampling for federated learning,” 2020, *arXiv: 2010.13723*.

[25] Y. J. Cho, J. Wang, and G. Joshi, “Towards understanding biased client selection in federated learning,” in *Proc. 25th Int. Conf. Artif. Intell. Statist.*, 2022, pp. 10351–10375.

[26] T. Nishio and R. Yonetani, “Client selection for federated learning with heterogeneous resources in mobile edge,” in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–7.

[27] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, “A general theory for client sampling in federated learning,” in *Proc. 1st Int. Workshop Trustworthy Federated Learn. Conjunction IJCAI*, Springer, 2022, pp. 46–58.

[28] P. Zhou, K. Wang, L. Guo, S. Gong, and B. Zheng, “A privacy-preserving distributed contextual federated online learning framework with Big Data support in social recommender systems,” *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 3, pp. 824–838, Mar. 2021.

[29] L. Zhang, T. Zhu, P. Xiong, W. Zhou, and P. Yu, “A robust game-theoretical federated learning framework with joint differential privacy,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 4, pp. 3333–3346, Apr. 2023.

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

[31] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, “Learning differentially private recurrent language models,” in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[32] K. Wei et al., “Federated learning with differential privacy: Algorithms and performance analysis,” *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, Apr. 2020.

[33] A. Cheng, P. Wang, X. S. Zhang, and J. Cheng, “Differentially private federated learning with local regularization and sparsification,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10112–10121.

[34] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, “Differentially private learning with adaptive clipping,” in *Proc. 35th Conf. Neural Inf. Process. Syst.*, 2021, pp. 17455–17466.

[35] J. Wang and Z.-H. Zhou, “Differentially private learning with small public data,” in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, pp. 6219–6226.

[36] R. Hu, Y. Gong, and Y. Guo, “Federated learning with sparsification-amplified privacy and adaptive optimization,” in *Proc. 13th Int. Joint Conf. Artif. Intell.*, 2021, pp. 1463–1469.

[37] Y. Zhao et al., “Local differential privacy-based federated learning for Internet of Things,” *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8836–8853, Jun. 2021.

[38] M. Abadi et al., “Deep learning with differential privacy,” in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2016, pp. 308–318.

[39] C. Dwork and G. N. Rothblum, “Concentrated differential privacy,” 2016, *arXiv:1603.01887*.

[40] M. Bun and T. Steinke, “Concentrated differential privacy: Simplifications, extensions, and lower bounds,” in *Proc. 14th Int. Conf. Theory Cryptogr.*, 2016, pp. 635–658.

[41] I. Mironov, “Rényi differential privacy,” in *Proc. 30th IEEE Comput. Secur. Found. Symp.*, 2017, pp. 263–275.

[42] M. Bun, C. Dwork, G. N. Rothblum, and T. Steinke, “Composable and versatile privacy via truncated CDP,” in *Proc. 50th Annu. ACM SIGACT Symp. Theory Comput.*, 2018, pp. 74–86.

[43] J. Hong, H. Wang, Z. Wang, and J. Zhou, “Learning model-based privacy protection under budget constraints,” in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 7702–7710.

[44] S. U. Stich, “Local SGD converges fast and communicates little,” in *Proc. 7th Int. Conf. Learn. Representations*, 2019.

[45] J. Wang and G. Joshi, “Cooperative SGD: A unified framework for the design and analysis of local-update SGD algorithms,” *J. Mach. Learn. Res.*, vol. 22, no. 213, pp. 1–50, 2021.

[46] J. Xu, W. Zhang, and F. Wang, “A $(\mathrm{DP})^2$SGD: Asynchronous decentralized parallel stochastic gradient descent with differential privacy,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 8036–8047, Nov. 2022.

[47] Z. Zhou, Y. Li, X. Ren, and S. Yang, “Towards efficient and stable K-asynchronous federated learning with unbounded stale gradients on non-IID data,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 3291–3305, Dec. 2022.

[48] Y. Ruan, X. Zhang, S.-C. Liang, and C. Joe-Wong, “Towards flexible device participation in federated learning,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 3403–3411.

[49] T.-M. H. Hsu, H. Qi, and M. Brown, “Measuring the effects of non-identical data distribution for federated visual classification,” 2019, *arXiv: 1909.06335*.

[50] F. Haddadpour and M. Mahdavi, “On the convergence of local descent methods in federated learning,” 2019, *arXiv: 1910.14425*.

[51] Z. Qu, K. Lin, Z. Li, and J. Zhou, “Federated learning’s blessing: FedAvg has linear speedup,” in *Proc. ICLR Workshop Distrib. Private Mach. Learn.*, 2021.

[52] H. Yang, X. Zhang, P. Khanduri, and J. Liu, “Anarchic federated learning,” in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 25331–25363.

[53] W. Liu, L. Chen, and W. Zhang, “Decentralized federated learning: Balancing communication and computing costs,” *IEEE Trans. Signal Inf. Process. Netw.*, vol. 8, pp. 131–143, Feb. 2022.

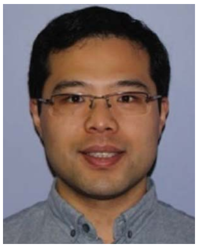
[54] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, “Subsampled Rényi differential privacy and analytical moments accountant,” in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1226–1235.

[55] I. Mironov, K. Talwar, and L. Zhang, “Rényi differential privacy of the sampled Gaussian mechanism,” 2019, *arXiv: 1908.10530*.

[56] N. Ponomareva et al., “How to DP-fy ML: A practical guide to machine learning with differential privacy,” *J. Artif. Intell. Res.*, vol. 77, pp. 1113–1201, 2023.

Lin Chen received the ME degree in control engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018. He is currently working toward the PhD degree with the School of Computer Science and Technology, HUST, Wuhan, P.R. China. His current research interests include machine learning, online learning, and privacy security.

Xiaofeng Ding (Member, IEEE) received the PhD degree in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2009. He is currently working as a professor with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan. His research interests mainly include data privacy and query processing, data encryption.



Zhifeng Bao received the PhD degree in computer science from the National University of Singapore, in 2011 as the winner of the Best PhD Thesis with the School of Computing. He is currently a professor with the RMIT University and leads the Big Data Research Group, RMIT. He is also an honorary fellow with the University of Melbourne in Australia. His current research interests include data usability, spatial database, data integration, and data visualization.

Pan Zhou (Senior Member, IEEE) received the PhD degree from the Georgia Institute of Technology (Georgia Tech), in 2011. He is currently a full professor and PhD advisor with the School of Cyber Science and Engineering, HUST. Also, he is currently an associate editor of *IEEE Transactions on Network Science and Engineering*. His research interests include security and privacy, Big Data analytics, machine learning, and information networks.

Hai Jin (Fellow, IEEE) received the PhD degree in computer engineering from the Huazhong University of Science and Technology (HUST), China, in 1994, where he is currently the Cheung Kong professor with the School of Computer Science and Technology. His research interests include distributed computing, computer architecture, Big Data privacy and security, crowdsourcing, network storage, and network security. He is an ACM member.