Projected Federated Averaging with Heterogeneous Differential Privacy

Junxu Liu*
Renmin University of China
junxu_liu@ruc.edu.cn

Jian Lou†
Xidian University
jlou@xidian.edu.cn

Li Xiong
Emory University
lxiong@emory.edu

Jinfei Liu‡
Zhejiang University
jinfeiliu@zju.edu.cn

Xiaofeng Meng§
Renmin University of China
xfmeng@ruc.edu.cn

ABSTRACT

Federated Learning (FL) is a promising framework for multiple clients to learn a joint model without directly sharing the data. In addition to high utility of the joint model, rigorous privacy protection of the data and communication efficiency are important design goals. Many existing efforts achieve rigorous privacy by ensuring differential privacy for intermediate model parameters, however, they assume a uniform privacy parameter for all the clients. In practice, different clients may have different privacy requirements due to varying policies or preferences.

In this paper, we focus on explicitly modeling and leveraging the heterogeneous privacy requirements of different clients and study how to optimize utility for the joint model while minimizing communication cost. As differentially private perturbations affect the model utility, a natural idea is to make better use of information submitted by the clients with higher privacy budgets (referred to as "public" clients, and the opposite as "private" clients). The challenge is how to use such information without biasing the joint model. We propose **P**rojected **F**ederated **A**veraging (PFA), which extracts the top singular subspace of the model updates submitted by "public" clients and utilizes them to project the model updates of "private" clients before aggregating them. We then propose communication-efficient PFA+, which allows "private" clients to upload projected model updates instead of original ones. Our experiments verify the utility boost of both algorithms compared to the baseline methods, whereby PFA+ achieves over 99% uplink communication reduction for "private" clients.

PVLDB Reference Format:
Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. Projected Federated Averaging with Heterogeneous
Differential Privacy . PVLDB, 15(4): 828-840, 2022.
doi:10.14778/3503585.3503592

*Work partially done while at Emory University.
†Work partially done while at Emory University.
‡Work partially done while at Emory University and Georgia Tech.
§Corresponding author: Xiaofeng Meng.

Proceedings of the VLDB Endowment, Vol. 15, No. 4 ISSN 2150-8097.
doi:10.14778/3503585.3503592

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/Emory-AIMS/PFA.

1 INTRODUCTION

Federated learning (FL) [6, 19, 21, 32] has attracted substantial attention in recent years, since it provides a promising approach for multiple devices (e.g., smartphones, IoT devices) or institutions (e.g., hospitals or banks) to collaboratively learn a joint model from the sensitive local data. The latter is also known as cross-silo federated learning, which will be the focus of this paper. We consider the horizontally partitioned setting in which the institutions have their own sets of data subjects (e.g. patients or customers) with the same set of features and wish to learn a joint model. For example, a formidable problem common for clinical studies is the inadequate number of subjects of a given condition (e.g. rare disease), esp. at a single institution. Cross-institutional learning is desired to mitigate the data shortage or bias and learn a more accurate and generalizable model. Figure 1 shows the typical FL framework where clients upload local model updates to a semi-trusted coordinating server which in turn aggregates the local models into a global model through multiple rounds of iterations.

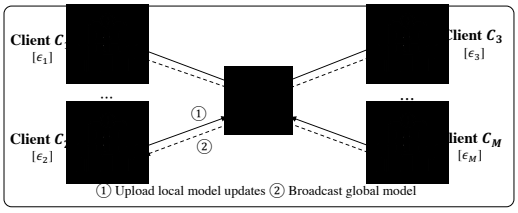


Figure 1: Framework of federated learning.

In addition to high utility of the joint model, rigorous protection of the sensitive data and communication efficiency are among the key design desiderata of a successful FL algorithm. Privacy is a key design consideration that motivates the FL framework, which requires each client's raw data to be stored locally and only intermediate model updates are transferred during collaborative training. By avoiding direct data exchange, it reduces the risk of sensitive information disclosure. However, the transferred intermediate variables can still disclose sensitive user information via advanced

adversarial algorithms and have been shown to be vulnerable to various privacy attacks such as reconstruction attacks and membership inference attacks [17, 36, 37, 40, 47]. To further enhance the privacy-preserving capability, differential privacy (DP) [11] has been introduced to ensure rigorous privacy protection for the intermediate model updates [14, 19, 41]. However, existing works assume a uniform privacy budget for all the clients. In practice, different institutions may have different privacy requirements due to varying privacy policies or varying privacy preferences of the data subjects (e.g. different demographic groups), which leads to heterogeneous privacy budgets across institutions. An open question is how to adapt the FL algorithms for better trade-offs among model utility and communication efficiency while allowing heterogeneous privacy guarantees for the clients.

We formalize the heterogeneous differentially private FL problem, where the privacy budgets are varying from institution to institution. The central goal is to exploit the heterogeneous privacy budgets to optimize utility during the federated training, while minimizing the uplink communication cost. To ensure DP, the utility of the joint model inevitably decreases due to the perturbations injected on local model updates submitted to the server. Informally, the "private" institutions with stricter privacy budgets require larger perturbation, leading to less accurate model updates, while the "public" clients with more relaxed privacy budgets submit more accurate model updates. Thus, a natural idea is to extract more useful information from the "public" clients' model updates. While intuitive, *the challenge is to decide which "right" information to extract and how to use it without access to the raw local data*, because we want neither the case where too much public institutions' information biases the joint model, nor the case where too less information does not help improve the joint model utility.

Previous work [15, 28, 35] have observed that the stochastic gradients of the model parameters stay in a low-dimensional space along the training trajectory in a typical Stochastic Gradient Descent (SGD) based learning algorithm and DPSGD algorithms. Motivated by this, our main idea is to extract the top singular subspace of the model updates submitted by "public" clients as such "right" information, i.e., a lower dimensional subspace extracted from the model updates. Since the model updates submitted by "private" clients contain more noise, projecting them onto the subspace from "public" clients will allow us to extract and aggregate the most useful information while mitigating the impact of the noise, hence higher utility for the joint model could be achieved. Simultaneously, the lower-dimensional subspace makes it possible for the "private" clients to only upload projected model updates in the reduced subspace, hence achieving significant communication reduction.

Contributions. In this paper, we introduce the definition of heterogeneous DP in FL (FL-HDP), and propose a novel projection based federated averaging (PFA) algorithm that achieves FL-HDP by perturbing each client's local update in a customized way. To fully utilize the model updates with heterogeneous perturbation noises, PFA exploits the top singular subspace extracted from the intermediate local updates of the "public" clients for aggregating the global updates with enhanced utility. We further propose a communication-efficient variant PFA+ in which "private" clients use subspace from previous model updates to upload projected model parameters instead of the original ones, with minimal impact on model utility. The contributions are summarized as follows.

(1) We formalize the problem of heterogeneous DP in FL which allows different institutions to have different privacy requirements.
(2) We first propose the basic weighted average (WeiAvg) approach that considers the varying privacy budgets of clients. The clients with tighter privacy budgets are assigned a larger weight when averaging the model updates, and vice versa. Then we propose an advanced projected federated average (PFA) approach, which partitions the clients into "public" and "private" ones with more relaxed and strict privacy budgets respectively, and utilizes the top singular subspace of the "public" clients for projecting the model updates before aggregating them, leading to noise reduction and higher utility compared with the standard FedAvg and the basic WeiAvg.
(3) We further propose a communication-efficient variant PFA+, which applies projected federated averaging in an asynchronous way and allows "private" clients to upload only projected model updates in the reduced subspace, and achieves significant communication reduction with minimal impact on utility.
(4) We provide a formal convergence analysis showing that PFA can achieve the same convergence rate as the non-private FedAvg under the same assumptions. We also show communication and computation complexity analysis of PFA.
(5) We perform a comprehensive evaluation on both statistical learning and deep learning models using several real world datasets from different domains. The results demonstrate that PFA and PFA+ greatly improve the utility of the joint model over the baseline methods in different settings. In addition, PFA+ achieves over 99% uplink communication reduction for "private" clients with comparable utility.

Organization. The rest of the paper is structured as follows. Section 2 reviews the related work. Section 3 describes the preliminaries. Section 4 presents the proposed methods. Section 5 presents the experiments. Finally, Section 6 draws a conclusion and discusses future directions.

2 RELATED WORK

2.1 Federated Learning

Federated learning (FL) is a form of decentralized learning with the goal of collaboratively training a global model from data stored in multiple remote clients [27]. It has received significant interest in recent years, owing to its distinct privacy advantages compared to the traditional centralized learning. However, there are also open challenges such as the communication and computation overheads, data and model heterogeneity, and unreliable or potentially malicious participating clients or servers [2, 19, 27, 30].

Federated optimization methods. One of the most widely used approaches for federated optimization problem is federated averaging (FedAvg) [32], in which a randomly sampled subset of clients run a certain number of Stochastic Gradient Descent (SGD) steps locally and independently and the central server then averages the local models to update the global model. While FedAvg has

emerged as a *de facto* standard algorithm, it has several limitations in overcoming the inherent challenges of FL, such as the communication and computation overheads, data heterogeneity, and constraints on the model architectures [2, 19, 27, 30]. To overcome these issues, some works approach FL in new perspectives. For example, Lin et al. [30] propose a distillation framework for robust federated model fusion (FedDF), which enables FL in more general settings such as heterogeneous client models and data. Al-Shedivat et al. [2] formulate FL as a posterior inference problem, and propose an approximate algorithm FedPA that runs local and global posterior inference on the clients and server side respectively.

Differentially private federated learning. Attempts to integrate differential privacy into federated learning for enhanced privacy protection have been made previously. Notably, Geyer et al. [14] achieve client-level differential privacy by adding random Gaussian noise on the aggregated local model updates to hide a single client's information. McMahan et al. [33] propose the user-level or sample-level differential privacy by allowing each client to perturb its computed gradients locally and upload the noisy model updates to the server. User-level differential privacy ensures that the intermediate model parameters and final model are indistinguishable regardless of the presence or absence of any given record (user) at each client. Other works also consider the combination of DP and blockchain [4], or add DP noise on both uplink and downlink communication channels [41]. In this work, we adopt the user-level differential privacy while allowing heterogeneous privacy parameters to be used at each client.

2.2 Heterogeneous Differential Privacy

There is rich literature exploring differentially private mechanism design considering heterogeneous privacy preferences. Most of them are focused on centralized setting [3, 18, 26, 34], where each user holds a single data entry and is associated with a personalized privacy parameter. For the case of cross-silo FL, each client contributes multiple data entries, and can be associated with a different client specific privacy parameter. We refer to this as client-level heterogeneous differential privacy to make a distinction.

User-level personalized differential privacy. The line of work on user-level personalized differential privacy in centralized setting starts by Alaggan et al. [3] who first introduced the notion of *heterogeneous differential privacy*. Another independent work by Jorgensen et al. [18] introduced a similar notion of *personalized differential privacy* (PDP). PDP considers the case where each user or record is associated with its own privacy parameter. They proposed two mechanisms based on non-uniform sampling and exponential mechanism, which are applicable to more common queries compared with [3]. Following the above two works, an extensive set of methods has been developed for adopting personalized privacy guarantees in different scenarios, more complex tasks, or persuing better data utility [9, 22, 26, 29, 34, 43, 45].

Client-level heterogeneous differential Privacy. There is limited work considers heterogeneous privacy preferences in the decentralized setting where each client is associated with its own privacy parameter. This can be due to the fact that each client has its own privacy policy or the client allows its users to specify personalized parameters. Given the heterogeneous privacy parameters from different clients, a straightforward approach is to design the learning algorithm to satisfy ϵ_{min}-DP, with ϵ_{min} as the minimum privacy budget across all clients [33], then apply conventional algorithms designed for uniform DP. Obviously, this approach could lead to a large amount of under-utilized privacy budgets.

Recently, an independent work by Chathoth et al. [8] has made efforts for addressing cohort-level privacy heterogeneity motivated by the IoT setting. More specifically, all clients are first grouped into cohorts and Gaussian noises are added on the aggregation of local updates from the clients in each cohort. In their approach, the privacy budgets used at each communication round are still identical for all cohorts. By a privacy accountant, clients with stricter privacy requirements will end sooner than those of more relaxed privacy requirements. Thus they adopted continual-learning based methods to ensure the model utility. In contrast, we consider client-level heterogeneous DP which is better motivated in our cross-silo FL setting. In addition, we use heterogeneous privacy parameters for the clients for each round and develop a projection-based approach for enhanced utility and community efficiency.

2.3 Gradient Projection

Previous work have observed that stochastic gradients stay in a low-dimensional space along the training trajectory of SGD [15, 28, 35]. Zhou et al. [46] further studied the noisy gradients and proposed a variant of DPSGD algorithm, named Projected DPSGD (PDPSGD), for reaching a better privacy-utility trade-off in deep learning. However, these observations mostly focused on the stochastic gradients in the centralized setting, we empirically prove that the projection-based approach can be adapted to the *noisy* model updates in the FL setting (see Figure 13 with its related description in Appendix B), and first exploit its benefit for both (heterogeneous) noise reduction and communication reduction in the federated setting. The key differences between our method and PDPSGD are discussed in details in Appendix D.1.

3 PRELIMINARIES

In this section, we will give a brief review of the related definitions and notations used in the paper. Table 2 in Appendix A summarizes the frequently used notations.

3.1 Federated Learning

Federated Learning Formulation. We focus on a typical cross-silo horizontally partitioned FL setting: there is a central coordinating server and M clients with each holding a local dataset with the same set of features drawn from an unknown data distribution denoted by $\mathcal{P}_m,\ m \in [M]$. The aim is to jointly optimize the objective function denoted by $\mathcal{L}(\mathrm{x})$, where x is the trainable model parameters.

Definition 1 (Federated Learning Formulation). The global empirical loss $\mathcal{L}(\mathrm{x})$ is taking the weighted average of the per-client

loss $\mathcal{L}_m(\mathrm{x})$ over all clients, i.e.,

$$\mathcal{L}(\mathrm{x}) \triangleq \sum_{m=1}^{M} w_m \mathcal{L}_m(\mathrm{x}), \quad s.t. \sum_{m=1}^{M} w_m = 1, \text{where } w_m = \frac{N_m}{N} \quad (1)$$

and $\mathcal{L}_m(\mathrm{x})$ is the local loss of client C_m on the local dataset defined as follows,

$$\mathcal{L}_m(\mathrm{x}) \triangleq \frac{1}{N_m} \sum_{i=1}^{N_m} \ell(\mathrm{x}, \xi_i), \quad (2)$$

where $\mathcal{D}_m$ with size N_m is the local dataset with each sample $\{\xi_1, \ldots, \xi_{N_m}\}$ drawn from $\mathcal{P}_m$.

Federated Averaging. Federated averaging (FedAvg) [32] is a basic algorithm for optimizing eq.(1), which is shown in Algorithm 6 in Appendix B. More specifically, at each communication round t_c, a random subset of clients $\mathcal{S}_{t_c}$ run some number of local update iteration steps (e.g., SGD steps) based on their local dataset $\mathcal{D}_m$, and upload the local model updates to the central server which aggregates the collected local information and updates the joint model via a simple average operation. There are three distinct features. For privacy consideration: 1) raw dataset $\mathcal{D}_m$ for $m \in [M]$ are kept local and only intermediate local model updates are communicated; For communication efficiency: 2) partial client participation in Step 4 is used, where only a sub-sample $\mathcal{S}_{t_c}$ of clients participate in round t_c by either server sampling or client volunteering; and 3) periodic communication is used, i.e. participating clients run multiple local updates, rather than one, before submitting the local updates to the server.

3.2 Differential Privacy

Definition of Differential Privacy. Differential Privacy (DP) [11], as a formal mathematical framework providing rigorous privacy protection, has been extensively incorporated into a broad range of data analysis tasks over the last decade, ranging from simple statistical estimations to complicated machine learning algorithms [1, 12, 14].

Definition 2 ((ϵ, δ)**-Differential Privacy [12]**). A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathbb{R}$ with domain $\mathcal{D}$ and range $\mathbb{R}$ satisfies (ϵ, δ)-differential privacy ((ϵ, δ)-DP) if for any two neighboring datasets $D, D' \in \mathcal{D}$ that differ only by a single record, and for any subsets of outputs $S \subseteq \mathbb{R}$, it holds that

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \in S] + \delta. \quad (3)$$

It ensures that the output of a DP algorithm is indistinguishable regardless of the presence or absence of a record, even in the worst-case scenario where the adversary has known all the records except one. The parameter ϵ, which is also known as the *privacy budget*, manifests the upper bound of the privacy loss. The smaller ϵ, the less the privacy loss. The other parameter $\delta \geq 0$ captures the probability that the ϵ-differential privacy (ϵ-DP) [11] is broken. Given a specified ϵ, the small δ, the less the probability of privacy leakage. And when $\delta = 0$, (ϵ, δ)-DP is also known as the pure ϵ-DP.

Composability of Differential Privacy. An appealing feature of DP is that it allows a complex DP algorithm to be built from much simpler building blocks whose DP is easier to analyze, as elaborated in the following two properties.

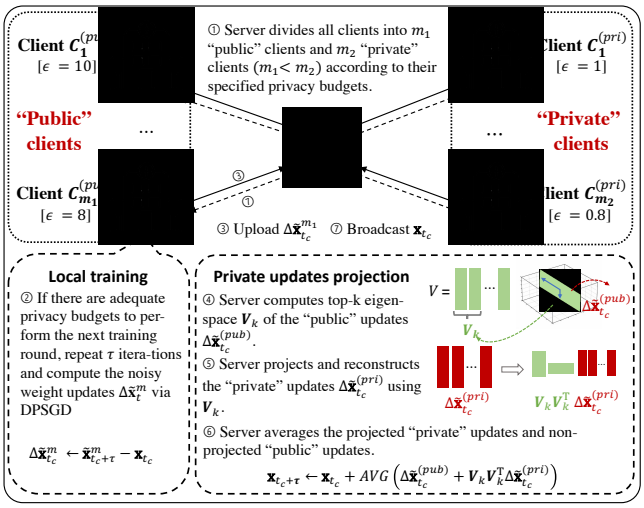


Figure 2: Framework of projected federated averaging with heterogeneous differential privacy (Algorithm 2).

Property 1 (**Sequential Composition[10]**). *Let the randomized mechanism $\mathcal{M}_i : \mathcal{D} \to \mathbb{R}$ each satisfies (ϵ_i, δ_i)-DP. The sequence of $\{\mathcal{M}_i\}_{i\in[M]}$ satisfy $(\sum_i \epsilon_i, \sum_i \delta_i)$-DP.*

Property 2 (**Parallel Composition[44]**). *Let the randomized mechanism $\mathcal{M}_i : \mathcal{D}_i \to \mathbb{R}$ each satisfies (ϵ_i, δ_i)-DP, where the sequence of $\{\mathcal{D}_i\}_{i\in[M]}$ are disjoint subsets of domain $\mathcal{D}$. Any functions g on the sequence of $\{\mathcal{M}_i\}_{i\in[M]}$, i.e., $g(\mathcal{M}_1, \mathcal{M}_2, \ldots)$ satisfy $(\max_i \epsilon_i, \max_i \delta_i)$-DP.*

Both of the above two properties are important and practical to analyze the *cumulative privacy loss* incurred by composing multiple DP mechanisms.

DP Stochastic Gradient Descent. A major milestone on DP machine learning is the DP stochastic gradient descent (DPSGD) [1] algorithm, which becomes a fundamental building block for training deep learning models with DP. DPSGD adds noise on the clipped gradients during each training iteration to ensure that each gradient descent is differentially private, and thus the final model also satisfies differential privacy due to the composition properties.

Moments Accountant. Prior work [1] points out the naive sequential composition and their extensions (e.g., the strong composition theorem[13]) cannot estimate the privacy loss accumulation for the iterative SGD training process tightly. To solve this problem, the moments accountant (MA) technique is proposed, which keeps track of the log moments of the privacy loss variable under the Poisson subsampling, and can be seen as a tighter variant of sequential composition. For completeness, we introduce MA definition and properties in Supplement. In this work, we will apply the moments accountant technique to trace the privacy loss during each client's local training process and terminate before the loss exceeds the target privacy budget.

Algorithm 1: Weighted Averaging with Heterogeneous Differential Privacy (WeiAvg)

input : clients' privacy preferences $\{(\epsilon_m, \delta_m)\}_{m\in[M]}$, total rounds T
output : global model x_T

$\mathrm{x}_0 \leftarrow$ (Initialize randomly)
for $t' = 0$ *to* $\lfloor \frac{T-1}{\tau} \rfloor$ **do**
  $t_c \leftarrow t'\tau$
  $\mathcal{S}_{t_c} \leftarrow$ (random subset of K clients)
  foreach *client* $C_m \in \mathcal{S}_{t_c}$ **do** *in parallel*
    $\Delta\mathrm{x}_{t_c}^m \leftarrow$ **LocalDPSGD**(x_{t_c})
  $\Delta\bar{\mathrm{x}}_{t_c} \leftarrow \sum_{m\in[K]} \mathcal{W}_{\mathcal{S}_{t_c}}(\epsilon_m) \cdot \Delta\mathrm{x}_{t_c}^m$
  $\mathrm{x}_{t_c+\tau} \leftarrow \mathrm{x}_{t_c} - \Delta\bar{\mathrm{x}}_{t_c}$
return x_T

Algorithm 2: Projected Federated Averaging with Heterogeneous Differential Privacy (PFA)

input : Clients' privacy preferences $\{(\epsilon_m, \delta_m)\}_{m\in[M]}$, total rounds T
output : global model x_T

$\mathrm{x}_0 \leftarrow$ (Initialize randomly)
for $t' = 0$ *to* $\lfloor \frac{T-1}{\tau} \rfloor$ **do**
  $t_c \leftarrow t'\tau$
  $\mathcal{S}_{t_c} \leftarrow$ (random subset of K clients)
  $\mathcal{S}_{t_c}^{(pub)}, \mathcal{S}_{t_c}^{(pri)} \leftarrow$(Partition $\mathcal{S}_{t_c}$ into two groups according to clients' privacy requirements)
  foreach *client* $C_m \in \mathcal{S}_{t_c}$ **do** *in parallel*
    $\Delta\mathrm{x}_{t_c}^m \leftarrow$ **LocalDPSGD**(x_{t_c})
  $\Delta\bar{\mathrm{x}}_{t_c} \leftarrow$ **ProjectedFedAvg**$(\{\Delta\mathrm{x}_{t_c}^m\}_{m\in[K]}, \mathcal{S}_{t_c}^{(pub)}, \mathcal{S}_{t_c}^{(pri)})$
  $\mathrm{x}_{t_c+\tau} \leftarrow \mathrm{x}_t - \Delta\bar{\mathrm{x}}_{t_c}$
return x_T

4 PROPOSED METHODS

In this section, we first formalize the federated learning with heterogeneous DP problem (Subsection 4.1) and describe the overall framework with local privacy mechanisms (Subsection 4.2). Then, we identify the key challenge: *How to pinpoint the key indicative information from the "public clients" that helps improve the overall utility of the global model without biasing it?* To answer the question, we propose three attempts, which are main technical contributions of the paper:

(1) *WeiAvg* (Subsection 4.3): A natural yet naive idea is to assign larger weight for the "public" clients while curtailing the weight for the "private" clients with hope that the less perturbed "public" updates will improve the utility of the global model; a potential limitation of this approach is that the influence from the "public" clients may dominate the "private" ones and bias the global model;

(2) *PFA* (Subsection 4.4): A more effective way to "reweight" the two types of clients is to make use of the leading singular space of "public" updates and project "private" updates onto it, which discards the useless information incurred by heavy private perturbation of the "private" updates while still keeping the most essential information lied along the top singular space;

(3) *PFA+* (Subsection 4.5): By following the observation that the top singular space evolves slowly between two consecutive rounds, we further propose to use delayed projection to aggressively reduce the uplink communication cost for the "private" clients without sacrificing utility.

Algorithm 3: Projected Federated Averaging

input : Noisy model updates $\{\Delta\mathrm{x}^m\}_{m\in[K]}$, "public" clients $\mathcal{S}^{(pub)}$, "private" clients $\mathcal{S}^{(pri)}$
output : the aggregated model updates $\Delta\bar{\mathrm{x}}$

Function ProjectedFedAvg$(\{\Delta\mathrm{x}^m\}_{m\in[K]}, \mathcal{S}^{(pub)}, \mathcal{S}^{(pri)})$:
  $\Delta\bar{\mathrm{x}}^{pub} \leftarrow \sum_{m\in\mathcal{S}^{(pub)}} \mathcal{W}_{\mathcal{S}^{(pub)}}(\epsilon_m) \cdot \Delta\mathrm{x}^m$
  $\Sigma \leftarrow \Delta\bar{\mathrm{x}}^{(pub)} \cdot (\Delta\bar{\mathrm{x}}^{(pub)})^\top$
  $\mathrm{V}_k \leftarrow$ (Compute the top-k eigenvectors of Σ)
  // Project the "private" updates
  $\Delta\bar{\mathrm{x}}^{(pri)} \leftarrow \sum_{m\in\mathcal{S}^{(pri)}} \mathcal{W}_{\mathcal{S}^{(pri)}}(\epsilon_m) \cdot \Delta\mathrm{x}^m$
  $\Delta\hat{\mathrm{x}}^{(pri)} \leftarrow \mathrm{V}_k\mathrm{V}_k^\top \Delta\bar{\mathrm{x}}^{(pri)}$
  // Projected federated averaging
  $\mathcal{S} \leftarrow \mathcal{S}^{(pub)} + \mathcal{S}^{(pri)}$
  $\Delta\bar{\mathrm{x}} \leftarrow \frac{\sum_{m\in\mathcal{S}^{(pub)}} \epsilon_m}{\sum_{m\in\mathcal{S}} \epsilon_m} \cdot \Delta\bar{\mathrm{x}}^{(pub)} + \frac{\sum_{m\in\mathcal{S}^{(pri)}} \epsilon_m}{\sum_{m\in\mathcal{S}} \epsilon_m} \cdot \Delta\hat{\mathrm{x}}^{(pri)}$
  return $\Delta\bar{\mathrm{x}}$

4.1 Federated Learning with Heterogeneous Differential Privacy

We first formalize heterogeneous DP for federated learning.

Definition 3 (Heterogeneous Differential Privacy in Federated Learning (FL-HDP)). Let the set of clients be $C = \{C_1, \ldots, C_M\}$, where each client $C_m \in C$ holds a local dataset $\mathcal{D}_m$. The federated learning satisfies $\{(\epsilon_m, \delta_m)\}_{m\in[M]}$-heterogeneous differential privacy, if each client satisfies (ϵ_m, δ_m)-differential privacy with respect to its local dataset.

Remark 1. In the cross-silo FL setting where each client has multiple users' data samples, the definition ensures user-level (or sample-level) DP for all the local datasets at each client while allowing client-level privacy customization. Each client can set ϵ_m based on its privacy policy or even allow their users to specify their own privacy preferences, i.e., *user-level* personalized DP. In the latter case, each client can use the minimum value over all the users as its privacy preference ϵ_m. While we focus on client-level heterogeneity, an interesting future research is to develop algorithms that achieve personalized DP for different users at each client with enhanced utility. We also assume the privacy preference of each client is public to the central server.

A naive approach: minimum mechanism. The *minimum* mechanism can be also applicable for the FL, i.e. using the minimum of ϵ_m of all clients, and then leveraging existing DP federated learning methods designed for uniform differential privacy. However, this approach would obviously lead to a large amount of underutilized privacy budgets and suboptimal utility, especially when there is a great diversity of privacy preferences among all clients. Hence, a more reasonable idea is to ensure the accumulated privacy loss of each client is bounded by the client-specific privacy budget ϵ_m, and develop new DP federated learning methods dedicated to the heterogeneous DP.

Overall Framework. An illustration of our overall proposed framework is provided in Figure 2. The basic idea is that the server partitions the clients into "public" and "private" clients according to their privacy budgets. We denote the total number of SGD iterations by T. At each communication round t_c, the server randomly samples a subset of K ($K \leq M$) clients $\mathcal{S}_{t_c}$ and broadcasts

the current global model x_{t_c} to all the participants. Each participating client $C_m \in \mathcal{S}_{t_c}$ runs τ local SGD steps independently with DP and then uploads noisy model updates to the server. The server then aggregates the noisy model updates to form the global model based on our proposed algorithms tailored to the heterogeneous DP including WeiAvg, PFA and PFA+ (the figure illustrate the PFA algorithm for the aggregation component).

Remark 2. Considering the diversity of individuals and organizations' privacy preferences, it is realistic to expect a small fraction of the clients may choose a larger privacy budget (i.e., a weaker privacy protection), especially when there are incentives such as in exchange for financial compensation or a more effective model. For simplicity, we consider each client's privacy budget as a random variable that follows an unknown distribution. (e.g., Gaussian mixture distribution, Pareto distribution, etc). For separating the "public" and "private" clients, clustering analysis (e.g., Gaussian mixture model) can be used to group the clients into two clusters. We can also choose the top-K clients in terms of the privacy budgets as the "public" clients, especially if the potential distribution is relatively uniform, which means the borders between the two types of clients are fuzzy, where the value of K tends to be small and depends on the applications.

4.2 Differentially Private Local Update

In this part, we present the client side DP mechanism and analysis, which are common components shared by all three algorithms. To ensure DP for the model updates for each client, we adapt the standard DPSGD algorithm [1] to federated setting and apply the moment accountant (MA) [1] for keeping track of the accumulated privacy loss at each client over the course of training.

Local DPSGD. For participating clients in each round, they perform a number of local SGD steps before uploading the model updates to the server. It is fairly straightforward to integrate the DPSGD algorithm into the local update procedure with well-designed Gaussian noise added on the gradients in each iteration in order to achieve user-level DP on the model updates to be uploaded to the server.

Budget Accountants. An important task for achieving FL-HDP is to keep track of the usage of privacy budget for each of the clients along the course of training. Once the privacy budget runs out, clients can opt out from the remaining training. In our work, we leverage the moment accountant technique and introduce a monitoring module, called the *budget accountant*, which is in charge of privacy budget accounting: (1) *Pre-check* at the beginning of the communication round if a client has sufficient privacy budget to participate in the current round; (2) *Compute and update* the accumulated privacy loss of a client after the current communication round is over.

Privacy budget per round. Based on the above modules, there are two feasible strategies for determining the privacy budget to use for each round.

(1) *Uniform budgets*: we enforce the same privacy budget for all the clients for each round, i.e. the additive Gaussian noises of all clients are drawn from an *identical* distribution. In this case, those clients with a smaller privacy budget will spend their privacy budgets relatively quickly and terminate the training earlier, while the others will continue participating in the following training until there are no sufficient clients left for the next communication round.

(2) *Heterogeneous budgets*: we use privacy budget for each client in each round proportional to their overall privacy budget ϵ_m, i.e. the additive Gaussian noises of the clients are drawn from *different* distributions determined by their privacy budget. For each of the clients, since its total privacy budget should be allocated to multiple local SGD steps, a small privacy budget will result in a large noise per iteration, and vice versa. In this case, all clients' privacy budgets will run out roughly at the same time.

For both of these strategies, aggregating the noisy local updates in a straightforward way (i.e., federated averaging [32]) to form a global model would inevitably lead to a biased estimator of the model updates and suboptimal model utility. For the former, the error is induced by *catastrophic forgetting* [20], that is, the knowledge from clients who stay until the end could overwrite the knowledge from clients who terminate early. For the latter, the error is induced by the additive *Gaussian mixture noise* if a simple average function is applied for model aggregation. In this paper, we focus on the second strategy (as shown in Alg. 7 in the supplement) and propose effective aggregation methods for error correction in Subsection 4.3 and 4.4.

4.3 WeiAvg: Weighted Averaging

To solve inherent error accumulation induced by the additive Gaussian mixture noise and make a better trade-off between privacy with heterogeneous privacy budgets and model utility, we first propose an intuitive **Wei**ghted **Av**erag**i**ng (WeiAvg) approach which allows each of the clients to contribute proportionately to their privacy budgets to the aggregated model. Technically, we define the weight function $\mathcal{W} : \mathbb{R} \to \mathbb{R}$ w.r.t clients' privacy budgets ϵ_m as follows:

$$\mathcal{W}_{\mathcal{S}}(\epsilon_m) = \frac{\epsilon_m}{\sum_{m \in \mathcal{S}} \epsilon_m}, \tag{4}$$

where $\mathcal{S}$ denotes a subset of sampled clients. Thus, the updates with a greater magnitude of noise will get a smaller weight, alleviating their negative impact on the utility of the aggregated model.

Our experimental results in Section 5 show that in most cases, the weighted averaging achieves obvious improvement compared with the federated averaging. However, such a straightforward amplification of the influence from the "public" clients' updates has a risk of dominating the "private" clients and biasing the global model, which will be illustrated in the experiments. Although suboptimal, WeiAvg provides us two important insights motivating us to design a better aggregation strategy for the FL-HDP problem:

(1) On the one hand, the superiority of WeiAvg over vanilla FedAvg indicates that making better use of more "public" clients update indeed help improve the utility of the global model.

(2) On the other hand, it is crucial to extract the *right piece of information* from "public" clients in order to avoid "public" clients dominating the "private" clients.

4.4 PFA: Projected Federated Averaging

In this section, we propose an advanced approach called **P**rojected **F**ederated **A**veraging with heterogeneous DP (PFA) in order to make the most of every noisy local updates and obtain an optimal noise-reduced aggregated model under heterogeneous privacy budgets conditions. The key idea is to extract the top singular space from the "public" clients' updates and project the "private" clients' updates onto the extracted low-rank space. PFA has two advantages over directly amplifying the influence of "public" clients' updates via WeiAvg:

(1) WeiAvg does not mitigate the noisy information contained in "public" clients' updates (recall that "public" clients still introduce DP noise, although smaller than "private" clients), while PFA filters out the noisy information in the bottom singular space and keeps only the useful information possessed by the top singular space.

(2) WeiAvg curtails the overall information including both useful and noisy components from "private" clients' updates, while PFA keeps the useful components by projection based on the observation that the useful component tends to lie on the common low-rank subspace shared by both "private" and "public" clients.

Projected Federated Averaging. Alg. 2 shows the overall PFA algorithm with client-server interactions and Alg. 3 shows the server-side projection-based averaging in detail (the procedure in line 7 of Alg. 2). PFA involves the following four critical steps on the server side:

(1) *Client division*: After the client sampling procedure at round $t_c \in [T]$, the server divides the participating clients $\mathcal{S}_{t_c}$ into two groups – the *private* clients $\mathcal{S}_{t_c}^{(pri)}$ with a small DP parameter ϵ_m (i.e., a relatively greater magnitude of noise) and the "public" clients $\mathcal{S}_{t_c}^{(pub)}$ with a relatively larger ϵ_m (i.e., a less magnitude of noise). This can be achieved by clustering algorithms such as k-means or Gaussian mixture models. Moreover, we refer to their respective model updates as "private" updates and "public" updates for simplicity (Line 4 in Alg. 2).

(2) *Subspace identification*: Once the server receives the local updates from participating clients, it uses a singular value decomposition (SVD)-based parallel projection method like [46] to identify the most useful subspace for aggregating the updates. The primary idea is that we obtain the approximated gradient subspace from the less noisy information of "public" clients compared to the "private" clients. More specifically, given the "public" updates $\Delta \mathrm{x}_{t_c}^m \in \mathbb{R}^p$ where $C_m \in \mathcal{S}_{t_c}^{(pub)}$ and the project dimension $k < p$, server computes the top-k eigenvectors $\mathrm{V}_k \in \mathbb{R}^{p \times k}$ of the second moment matrix of the average "public" updates $\Delta \mathrm{x}_{t_c}^m$ (Line 2-3 in Alg. 3). Alternatively, we can also find projection matrices through random projection methods such as Johnson-Lindenstrauss transform and Gaussian transform [5]. However, our empirical results in Sec. 5 verify that this approach could lead to a biased estimator of the model updates and a dramatic decrease of model utility compared to the proposed approach.

(3) *"Private" updates projection*: Depending on the calculated projection matrices, the server projects and reconstructs the "private" updates $\Delta \mathrm{x}_{t_c}^{(pri)}$ via the projection $\mathrm{V}_k \mathrm{V}_k^\top$. In practice, instead of applying the parallel projection to each of "private" updates and then computing the average later, we apply it directly to the weighted mean of the "private" updates from all "private" clients for computation efficiency, i.e., $\mathrm{V}_k \mathrm{V}_k^\top \Delta \bar{\mathrm{x}}_{t_c}^{(pri)}$, which is equivalent to the canonical form since the projection onto a subspace is a linear transformation (Line 4-5 in Alg. 3).

(4) *Projected federated averaging*: Finally, the server aggregates the original "public" updates and the projected "private" updates via weighted average to form the global model. Following the transformations in the previous steps, here we actually compute the average by using their weighted mean value $\Delta \bar{\mathrm{x}}_{t_c}^{(pub)}$ and $\Delta \hat{\mathrm{x}}_{t_c}^{(pri)}$ (Line 6 in Alg. 3).

Remark 3. A good rule of thumb of choosing the projection dimension k is to satisfy the common heuristic that the top-k singular values are at least c times as big as the sum of the other singular values, where c is a domain-dependent constant. Strikingly, our experiments show that it suffices to set $k = 1$ (i.e., preserve only the largest singular value space) to gain large utility boost, which in addition allows both efficient SVD computation by iterative algorithms like Lanczos method or Power method and aggressive uplink communication compression to be introduced by PFA+.

Privacy Analysis. We now state the heterogeneous DP guarantees of our PFA approach in federated learning.

Theorem 1 (Differential Privacy Guarantee of local DPSGD). *For each client $C_m \in C$, given the sampling probability $q = B/N_m$ and the total number of local SGD steps $T^* = \frac{KT}{M}$, for any $\epsilon < c_1 q^2 T^*$, the local update satisfies (ϵ_m, δ_m)-differential privacy for any $\delta_m > 0$ if the noise scale*

$$\sigma_m \geq c_2 \frac{q\sqrt{T^* \log(1/\delta)}}{\epsilon} \tag{5}$$

where both c_1 and c_2 are constants, M and K are the number of total clients and sampled clients at each round respectively, N_m is the size of local dataset of client $C_m \in C$, and B is the size of a random batch in local SGD iterations.

Proof. The proof follows the DP proof of DPSGD [1]. □

Theorem 2 (Heterogeneous Differential Privacy Guarantee of PFA). *The PFA algorithm in Alg. 2 satisfies $\{(\epsilon_m, \delta_m)\}_{m \in [M]}$-FL-HDP and $(\max_m \epsilon_m, \max_m \delta_m)$-DP.*

Proof. In PFA, where the sequence of clients' local datasets $\{\mathcal{D}_m\}_{m \in [M]}$ are disjoint subsets drawn from domain $\mathcal{D}$ and each client $C_m \in C$ runs a maximum number of $T^* = \frac{KT}{M}$ steps of DPSGD algorithm $\mathcal{M}_m : \mathcal{D}_m \to \mathbb{R}^p$ that satisfies (ϵ_m, δ_m)-DP locally and independently, Alg. 2 satisfies $\{(\epsilon_m, \delta_m)\}_{m \in [M]}$-heterogeneous DP in federated learning setting.

Moreover, according to the parallel composability of DP as shown in Property 2 and the fact that the PFA algorithm consists of a series of linear operators on the sequence of $\{\mathcal{M}_m\}_{m \in [M]}$, PFA also satisfies $(max_m \epsilon_m, \max_m \delta_m)$-DP. □

Remark 4. We emphasize that in this paper, we do not focus on the task of DP analysis. Thus we follow the widely accepted DPSGD algorithm and moments accountant which enable a direct comparison with the existing works utilizing the same techniques[38]. Using more advanced DP analysis for a

Algorithm 4: Communication-efficient Projected Federated Averaging with Heterogeneous Differential Privacy (PFA+)

input : Clients' privacy preferences $\{(\epsilon_m, \delta_m)\}_{m\in[M]}$, total rounds T
output: global model x_T

1 $\mathrm{x}_0 \leftarrow$ (Initialize randomly)
2 **for** $t' = 0$ *to* $\lfloor \frac{T-1}{\tau} \rfloor$ **do**
3 $t_c \leftarrow t'\tau$
4 $\mathcal{S}_{t_c} \leftarrow$ (random subset of K clients)
5 $\mathcal{S}_{t_c}^{(pub)}, \mathcal{S}_{t_c}^{(pri)} \leftarrow$(Partition $\mathcal{S}_{t_c}$ into two groups according to clients' privacy requirements)
6 **foreach** *client* $C_m \in \mathcal{S}_{t_c}$ **do** *in parallel*
7 $\Delta\tilde{\mathrm{x}}_t^m \leftarrow$ **LocalDPSGD**(x_{t_c})
8 **if** *warmup round* **or** *"public" client* **then**
9 // Full dimensional noisy model updates
  return $\Delta\tilde{\mathrm{x}}_t^m$
10 **else**
11 // Projected noisy model updates
  $\Delta\hat{\mathrm{x}}_{t_c}^m \leftarrow \mathrm{V}_k[t'-1]^\top \Delta\tilde{\mathrm{x}}_{t_c}^m$
12 **return** $\Delta\hat{\mathrm{x}}_{t_c}^m$
13 $\Delta\bar{\mathrm{x}}_{t_c}, \mathrm{V}_k[t'] \leftarrow$ **ProjectedFedAvgPro**$(\{\Delta\mathrm{x}_{t_c}^m\}_{m\in[K]}, \mathcal{S}_{t_c}^{(pub)}, \mathcal{S}_{t_c}^{(pri)}, \mathrm{V}_k[t'-1])$
14 $\mathrm{x}_{t_c+\tau} \leftarrow \mathrm{x}_{t_c} - \Delta\bar{\mathrm{x}}_{t_c}$
15 **return** x_T

tighter privacy amplification result[39] as well as tighter privacy composition[7] is also recommended.

Convergence Analysis. The convergence analysis of PFA is presented in Theorem 3. The proof is in Appendix A.

Theorem 3. *Under Assumptions 1-3, for non-convex L-Lipschitz smooth objective function, the convergence result of PFA is*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\Big[\|\widetilde{\mathrm{V}}_{t_c}\widetilde{\mathrm{V}}_{t_c}^\top \nabla\mathcal{L}(\bar{\mathrm{x}}_t)\|_2^2\Big] \leq \frac{2}{\eta T}(\mathcal{L}(\bar{\mathrm{x}}_0) - \mathcal{L}^*) + \frac{\eta^2 L^2(\tau+1)(K+2)}{K}(\frac{\kappa^2}{B} + \bar{\sigma}) + \frac{L\eta\kappa^2}{KB}, \tag{6}$$

which is under the choice of η satisfies

$$-\frac{\eta}{2} + \frac{\lambda\eta^3 L^2[2C_1 + \tau(\tau+1)](K+2)}{2K} + \frac{\lambda\eta^2 L[C_1+K]}{K^2} \leq 0. \tag{7}$$

Remark 5. Theorem 3 shows that the convergence of PFA in terms of the projected gradient norm has the same convergence rate as the non-private FedAvg under the same set of assumptions. Thus, our algorithm is able to deliver the desired heterogeneous differential privacy without affecting the overall convergence speed.

4.5 PFA+: A Communication-efficient Variant

As we have mentioned before, communication overhead is a primary bottleneck for federated learning. PFA has not made any progress in this challenge since the size of model updates communicated between clients and server has not changed. In this section, we introduce an improved variant, named PFA+which reduces the size of the communicated local model updates in order to greatly reduce the uplink communication cost. Considering the asymmetry of the uplink and downlink bandwidth on the clients' side, it is especially important to reduce the uplink bandwidth for a more significant practical impact in terms of total runtime. In section 5, our empirical studies suggest that compared with other baselines, PFA+ has offered a significant 99% reduction for "private" clients in terms of the megabytes that communicated from clients to the server, while the test accuracy remains the same level as PFA.

Algorithm 5: Asynchronous Projection Federated Averaging

input : noisy model updates $\{\Delta\mathrm{x}^m\}_{C_m\in\mathcal{S}}$, "public" clients $\mathcal{S}^{(pub)}$, "private" clients $\mathcal{S}^{(pri)}$
output: the aggregated model updates $\Delta\bar{\mathrm{x}}$

Function ProjectedFedAvgPro($\{\Delta\mathrm{x}^m\}_{C_m\in\mathcal{S}}, \mathcal{S}^{(pub)}, \mathcal{S}^{(pri)}, \mathrm{V}_k^{(pre)}$):
  // Reconstruct the projected privacy updates using the previous projection matrix
  $\Delta\bar{\mathrm{x}}^{(pri)} \leftarrow \sum_{m\in\mathcal{S}^{(pri)}} \mathcal{W}_{\mathcal{S}^{(pri)}}(\epsilon_m)\cdot\Delta\mathrm{x}^m$
  $\Delta\hat{\mathrm{x}}^{(pri)} \leftarrow \mathrm{V}_k^{(pre)}\Delta\bar{\mathrm{x}}^{(pri)}$
  $\mathcal{S} \leftarrow \mathcal{S}^{(pub)} + \mathcal{S}^{(pri)}$
  $\Delta\bar{\mathrm{x}} \leftarrow \frac{\sum_{m\in\mathcal{S}^{(pub)}}\epsilon_m}{\sum_{m\in\mathcal{S}}\epsilon_m}\cdot\Delta\bar{\mathrm{x}}^{(pub)} + \frac{\sum_{m\in\mathcal{S}^{(pri)}}\epsilon_m}{\sum_{m\in\mathcal{S}}\epsilon_m}\cdot\Delta\hat{\mathrm{x}}^{(pri)}$
  // Compute the current projection matrix
  $\Delta\bar{\mathrm{x}}^{(pub)} \leftarrow \sum_{m\in\mathcal{S}^{(pub)}} \mathcal{W}_{\mathcal{S}^{(pub)}}(\epsilon_m)\cdot\Delta\mathrm{x}^m$
  $\Sigma \leftarrow \Delta\bar{\mathrm{x}}^{(pub)}\cdot(\Delta\bar{\mathrm{x}}^{(pub)})^\top$
  $\mathrm{V}_k \leftarrow$ (Compute the top-k eigenvectors of Σ)
  return $\Delta\bar{\mathrm{x}}, \mathrm{V}_k$

The main idea of PFA+ is to communicate the dimension-reduced model updates, i.e. projected updates on the subspace, from clients to the server. More specifically, upon the PFA's architecture in Alg. 2, PFA+ makes the following algorithmic changes on the clients' side and server side respectively:

(1) *Local update with projection.* On the clients' side, instead of uploading the model updates $\Delta\mathrm{x}_{t_c}^m$, each "private" client $C_m \in \mathcal{S}_{t_c}^{(pri)}$ first projects the computed model updates onto the top-k subspace (obtained from the server based on the "public" clients' update from previous round), then upload the projected updates $\mathrm{V}_k \in \mathbb{R}^{p\times k}$ to the server. As for the "public" clients, they still upload the full-dimensional model updates. (Line 7-11 in Alg. 4).

(2) *Projected federated averaging with delayed subspace.* On the server side, similarly with PFA, all full-dimensional "public" updates are used to calculate top-k subspace as shown in Alg. 5. Instead of projecting the "private" updates onto the calculated subspace, the server reconstructs the projected "private" updates back to the original shapes via the same projection matrices from previous round, then merges them with the public updates via average.

We note the slight discrepancy or asynchronization: the subspace computed from the public updates in the previous round is used to project the "private" updates in the current round, and the current subspace is used for the next round. In PFA+, we apply the projected federated averaging in a delayed manner , considering that the top singular space usually evolves slowly between two consecutive rounds. For the first round, which we call it the *warmup* round, we follow the same procedure as in PFA, i.e., all clients upload the original model updates and the server aggregates them to compute the first top-k subspace. This first subspace will be applied in the second round. We verified empirically that this small asynchronization of the projection subspace has minimal impact on the final model utility while allowing significant communication reduction. Alg. 5 shows the complete details.

Privacy Analysis. Theorem 4 gives the heterogeneous DP guarantee of PFA+. Since the DP randomizers of PFA+ remains essentially

unchanged compared with PFA, thus the proof is similar with Theorem 2, and we omit it here for brevity.

Theorem 4 (Heterogeneous Privacy Guarantee of PFA+). *The PFA+ in Alg. 4 satisfies* $\{(\epsilon_m, \delta_m)\}_{m\in[M]}$*-FL-HDP and* $(max_m\epsilon_m, \max_m \delta_m)$*-DP.*

Computation Cost Analysis. Theorem 5 gives the computation cost per round of training for calculating the subspaces and projections on the server and clients side. Note that the analysis for PFA is the same except that both subspaces and projections are computed on the server side. The proof is in Appendix D.2.

Theorem 5. *Let p denote the dimension of the parameter space and m be the number of "public" clients participating in a training round, the total complexity of computing the top-k subspaces for central server is $O(kmp)$, and the complexity of computing the projections for each client is $O(k)$.*

Communication Cost Analysis. We consider the uplink bandwidth as the total number of uploaded bits in the course of the training process. Without loss of generality, we assume the machine learning model with total number of parameters p is pF bits, where F is typically 32 bits. For PFA+, the original parameters in a given layer of the model will be compressed to a k-dimensional ($k \ll p$) vector. For any model, e.g., a 2-layer logistic regression model, each "private" client only requires uplink bandwidth with $2kF$ bits in each round. If $k = 1$, this value will be aggressively reduced to merely $2F$ bits. For complex deep learning models with hundreds and thousands of parameters, PFA+ would offer a considerable communication reduction.

5 EVALUATION

In this section, we perform a comprehensive evaluation with a series of experiments to demonstrate that WeiAvg, *PFA* and *PFA+* can attain a higher test accuracy compared to the common federated averaging algorithms which do not take the heterogeneous privacy preferences into account, and are significantly better than Minimum baseline which achieves the most strict level of privacy.

5.1 Experiment Setup

Datasets and Models. We evaluate our methods on two open benchmark datasets: MNIST[24], Fashion-MNIST (fMNIST) [42], CIFAR10 [23] and IMDb movie review [31]. More details of datasets and the models we used are presented in Table 6 in Appendix B.

Baselines and Methods. We compare the three algorithms we proposed with four baselines as follows.

- **WeiAvg**: weighted average algorithm in Algorithm 1.
- **PFA**: PFA method in Algorithm 2.
- **PFA+**: communication-efficient variant of PFA in Algorithm 4.
- **FedAvg**: FedAvg [32] algorithm with heterogeneous DP.
- **Minimum**: FedAvg with uniform $(\min_m \epsilon_m, \min_m \delta_m)$-DP.
- **Maximum**: FedAvg with uniform $(\max_m \epsilon_m, \max_m \delta_m)$-DP [1].
- **NP-FedAvg**: the non-private FedAvg algorithm.

Privacy preferences. For explicitly simulating the potential distribution of clients' privacy preferences and compare the methods with those different settings, we consider 4 different types of random distributions as shown in Tab. 1, i.e., Uniform, Pareto, Gaussian and multimodal distribution (a mixture of two or three different Gaussian distributions). Fig. 9 in Appendix B gives examples of clients' privacy preferences for M being 20, 30, 40 and 50, respectively. For example, considering a scenario where the privacy preferences of 20 clients are drawn from a MixGauss1 distribution, there would exist about 2 "public" clients with privacy budgets around 10.0, and the others with privacy budgets around 0.1 would be treated as "private" clients.

Table 1: Distribution of privacy preferences

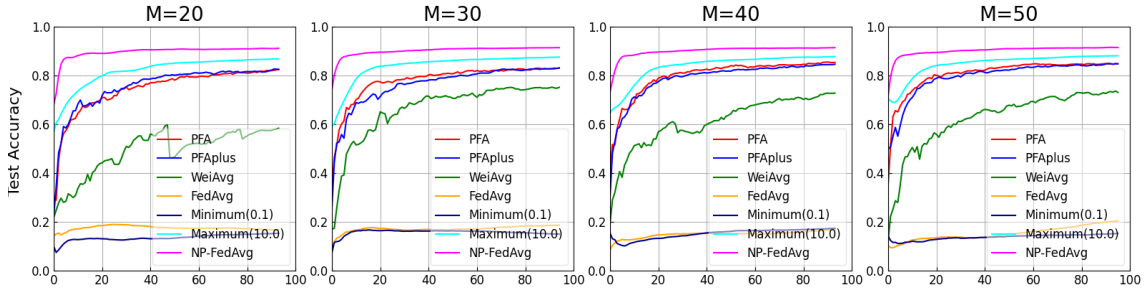
Distribution	Parameters Setting
Uniform	Uniform distribution $U(1.0, 10.0)$
Gauss	Gaussian distribution $\mathcal{N}(3.0, 1.0)$
Pareto	Pareto distribution
MixGauss1	Mixture of $\mathcal{N}_1(0.1, 0.01)$ and $\mathcal{N}_2(10.0, 0.1)$ with mixture weights 0.9 and 0.1
MixGauss2	Mixture of $\mathcal{N}_1(0.5, 0.01)$ and $\mathcal{N}_2(10.0, 0.1)$ with mixture weights 0.9 and 0.1
MixGauss3	Mixture of $\mathcal{N}_1(1.0, 0.1)$ and $\mathcal{N}_2(10.0, 0.1)$ with mixture weights 0.9 and 0.1
MixGauss4	Mixture of $\mathcal{N}_1(0.1, 0.01)$, $\mathcal{N}_2(1.0, 0.1)$ and $\mathcal{N}_3(10.0, 1.0)$ with mixture weights 0.5, 0.4 and 0.1

Implementations. We run experiments on a machine with an Intel Core i7-8700K and one NVIDIA GeForce GTX 1080 Ti running Ubuntu with 64GB memory. To simulate the cross-silo FL setting in the cases of IID and non-IID data distribution across the clients respectively, we follow the partitioning strategies in [32]. For example, the training examples of MNIST/fMNIST are partitioned into $M = 20, 30, 40$ and 50 subsets, each client holds the same size of 1200 training examples. For IID case, each client's local dataset consists of examples from all classes; for non-IID case, each client has an imbalanced dataset with non-uniform distribution of class labels. Table 3 and 4 in Appendix B show the examples of IID and non-IID partition of MNIST for $M = 20$ respectively.

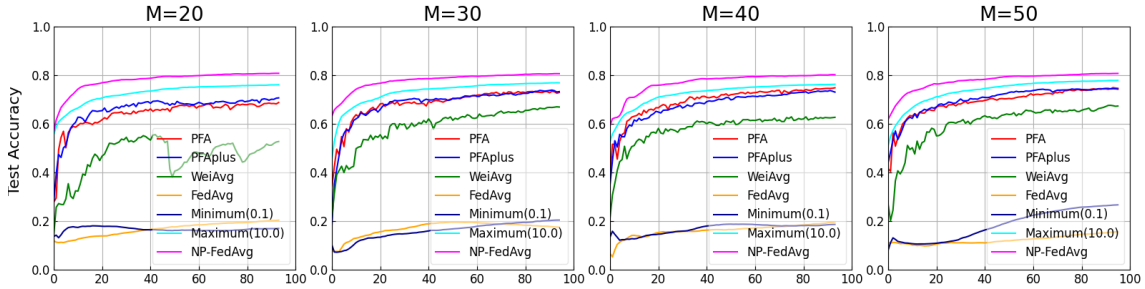
We sequentially run the local training procedures, thus we do not explore hyper-parameters as thoroughly since these experiments require significant computational resources. All of our experiments explore a variety of constant learning rates (e.g., [0.1, 0.05, 0.01, 0.025, 0.005, 0.001]) and a variety of projection dimensions k (e.g., [1, 2, 5, 10, 64, 128, 256]) for the best results. Unless otherwise stated, we fix the random client sampling ratio $\frac{K}{M} = 0.8$ per round and up to $T = 100$ communication rounds in the course of the learning process. For local update procedures, we usually set the number of local iterations $\tau = 100$ and the random batch size $B = 8$. Since the model training is a randomized procedure, we repeat all the experiments 5 times and report the mean accuracy.

For the evaluation metrics, we use test accuracy on the test examples to measure the model utility, and the number of transferred megabytes between clients and server along the training process to compare the uplink bandwidth cost.

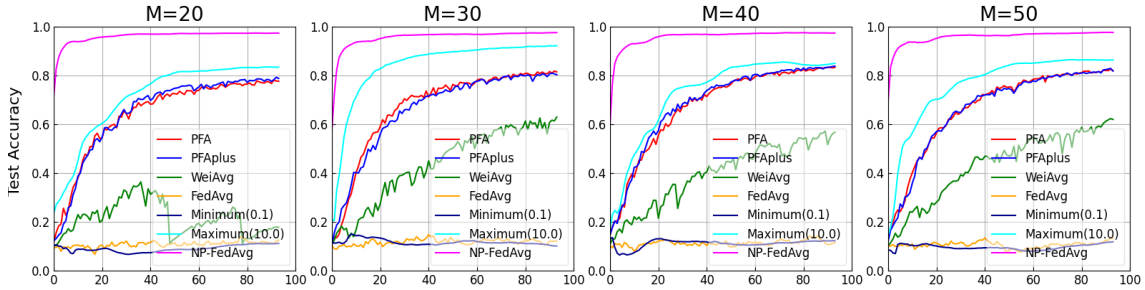
[1] We introduce this additional baseline as an "upper bound" of the performance considering the extreme case where all clients uses the identical maximum privacy budget.



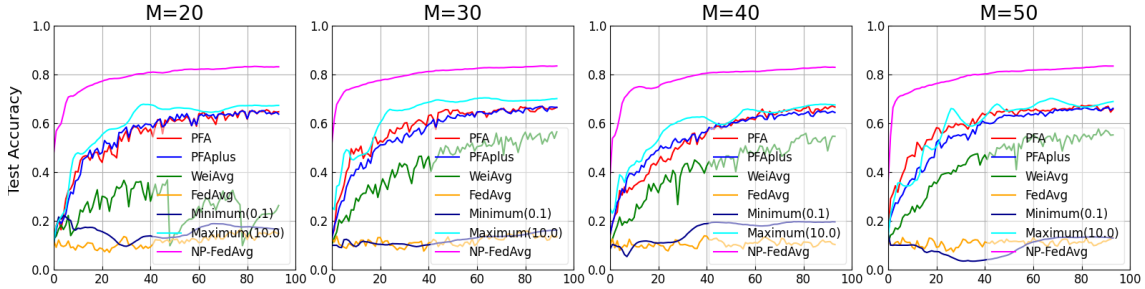
(a) MNIST-LogR-IID-MixGauss1



(b) fMNIST-LogR-IID-MixGauss1



(c) MNIST-CNN-IID-MixGauss1



(d) fMNIST-CNN-IID-MixGauss1

Figure 3: Test accuracy versus communication rounds evaluated on MNIST and Fashion-MNIST (fMNIST) in IID data setting with privacy preferences distribution of MixGauss1.

5.2 Evaluation Results on IID Data

5.2.1 The Effect of Privacy Specification. In this section, we focus on two candidate privacy preference settings (i.e., MixGauss1 and MixGauss3), both of which have the same level of maximal privacy preference 10.0, while the minimal privacy preferences are around 0.1 and 1.0, respectively. In Fig. 3, we show the experimental results evaluated on MNIST and fMNIST in IID setting with the distribution of MixGauss1, while relegating the results of other settings to

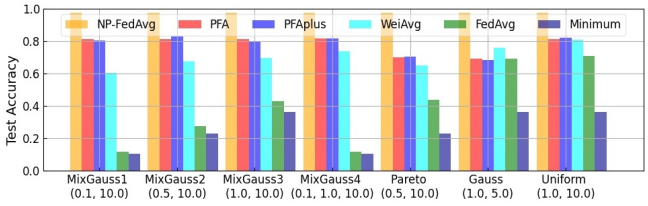


Figure 4: Effect of privacy specification: test accuracy on the MNIST-CNN experiments in IID data setting.

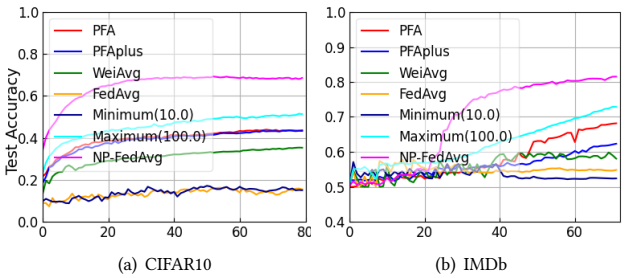
Appendix B due to space limitation. In Fig. 4, we further present the results with other distributions of privacy budget as listed in Table 1, which compare all algorithms under a wider selection of privacy specifications that manifest various scattering degree of the privacy budgets between "public" and "private" clients.

From Fig. 3, compared with WeiAvg, both PFA and PFA+ attain more distinct utility advantages over the FedAvg with HDP algorithm and the Minimal mechanism for both logistic regression and CNN model. Although all private algorithms have worse accuracy than non-private baselines, a reasonable level of model utility remains for projection-based methods, while FedAvg with DP becomes ineffective and unusable, e.g., the test accuracy of FedAvg fluctuates around 0.1 in Fig. 3(c). As for the comparison between PFA+ and PFA, we can observe from all plots that PFA+ attains a model performance very close to PFA while achieving significant communication reduction. This evaluation results will be further discussed in Section 5.2.2. Fig. 10 in Appendix B gives the analogous experiments with distribution of MixGauss3. The main difference is the utility gap between FedAvg/Minimum and our proposed methods significantly shrinks. It makes sense since greater privacy budgets lead to weaker perturbation, resulting in performance improvement of baseline methods.

As a comparison, Fig. 4 further plots the impact of other distributions listed in Table 1. We only present the experiment results on MNIST-CNN-IID with $M = 30$ in the paper, while deferring other cases that show a similar trend due to space limitation. By default, all test accuracy values are averaged over the last 10 rounds before the round limit is reached. We can observe a more clear trend that, with increasing the scattering degree of the privacy budgets between "public" and "private" clients, PFA and PFA+ always outperform the FedAvg with HDP and Minimum mechanism, and maintain reasonable test accuracy. For WeiAvg and FedAvg with HDP methods, both of them are vulnerable to the scattering degree and the minimal value of clients' privacy preferences. When the privacy budgets follow the Uniform or the Gaussian distribution, they could reach a model utility that is pretty similar to our projection-based methods.

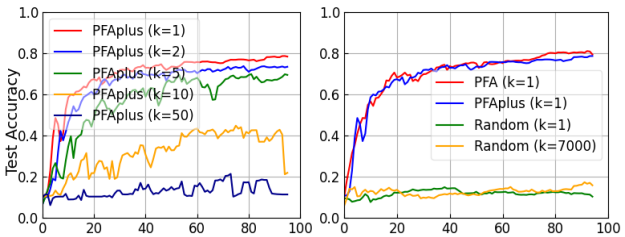
As for the evaluations on more complex datasets CIFAR10 and IMDB from Fig. 5, we can also observe PFA and PFA+ outperforms weighted averaging slightly, and all three of them get significantly better performance compared with the basic FedAvg.

5.2.2 The Effect of Projection Methods. In this subsection, in order to verify that the low-rank subspace extracted from "public" updates indeed contains indicative information, we compare it with



(a) CIFAR10 (b) IMDb

Figure 5: Test accuracy versus communication rounds evaluated on CIFAR10 and IMDb in IID data setting for $M = 50$.



(a) Effect of projection dimensions (b) Effect of projection methods

Figure 6: Effect of projection dimensions and projection methods: test accuracy versus communication rounds on the MNIST-LogR experiments in IID data setting when $M = 30$.

random low-rank projection methods (e.g., via the Gaussian transform) in terms of model accuracy with a variety of projection dimensions and fixed local update iterations. Then, we show the unique superiority of PFA+ in reducing the communication cost compared with the other baselines.

Random projection versus "public" updates-extracted projection. Fig. 6 shows the impact of projection dimensions for the random projection and the "public" updates-extracted projection respectively on the MNIST-LogR-MixGauss1 experiments in IID data setting. In Fig. 6(a), we can see PFA+ reaches the optimal test accuracy (around 80%) when the projection dimension $k = 1$, and the performance is quite close to the case when $k = 2$. However, as the dimension increases, it leads to a greater variance and poorer test accuracy. This is a indicator of the fact that most stochastic gradients in general stay in a low-dimensional subspace.

In contrast, the random projection method has the worst performance when k is far less than the original dimension p. Moreover, when the "private" clients have strict privacy preferences, either a large dimension or a small dimension cannot reach an acceptable model utility, as shown in Fig. 6(b). These strong experimental results underscore the effectiveness of the parallel projection method applied in our proposed frameworks.

Communication-reduction effects of PFA+. Fig. 7 plots the communication-reduction effects of PFA+ in terms of the accumulated number of megabytes that communicated from clients to the server during the FL period when there are 5 "public" clients and 45 "private" clients participating in the training process.

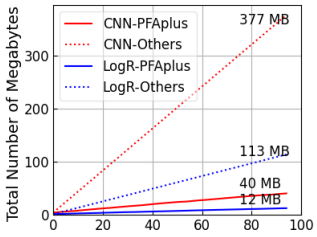


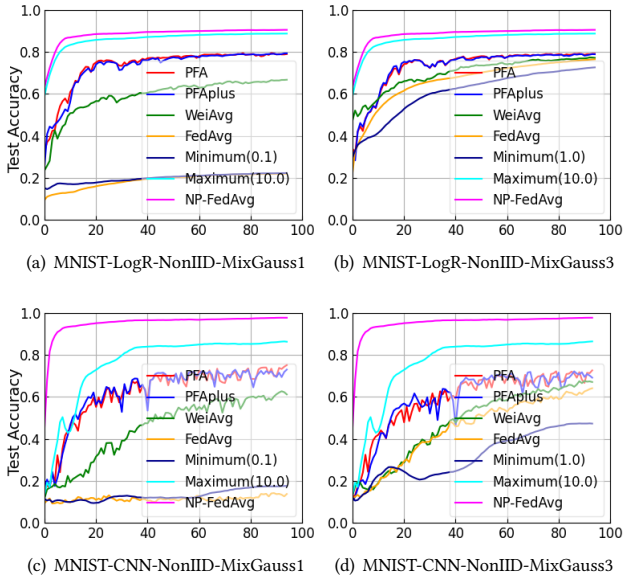
Figure 7: Effect of communication reduction: number of megabytes on MNIST in IID data setting when $M = 50$.

For a logistic regression model with 2 layers (a fully connected layer and a softmax layer) and projection dimension $k = 1$, PFA+ only requires $2F$ bits ($F = 32$) for each "private" client at each communication round, and requires a total number of 12MB uplink bandwidth after 100 communication rounds, while the other methods need around 113MB since all clients need to transfer the full-dimensional model parameters. For a CNN model with a total of 8 layers (see subsection 5.1 for detailed model description) and the projection dimension $k = 1$, PFA+ requires a total number of 40MB uplink bandwidth while the others need 377MB. Overall, for both logistic regression and CNN models, PFA+ could offer a significant uplink bandwidth reduction by nearly 99% during the entire training process w.r.t the "private" clients and 90% w.r.t all clients with a negligible decrease in model utility, which indicates that the communication-reduction effects of PFA+ depend on the number of "private" clients participating in the training process. Note that to design a more reasonable "private" client identification mechanism and make a better trade-off between communication reduction and model utility is also part of our future work.

Computation overhead of PFA+. Tab. 5 in Appendix B summarizes the empirical results of computation cost of PFA+ for both the server and the clients in terms of the averaged runtime (s) per round over the entire training process for computing the subspaces and projecting the "private" updates, respectively. We can see the overhead for clients are relatively low, and the complexity of the model architecture will significantly affect the overhead for computing the subspaces.

5.3 Evaluation Results on Non-IID Data

In Fig. 11 and Fig. 12 in Appendix B, we show full experiment results for both MNIST and fMNIST in Non-IID setting with privacy preferences distribution MixGauss1 and MixGauss3. We simulated a relatively weak but reasonable non-IID setting, where all training examples in the benchmark dataset are first sorted by labels, divided into $10M$ shards of size $1200/10 = 120$, and assigned to each of M clients 10 shards like [32]. In this way, each client could be assigned to a local dataset with a varying number of labels, and each label has a varying number of examples. Only a very few exceptions could possess training examples with all labels. Compared with the learning results of the three methods we proposed in the case of IID setting, an obvious decrease in test accuracy can be observed in all experiments. The gaps between the FedAvg and the projection-based methods have also narrowed. Nonetheless, PFA and PFA+ still outperform other methods in most cases.



(a) MNIST-LogR-NonIID-MixGauss1 (b) MNIST-LogR-NonIID-MixGauss3

(c) MNIST-CNN-NonIID-MixGauss1 (d) MNIST-CNN-NonIID-MixGauss3

Figure 8: Test accuracy versus communication rounds evaluated on MNIST in NonIID data setting when $M = 30$.

6 CONCLUSION

This paper focuses on the heterogeneous differentially private federated learning problem and studies how to optimize utility for the joint model under the heterogeneous privacy restriction across different institutions. Towards overcoming two inherent challenging problems of federated learning: data privacy and communication cost, we first propose the *PFA* approach which leads to a larger noise reduction and higher utility compared with the standard federated average algorithm, then further propose a communication-efficient variant *PFA+* which offers a significant 99% communication reduction for "private" clients. Through a comprehensive evaluation on both statistical learning and deep learning, we demonstrate the effectiveness of both algorithms compared to the standard federated average algorithm.

There are several different directions towards generalizing the work here. First, we plan to theoretically demonstrate the effectiveness and robustness of our proposed methods. We also plan to integrate this work into the personalized federated learning tasks especially for non-IID data distributions, in which each institution not only could enjoy a customized privacy protection but also gain a personalized machine learning model.

ACKNOWLEDGMENTS

This work was supported in part by the NSFC grants (61941121, 62172423, 91846204, and 62102352), NSF grants (CNS-2124104, CNS-1952192, and IIS-1838042), AFOSR under DDDAS program (FA9550-12-1-0240), the NIH grants (R01GM118609), CTSA UL1TR002378, Cisco Research University Award (2738379), the National Key R & D Program of China (2021YFB3101100), the Fundamental Research Funds for the Central Universities, and an Emory University URC award.

REFERENCES

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[2] M. Al-Shedivat, J. Gillenwater, E. Xing, and A. Rostamizadeh. Federated learning via posterior averaging: A new perspective and practical algorithms. *arXiv preprint arXiv:2010.05273*, 2020.

[3] M. Alaggan, S. Gambs, and A. Kermarrec. Heterogeneous differential privacy. *J. Priv. Confidentiality*, 7(2), 2016.

[4] A. Bellet, R. Guerraoui, M. Taziki, and M. Tommasi. Personalized and private peer-to-peer machine learning. In *International Conference on Artificial Intelligence and Statistics*, pages 473–481. PMLR, 2018.

[5] J. Blocki, A. Blum, A. Datta, and O. Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 410–419. IEEE Computer Society, 2012.

[6] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecnỳ, S. Mazzocchi, H. B. McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

[7] Z. Bu, J. Dong, Q. Long, and W. J. Su. Deep learning with gaussian differential privacy. *arXiv preprint arXiv:1911.11607*, 2019.

[8] A. K. Chathoth, A. Jagannatha, and S. Lee. Federated intrusion detection for iot with heterogeneous cohort privacy. *arXiv preprint arXiv:2101.09878*, 2021.

[9] R. Chen, H. Li, A. K. Qin, S. P. Kasiviswanathan, and H. Jin. Private spatial data aggregation in the local setting. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 289–300. IEEE, 2016.

[10] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 371–380, 2009.

[11] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[12] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[13] C. Dwork, G. N. Rothblum, and S. Vadhan. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60. IEEE, 2010.

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

[15] G. Gur-Ari, D. A. Roberts, and E. Dyer. Gradient descent happens in a tiny subspace. *arXiv preprint arXiv:1812.04754*, 2018.

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

[17] B. Hitaj, G. Ateniese, and F. Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 603–618, 2017.

[18] Z. Jorgensen, T. Yu, and G. Cormode. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*, pages 1023–1034. IEEE, 2015.

[19] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[20] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

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

[22] I. Kotsogiannis, S. Doudalis, S. Haney, A. Machanavajjhala, and S. Mehrotra. One-sided differential privacy. In *36th IEEE International Conference on Data Engineering, ICDE 2020, Dallas, TX, USA, April 20-24, 2020*, pages 493–504. IEEE, 2020.

[23] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[25] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.

[26] H. Li, L. Xiong, Z. Ji, and X. Jiang. Partitioning-based mechanisms under personalized differential privacy. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 615–627. Springer, 2017.

[27] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

[28] X. Li, Q. Gu, Y. Zhou, T. Chen, and A. Banerjee. Hessian based analysis of sgd for deep nets: Dynamics and generalization. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 190–198. SIAM, 2020.

[29] Y. Li, S. Liu, J. Wang, and M. Liu. A local-clustering-based personalized differential privacy framework for user-based collaborative filtering. In *International Conference on Database Systems for Advanced Applications*, pages 543–558. Springer, 2017.

[30] T. Lin, L. Kong, S. U. Stich, and M. Jaggi. Ensemble distillation for robust model fusion in federated learning. *arXiv preprint arXiv:2006.07242*, 2020.

[31] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[32] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

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

[34] B. Niu, Y. Chen, B. Wang, J. Cao, and F. Li. Utility-aware exponential mechanism for personalized differential privacy. In *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2020.

[35] V. Papyan. Measurements of three-level hierarchical structure in the outliers in the spectrum of deepnet hessians. *arXiv preprint arXiv:1901.08244*, 2019.

[36] M. Rigaki and S. Garcia. A survey of privacy attacks in machine learning. *arXiv preprint arXiv:2007.07646*, 2020.

[37] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu. Data poisoning attacks against federated learning systems. In *European Symposium on Research in Computer Security*, pages 480–501. Springer, 2020.

[38] D. Wang, C. Chen, and J. Xu. Differentially private empirical risk minimization with non-convex loss functions. In *ICML*, 2019.

[39] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *AISTATS*. PMLR, 2019.

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

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

[42] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[43] M. Yang, T. Zhu, Y. Xiang, and W. Zhou. Personalized privacy preserving collaborative filtering. In *International Conference on Green, Pervasive, and Cloud Computing*, pages 371–385. Springer, 2017.

[44] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex. Differentially private model publishing for deep learning. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 332–349. IEEE, 2019.

[45] S. Zhang, L. Liu, Z. Chen, and H. Zhong. Probabilistic matrix factorization with personalized differential privacy. *Knowledge-Based Systems*, 183:104864, 2019.

[46] Y. Zhou, S. Wu, and A. Banerjee. Bypassing the ambient dimension: Private SGD with gradient subspace identification. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[47] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, pages 14774–14784, 2019.