Federated Learning with Heterogeneous Differential Privacy

Anonymous authors
Paper under double-blind review

Abstract

Federated learning (FL) takes a first step towards preserving privacy by training statistical models while keeping client data local. Models trained using FL may still indirectly leak private client information through model updates during training. Differential privacy (DP) can be employed on model updates to provide privacy guarantees within FL, typically at the cost of degraded accuracy of the final trained model. Both non-private FL and DP-FL can be solved using variants of the federated averaging (FedAvg) algorithm. In this work, we consider a heterogeneous DP setup where clients may require varying degrees of privacy guarantees. First, we analyze the optimal solution to a simplified linear problem with (heterogeneous) DP in a Bayesian setup. We find that unlike the non-private setup, where the optimal solution for homogeneous data amounts to a single global solution for all clients learned through FedAvg, the optimal solution for each client in this setup would be a different one even when data is homogeneous. We also analyze the privacy-utility tradeoff for this problem, where we characterize the gains obtained from the heterogeneous privacy where some clients opt for less stringent privacy guarantees. We propose a new algorithm for federated learning with heterogeneous DP, referred to as FedHDP, which employs personalization and weighted averaging at the server using privacy choices by clients, to achieve the Bayes optimal solution on a class of liner problems for all clients. Through numerical experiments we show that FedHDP provides up to 9.27% performance gain compared to the baseline DP-FL for the considered datasets where 5% of clients opt out of DP. Additionally, we show a gap in the average performance of local models between non-private and private clients of up to 3.49%, empirically illustrating that the baseline DP-FL might incur a large utility cost when not all clients require the stricter privacy guarantees.

1 Introduction

The abundance of data and advances in computation infrastructure have enabled the training of high-quality machine learning models. On the other hand, the data is distributed over many devices that are typically power-constrained and have limited computational capabilities. To reduce the amount of data transmission over networks and maintain the privacy of raw data, (McMahan et al., 2017) proposed the federated learning (FL) framework for training a central server-side model using decentralized data at clients. See the recent surveys (Kairouz et al., 2019; Li et al., 2020; Wang et al., 2021) for more.

Federated learning frameworks aim to train a global model iteratively and collaboratively using clients' data. During each round, the server has access to a select number of clients, each of whom has a local dataset. The server broadcasts the current model to such clients, who train the model by taking gradient steps using their local data on the model and return the gradient-based update back to the server. The server then aggregates the updates and produces the new global model for the next round. Several prior works on federated learning algorithms have been proposed in the literature to overcome various issues that arise in realistic federated learning setups, e.g., data heterogeneity (Konečnỳ et al., 2016; Zhao et al., 2018; Corinzia et al., 2019; Hsu et al., 2019; Karimireddy et al., 2020; Reddi et al., 2020), and device dropout and communication cost (Li et al., 2018; Zhu & Jin, 2019; Wang et al., 2020; Al-Shedivat et al., 2020).

Despite the clients' data being kept on device in federated learning, the deployed model at the central server is still vulnerable to various privacy attacks, such as membership inference attacks (Shokri et al., 2017) and model inversion attacks (Fredrikson et al., 2015), among others. In order to mitigate such a critical issue, privacy-preserving variations of federated learning algorithms are proposed in the literature. One promising

approach to privacy-preserving FL utilizes differential privacy in order to provide the privacy guarantees. Differential privacy is a widely studied and accepted mathematical notion that describes privacy-preserving algorithms where the information leakage of private data is bounded. Differential privacy is defined as follows

Definition 1 (differential privacy (DP) (Dwork et al., 2014))**.** *A randomized algorithm $A(\cdot)$, whose image is denoted as $\boldsymbol{O}$, is said to be (ϵ, δ)-DP if for any two inputs $\boldsymbol{D}$ and $\boldsymbol{D}'$ that differ in just one entry, and all subsets $O \subseteq \boldsymbol{O}$ the following relationship holds*

$$\Pr(A(\boldsymbol{D}) \in O) \leq e^{\epsilon} \Pr(A(\boldsymbol{D}') \in O) + \delta. \tag{1}$$

In federated learning, instead of targeting privacy guarantees for individual samples of each client, it is common to consider a different differential privacy guarantee by having the adjacent datasets describe the case where we seek to provide privacy at the client-level data (McMahan et al., 2018), i.e., global DP. Moreover, heterogeneous differential privacy has been a topic of interest in the literature and has been considered in various works such as (Alaggan et al., 2016; Jorgensen et al., 2015) that aim at examining the problem from a theoretical point of view. It is worth noting that using differential privacy in federated learning causes unavoidable degradation in performance. Several prior works utilized differential privacy to provide privacy guarantees for federated learning algorithms. For example, (Truex et al., 2020; Sun et al., 2020; Kim et al., 2021; Song et al., 2015) apply DP mechanisms at clients to ensure local DP guarantees, where clients have complete control over the amount of privacy they desire. On the other hand, (Geyer et al., 2017; McMahan et al., 2018; Andrew et al., 2019; Wei et al., 2020; Bietti et al., 2022) apply DP mechanisms at the server to ensure global DP guarantees for all clients. Applying DP typically causes some degradation in utility, i.e., the model's performance degrades as the privacy budget gets smaller (Alvim et al., 2011; Sankar et al., 2013; Makhdoumi et al., 2014; Calmon et al., 2015). These approaches to privacy-preserving federated learning use fixed privacy budgets for all clients, an approach that can be overly strict and cause unnecessary degradation in performance. A variation of DP setups is proposed in the literature, for scenarios with clients and a server, where a hybrid model is used by combining local DP with global DP and giving clients the option to opt in either (Avent et al., 2017; Beimel et al., 2019). In (Avent et al., 2017), a *blender* process is considered for computing heavy hitters where some clients opt in local DP while the remaining opt in global DP. Some drawbacks of these works include their assumption of clients' data to be IID, as well as applying local DP which requires a large number of samples at clients. These two assumptions make applying such an approach in FL setups difficult due to the non-IID nature of clients' data in FL, and the relatively small number of samples generated by clients in FL which requires either increasing the variance of the added noise or relaxing the privacy leakage budget leading to either large degradation in performance or higher privacy leakage budgets.

Heterogeneity is a fundamental feature of federated learning, e.g., clients' datasets can no longer be considered IID in FL (Li et al., 2020). This introduces a challenging and critical problem that needs to be resolved to realize the full potential of privacy-preserving FL in realistic environments. One possible solution is based on model personalization, where clients learn personalized local models that performs better on their local data compared to the global model when heterogeneity exists. There are different approaches to personalization in the literature by introducing different modifications to FL algorithms, e.g., (Smith et al., 2017; Wang et al., 2019; Arivazhagan et al., 2019; Khodak et al., 2019; Mansour et al., 2020; Fallah et al., 2020; Deng et al., 2020; Dinh et al., 2020; Li et al., 2021). Another type of heterogeneity include systems heterogeneity where different devices have different capabilities, in terms of various characteristics such as connection, computational, and power capabilities (Li et al., 2018). Solutions to system heterogeneity include designing algorithms that can tolerate device dropout, reduce communication cost, or reduce computations cost (Caldas et al., 2018; Gu et al., 2021; Horvath et al., 2021; Li et al., 2018). In this work, we study heterogeneity along the privacy axis. We find that, similar to other notions of heterogeneity, addressing this problem in an optimal fashion requires curating personalized solutions for each client, which is different from the homogeneous non-private setup where one global model can be used to serve all clients.

1.1 Our Contributions

In this work, we develop a novel framework to study heterogeneity in privacy requirements in federated learning setups. More specifically, we consider a new setup for privacy-preserving federated learning where privacy parameters are no longer fixed across all clients. We show that existence of clients who choose to relax

their privacy choices, even if they represent a small percentage of the overall population, can be leveraged to improve the performance of the global model as well as the personalized local models. Specifically,

1. We propose a heterogeneous setup for privacy in federated learning frameworks. The proposed setup considers heterogeneity in privacy choices of clients in FL. Instead of granting the same level of privacy for all clients, each client is given the option to choose their desired level of privacy. In this setup, the server presents each client with a set of privacy levels from which the client chooses their desired option according to their needs and goals. In this case, the client should expect to observe a privacy-utility trade-off similar to various other differentially-private learning setups.
2. We consider the simplified Bayesian setup of federated point estimation, introduced by (Li et al., 2021), and show that unlike the case of *non-private FL with homogeneous data*[1], where the Bayes optimal solution is a single global model that could be learnt via vanilla federated averaging, the optimal Bayes solution in private FL requires personalization. We also characterize the optimal degree of personalization based on the privacy requirements, degree of data heterogeneity, and other parameters (See Theorem 3). Further, we characterize a privacy-utility tradeoff observed at clients.
3. We propose the federated learning with heterogeneous differential privacy algorithm, referred to as FedHDP, for the heterogeneous privacy setup. The FedHDP algorithm is Bayes optimal for federated point estimation as it is designed to learn a global model that is a sufficient statistic of client-level data subject to their differential privacy guarantees.
4. Finally, we provide experimental results of the FedHDP algorithm using various synthetic and realistic federated datasets from TensorFlow Federated (TFF) (Google, 2019) using reasonable privacy choices. Although the design guarantees of FedHDP don't apply in these complex settings, we experimentally show that it provides significant gains compared to the baseline DP-FedAvg algorithm and another comparable baseline.

2 Privacy Guarantees within Federated Learning

In this section, we briefly describe the federated learning setup. It consists of a central server, who wishes to learn a model, and a set of clients, who cooperate with the server to learn a model while keeping their data on device. In particular, the central server coordinates the training of the model using the clients over multiple training rounds. The set of all clients, denoted by $\mathcal{C}$, contains all clients that wish to cooperate in training the model. Each client $c_i \in \mathcal{C}$ has a local loss $f_i(\cdot)$ and a local dataset denoted by $\boldsymbol{D}_i = \{\boldsymbol{d}_{i_1}, \boldsymbol{d}_{i_2}, ..., \boldsymbol{d}_{i_{n_i}}\}$, where $\boldsymbol{d}_{i_j}$ is the j-th sample at the i-th client.

During communication round t, the server sends the current model state, i.e., $\boldsymbol{\theta}^t$, to the set of available clients in that round, denoted by $\mathcal{C}^t$, who take multiple gradient steps on the model using their own local datasets to minimize their local loss functions $f_i(\cdot)$. The clients then return the updated model to the server who aggregates them, e.g., by taking the average, to produce the next model state $\boldsymbol{\theta}^{t+1}$. This general procedure describes a large class of learning global models with federated learning, such as federated averaging (FedAvg) (McMahan et al., 2017).

To design privacy-preserving federated learning algorithms using differential privacy, certain modifications to the baseline federated averaging algorithm are required. In particular, the following modifications are introduced: clipping and noising. Considering client-level privacy, the averaging operation at the server is the target of such modifications. Suppose that clients are selected at each round from the population of all clients of size N, with a certain probability denoted by q. First, each client update is clipped to have a norm at most S, then the average is computed followed by adding a Gaussian noise with mean zero and co-variance $\sigma^2 I = z^2(\frac{S}{qN})^2 I$. The variable z is referred to as the noise multiplier, which dictates the achievable values of (ϵ, δ)-DP. Training the model through multiple rounds increases the amount of leaked information. Luckily, the moment accountant method in (Abadi et al., 2016) can be used to provide a tighter estimate of the resulting DP parameters (ϵ, δ). This method achieves client-level differential privacy defined in Definition 1. It is worth noting that the noise can be added at the client side but needs to achieve the desired resulting noise variance in the output of the aggregator at the server, which is still the desired client-level DP.

Selecting the clipping threshold as well as the noise multiplier is essential to obtaining useful models with meaningful privacy guarantees. During training, the norm of updates can either increase or decrease; if the

[1]Homogeneous data refers to the case where the data for all clients is independent and identically distributed (IID).

norm increases or decreases significantly compared to the clipping norm, the algorithm may slow down or diverge. Hence, (Andrew et al., 2019) presented a solution to privately and adaptively update the clipping norm during each round of communication in federated learning based on the feedback from clients on whether or not their update norm exceeded the clipping norm. We consider this as the baseline for privacy-preserving federated learning algorithm and refer to it in the rest of the paper as DP-FedAvg. The case where no noise is added is the baseline for non-private federated learning algorithm, which is referred to simply as Non-Private.

One fundamental aspect of DP-FedAvg is that it provides an equal level of privacy to *all* clients. This naturally arises given the assumption that all clients have similar behavior towards their own privacy in the federated learning setup. In other words, DP-FedAvg implicitly assumes a homogeneity of the privacy level is required by all clients. This is in contrast to the heterogeneity feature of federated learning setups, where different clients have different data, capabilities, and objectives. Next we describe our proposed setup for federated learning with heterogeneous differential privacy.

2.1 Proposed Setup: Heterogeneous Privacy within Federated Learning

The proposed setup for federated learning with heterogeneous differential privacy is as follows. Prior to training, the server presents each client with a set of different privacy parameters $\mathcal{P} = \{(\epsilon_1, \delta_1), (\epsilon_2, \delta_2), ..., (\epsilon_l, \delta_l)\}$. Each client $c_i \in \mathcal{C}$ then makes their choice from the set of privacy parameters based on their desired level of privacy. The server then creates l subsets of clients who share the same choice of privacy parameters, i.e., $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_l$, each with their corresponding privacy parameters. The server then coordinates the training of a global model through updates from clients while ensuring the privacy of each group of clients according to their privacy parameters is met.

We further examine what the server and clients agree upon at the beginning of training a federated learning model in terms of privacy to formally define the considered privacy measures. Each client c_j, whose dataset is denoted as $\boldsymbol{D}_j$ that is disjoint from all other clients, requires the server to apply some randomized algorithm $A_j(\cdot)$, whose image is denoted as $\boldsymbol{O}_j$, such that the following holds

$$\Pr(A_j(\boldsymbol{D}_j) \in O_j) \leq e^{\epsilon_j} \Pr(A_j(\boldsymbol{D}_e) \in O_j) + \delta_j, \tag{2}$$

where $\boldsymbol{D}_e$ is the empty dataset, and the relationship holds for all subsets $O_j \subseteq \boldsymbol{O}_j$. This achieves client-level privacy with parameters (ϵ_j, δ_j) from client c_j's point of view. Let us assume we have N clients, each has their own privacy requirements for the server (ϵ_j, δ_j) for $j \in [N]$, which should hold regardless the choices made by any other client. Now, let us have a randomized algorithm $A(\cdot)$, which denotes the composition of all $A_j(\cdot)$'s; then, the parallel composition property of differential privacy states that the algorithm $A(\cdot)$ is (ϵ_c, δ_c)-DP, which satisfies the following:

$$\Pr(A(\boldsymbol{D}) \in O) \leq e^{\epsilon_c} \Pr(A(\boldsymbol{D}') \in O) + \delta_c, \tag{3}$$

where $\boldsymbol{D}$ contains all datasets from all clients and $\boldsymbol{D}'$ contains datasets from all clients but one, $\boldsymbol{O}$ is the image of $A(\cdot)$, and the relationship holds for all neighboring datasets $\boldsymbol{D}$ and $\boldsymbol{D}'$ that differ by only one client and all $O \subseteq \boldsymbol{O}$. The parallel composition property of differential privacy states that the resulting $\epsilon_c = \max_j \epsilon_j$, and $\delta_c = \max_j \delta_j$. Next, considering our setup, let us have l sets of private clients $\mathcal{C}_i$'s. Each client in the i-th set of clients requires $(\epsilon_{p_i}, \delta_{p_i})$-DP, and without loss of generality, assume that $\epsilon_{p_i} \geq \epsilon_{p_l}$ and $\delta_{p_i} \geq \delta_{p_l}$ $\forall i < l$. This is the case we consider in this paper, where we apply a randomized algorithm $A_{p_i}(\cdot)$, whose image is denoted as $\boldsymbol{O}_{p_i}$, to the dataset that includes all clients in the set $\mathcal{C}_i$ and the following holds

$$\Pr(A_{p_i}(\boldsymbol{D}_{p_i}) \in O_{p_i}) \leq e^{\epsilon_{p_i}} \Pr(A_{p_i}(\boldsymbol{D}'_{p_i}) \in O_{p_i}) + \delta_{p_i}, \tag{4}$$

where $\boldsymbol{D}_{p_i}$ contains all datasets from all clients in $\mathcal{C}_i$ and $\boldsymbol{D}'_{p_i}$ contains datasets from all clients in that subset except one, and the relationship holds for all neighboring datasets $\boldsymbol{D}_{p_i}$ and $\boldsymbol{D}'_{p_i}$ that differ by only one client and all $O_{p_i} \subseteq \boldsymbol{O}_{p_i}$.

Now, let us assume in the proposed heterogeneous differential privacy setup that each client in $\mathcal{C}_i$ requires $(\epsilon_{p_i}, \delta_{p_i})$-DP in the sense of (2). As a result, we can see that the only way for DP-FedAvg to *guarantee* meeting the privacy requirement for the clients in $\mathcal{C}_l$ with $(\epsilon_{p_l}, \delta_{p_l})$ is to enforce $(\epsilon_{p_l}, \delta_{p_l})$-DP for *all* clients. In other words, DP-FedAvg needs to be $(\epsilon_{p_l}, \delta_{p_l})$-DP, i.e., it needs to apply the strictest privacy parameters

to all clients in the sense of (3). On the other hand, in our setup we can *guarantee* meeting the privacy requirements for each set of clients by ensuring an $(\epsilon_{p_i}, \delta_{p_i})$-DP for clients in $\mathcal{C}_i$, respectively, in the sense of (4). In other words, we need to only apply the appropriate DP algorithm with its appropriate metrics for each subset of clients to ensure the privacy metrics are met. This in turn results in our setup satisfying the corresponding privacy requirements needed by each set of clients, which are the main targets that need to be achieved in both algorithms from the clients' point of view in terms of their desired privacy levels.

Next, in terms of objectives in federated learning setups, the server's goal is to utilize the clients updates by averaging them to produce the next model state, and in our case, these updates are subject to specific differential privacy conditions. On the other hand, clients have a different objective when it comes to their performance measures. The clients' goal is to minimize their loss function given all other clients datasets including their own. However, since clients do not have access to other clients' raw data, a client desires to use the information from the differentially-private datasets by other clients as well as its own local update to reach a solution. Assume that the client c_j observes all other clients DP datasets $\{\widetilde{\boldsymbol{D}}_i : i \in [N] \backslash j\}$, which are the outputs of a randomized function that satisfies the privacy condition in (2), as well as its own non-DP dataset observation $\boldsymbol{D}_j$, then the client's Bayes optimal solution is

$$\boldsymbol{\theta}_j^* = \arg\min_{\widehat{\boldsymbol{\theta}}_j} \left\{ \mathbb{E}_{\mathcal{D}_j} \left[\ell_j(\widehat{\boldsymbol{\theta}}_j) \middle| \{\widetilde{\boldsymbol{D}}_i : i \in [N] \backslash j\}, \boldsymbol{D}_j \right] \right\}. \qquad \text{(Local Bayes objective)}$$

where $\ell_j(\cdot)$ is the loss function used to train a model that is kept on device, and $\mathcal{D}_j$ is the true distribution of the dataset at client c_j. Notice that clients here do not use their differentially-private local datasets, but rather their raw local datasets, since they do not need any privacy protections in their local models which will be maintained on-device and not shared. Furthermore, this is typically not practical in federated learning setups, due to the fact that even individual datasets as well as individual updates from other clients are not available to the client to utilize, but rather the updated global model. In this case, each client utilizes the global model state $\widehat{\boldsymbol{\theta}}$ to find the following

$$\widehat{\boldsymbol{\theta}}_j^* = \arg\min_{\widehat{\boldsymbol{\theta}}_j} \left\{ \mathbb{E}_{\boldsymbol{D}_j} \left[\ell_j(\widehat{\boldsymbol{\theta}}_j) \middle| \widehat{\boldsymbol{\theta}}, \boldsymbol{D}_j \right] \right\}. \tag{5}$$

We notice that this solution is a form of personalization in federated learning, where clients no longer deploy the global model locally by default, but rather utilize it to derive better local models that perform well on their own local dataset. In the remainder of this paper we will demonstrate this approach's ability to learn good (even optimal as we shall see in the next section) personalized local models compared to baseline private federated learning. Next, we will consider the proposed setup for a simplified federated problem known as the federated point estimation.

3 Analyzing Heterogeneous Privacy Guarantees

In this section, we provide some insights into our proposed solution in a simplified setup known as federated point estimation inspired by the one proposed by Li et al. (2021). As discussed earlier, in the federated learning setup, clients are interested in learning good models that perform best on their local datasets. Specifically, in the federated point estimation setup, clients are interested in learning Bayes optimal models in the sense of (5). In this case, the solution to this problem is a bi-level optimization problem, which can be solved as a personalized federated learning problem. We first start by considering the global estimation on the server and show the proposed solution is Bayes optimal when using the appropriate hyperparameters. Then we consider local estimations for all sets of clients and show that the proposed solution is Bayes optimal for all clients when using appropriate values of the the respective hyperparameters. Then, we show the privacy-utility tradeoff in the proposed personalized setup compared to the baseline. In this part, the federated point estimation with two levels of privacy, namely private and non-private, is presented for simplicity and clarity of discussion. Refer to Appendix B for additional discussions.

3.1 Federated Point Estimation Setup

In this simplified setup, we consider a single round of communication and assume that all clients have the same number of samples, and the effect of clipping to be negligible. Assume that the set of clients is split into two subsets, $\mathcal{C}_{\text{np}}$ and $\mathcal{C}_{\text{p}}$ denoting the set of non-private and private clients, respectively, with $N_{\text{np}} = |\mathcal{C}_{\text{np}}|$

and $N_{\text{p}} = |\mathcal{C}_{\text{p}}|$. Let ρ_{np} denote the fraction of non-private clients, and n_s denote the number of samples held by each client. Also, let us denote the point to be estimated at client c_j as $\phi_j = \phi + p_j$, where ϕ is the parameter to be estimated at the server, $p_j \sim \mathcal{N}(0, \tau^2)$ is the inherent Gaussian noise, which encompasses the non-IID nature in federated learning setups we are interested in. Increasing τ^2 makes the points more unrelated at different clients, i.e., increasing data heterogeneity, and setting $\tau^2 = 0$ denotes the case of IID clients, i.e., data homogeneity. The observed samples at client c_j are denoted by $\boldsymbol{x}_j = \{x_{j,1}, x_{j,2}, ..., x_{j,n_s}\}$, where $x_{j,i} = \phi_j + v_{j,i}$, where $v_{j,i} \sim \mathcal{N}(0, \beta^2)$ is the additive noise in the observations. The loss function at the client is

$$f_j(\phi) = \frac{1}{2}\left(\phi - \frac{1}{n_s}\sum_{i=1}^{n_s} x_{j,i}\right)^2. \tag{6}$$

Let $\alpha^2 = \frac{\beta^2}{n_s}$. Then minimizing $f_j(\phi)$ leads the client to have the estimate $\hat{\phi}_j = \frac{1}{n_s}\sum_{i=1}^{n_s} x_{j,i}$, whose variance is $\sigma_c^2 = \alpha^2 + \tau^2$. For simplicity and clarity of analysis, we move the noise addition process from the server-side to the client side such that when the server aggregates the private clients' updates the resulting noise variance for privacy is equivalent to the desired value by the server. It is worth that the notion of privacy here remains a client-level privacy despite the location of noise addition. We denote the updates sent to the server by client c_j as $\psi_j = \hat{\phi}_j + l_j$, where $l_j = 0$ for non-private clients and $l_j \sim \mathcal{N}(0, N_p\gamma^2)$ for private clients. Also, $\gamma^2 \propto \frac{1}{N_{\text{p}}^2}$ is the desired privacy noise variance at the server, which is related to the value of the noise multiplier z^2. We will refer to our solution and algorithm as FedHDP in the remainder of the paper. The algorithm's pseudocode for federated point estimation is described in Algorithm 2 in Appendix B. In this setup, the server and clients goals are to minimize the Bayes risk (i.e., test error), defined as follows

$$\theta^* := \arg\min_{\widehat{\theta}} \left\{ \mathbb{E}\left[\frac{1}{2}\left(\phi - \widehat{\theta}\right)^2 \middle| \psi_1, ..., \psi_N \right] \right\}. \tag{7}$$

$$\theta_j^* := \arg\min_{\widehat{\theta}} \left\{ \mathbb{E}\left[\frac{1}{2}\left(\phi_j - \widehat{\theta}\right)^2 \middle| \{\psi_i : i \in [N] \backslash j\}, \hat{\phi}_j \right] \right\}. \tag{8}$$

In the case of regression in our algorithm, which will be presented in the next section in detail, the server's goal is to find the following

$$\widehat{\theta}^* := \arg\min_{\widehat{\theta}} \left\{ \frac{1}{2}\left\| \sum_{i\in[N]} w_i\psi_i - \widehat{\theta} \right\|_2^2 \right\}, \tag{9}$$

while each client has a goal of finding the minimizer of their local objective function, i.e.,

$$\widehat{\theta}_j^* := \arg\min_{\widehat{\theta}} \left\{ \frac{1}{2}\|\widehat{\theta} - \widehat{\phi}_j\|_2^2 + \frac{\lambda_j}{2}\|\widehat{\theta} - \widehat{\theta}^*\|_2^2 \right\}. \qquad \text{(Local FedHDP objective)}$$

Next, we will discuss why we chose this case and show that the proposed FedHDP solution converges to the Bayes optimal solution for the server as well as the clients.

3.2 The Case for Federated Learning with Opt-Out Differential Privacy

In the our discussion so far, we assumed that clients have multiple privacy levels to choose from. In realistic setups, clients are expected to be individuals who may not have complete awareness about what each parameter means in terms of their privacy. Therefore, the server needs to make a choice on how these parameters are presented to clients. A special case of which we consider extensively in this paper is the case of opt-out of differential privacy. The server in this spacial case provides only two privacy choices for each client, to be private or non-private. Clients who choose to be private are guaranteed a fixed (ϵ, δ)-DP, while clients who choose otherwise are not private. Moreover, an even better and more realistic solution from the server's point of view is to enable privacy by default for all clients and give each client the option to opt out of privacy if they desire, which is a more practical solution because clients can make informed decisions about their privacy. The opt-out choice can be suitable for different types of clients such as enthusiasts, beta testers, volunteers, and company employees, among others.

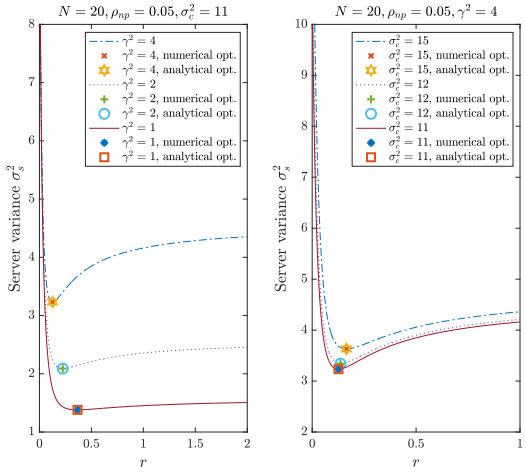

Figure 1: Server noise variance σ_s^2 vs the ratio hyperparameter r. (left) Trade-off for three values of γ^2, (right) trade-off for three values of σ_c^2.

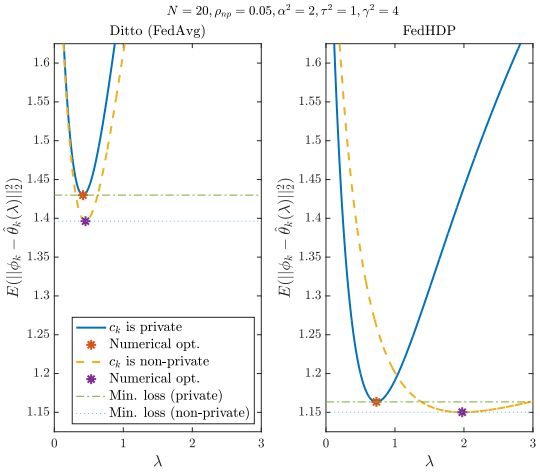

Figure 2: The effect of opting out on the personalized local model estimate as a function of λ when employing (left) DITTO with vanilla FEDAVG and (right) FEDHDP.

3.3 Global Estimate on the Server

The server's goal is to combine the updates received from clients such that the resulting noise variance is minimized, while ensuring the privacy of the set of private clients. To this end, we use Lemma 11 by Mahdavifar et al. (2018) to find the optimal aggregator at the server. The server first computes the two intermediate average values for non-private and private clients as

$$\theta_{\text{np}} = \frac{1}{N_{\text{np}}} \sum_{i \in \mathcal{C}_{\text{np}}} \psi_i, \qquad \theta_{\text{p}} = \frac{1}{N_{\text{p}}} \sum_{i \in \mathcal{C}_{\text{p}}} \psi_i. \tag{10}$$

where $\theta_{\text{np}} \sim \mathcal{N}(\phi, \frac{1}{N_{\text{np}}}\sigma_c^2)$, and $\theta_{\text{p}} \sim \mathcal{N}(\phi, \frac{1}{N_{\text{p}}}\sigma_c^2 + \gamma^2)$. Now, the server aims to combine such values to compute its estimation θ of the value of ϕ with the goal of minimizing the resulting estimation noise variance σ_s^2. In this case, considering the weights used in the weighted average, let us denote the ratio of weights w_i's dedicated for private clients to weights for non-private clients by r.

Lemma 1 (Global estimate optimality). *FEDHDP from the server's point of view, with ratio $r^* = \frac{\sigma_c^2}{\sigma_c^2 + N_p\gamma^2}$, is Bayes optimal (i.e., θ converges to θ^*) in the considered federated point estimation problem. Furthermore, the resulting variance is*

$$\sigma_{s,\text{opt}}^2 = \frac{1}{N}\left[\frac{\sigma_c^2(\sigma_c^2 + N_p\gamma^2)}{\sigma_c^2 + \rho_{np}N_p\gamma^2}\right]. \tag{11}$$

The proof is relegated to the appendix to conserve space. Next, we show some simulation results for the server noise σ_s^2 against the ratio r for different values of σ_c^2 and γ^2 in the federated point estimation setup with N clients and ρ_{np} opt-put fraction of clients. The results are shown in Figure 1, and we can see that the optimal ratio r^* in Lemma 1 minimizes the server variance as expected.

The resulting server noise variance when FEDHDP algorithm is used with vanilla FEDAVG, i.e., $r = 1$ is

$$\sigma_{s,\text{fedavg}}^2 = \frac{1}{N}(\sigma_c^2 + (1 - \rho_{\text{np}})N_p\gamma^2). \tag{12}$$

Lemma 2 (Performance gap between baselines and optimal FEDHDP). *The gap in server's performance between FEDHDP with FEDAVG and the optimal FEDHDP, and the gap between DP-FEDAVG and the optimal FEDHDP are as follows*

$$\sigma_{s,fedavg}^2 - \sigma_{s,opt}^2 = \frac{1}{N}\left(\frac{\rho_{np}(1-\rho_{np})N_p^2\gamma^4}{\sigma_c^2 + \rho_{np}N_p\gamma^2}\right), \tag{13}$$

$$\sigma_{s,dp\text{-}fedavg}^2 - \sigma_{s,opt}^2 = \frac{1}{N}\left(\frac{N_p\gamma^2\rho_{np}(\sigma_c^2 + N_p\gamma^2)}{\sigma_c^2 + \rho_{np}N_p\gamma^2}\right). \tag{14}$$

Note that both (13) and (14) are positive. It can be seen that if the number of clients is large ($N \to \infty$), the gap approaches $(1-\rho_{\text{np}})^2\gamma^2$ and $(1-\rho_{\text{np}})\gamma^2$ in (13) and (14), respectively. This is expected since the noise in the observation itself decreases as the number of clients increase. On the other hand, if $\rho_{\text{np}} \to 0$ or $\rho_{\text{np}} \to 1$, the gap vanishes as expected. Furthermore, if the noise added for privacy γ^2 is large ($\gamma^2 \to \infty$), the gap become significant.

3.4 Personalized Local Estimates on Clients

In this part, we consider using personalization to train local models at clients. We show that the aforementioned solution is Bayes optimal for local estimates at clients. Next, we discuss the optimality of the proposed solution, namely FedHDP, using the estimate θ^* for both private and non-private clients in the federated point estimation problem. To this end, we have the following theorem.

Theorem 3 (Personalized local estimate optimality)**.** *Assuming using FedHDP with ratio r^* in Lemma 1, and using the values λ^*_{np} for non-private clients and λ^*_p for private clients stated below, FedHDP is Bayes optimal (i.e., θ_j converges to θ^*_j for each client $j \in [N]$)*

$$\lambda^*_{np} = \frac{1}{\Upsilon^2}, \tag{15}$$

$$\lambda^*_p = \frac{N + N\Upsilon^2 + (N-N_p)\Gamma^2}{N\Upsilon^2(\Upsilon^2+1) + (N-N_p+1)\Upsilon^2\Gamma^2 + \Gamma^2}. \tag{16}$$

where $\Upsilon^2 = \frac{\tau^2}{\alpha^2}$ *and* $\Gamma^2 = \frac{N_p\gamma^2}{\alpha^2}$.

The proof is relegated to the appendix to conserve space. We notice that the values of λ^* are not equal for private and non-private clients.

The derived expression for the personalization parameters for all clients consider the presence of data heterogeneity as well as privacy heterogeneity. Next, we provide a few examples of corner cases for both λ^*_{p} and λ^*_{np} for the considered federated point estimation problem:

- *homogeneous data*: When all clients have IID samples, then $\tau^2 \to 0$. Resulting in $\lambda^*_{\text{np}} \to \infty$ and $\lambda^*_{\text{p}} \to \frac{N+\Gamma^2(N-N_{\text{p}})}{\Gamma^2}$. Specifically, personalization is needed for the private clients only, while non-private clients utilize the global model.
- *homogeneous privacy*: When $N_{\text{p}} \to N$, then we have $\lambda^*_{\text{p}} \to \frac{N}{\Upsilon^2 N+\Gamma^2}$.
- *homogeneous data and privacy*: When $\tau^2 \to 0$ and $N_{\text{p}} \to N$, then $\lambda^*_{\text{p}} \to \frac{N}{\Gamma^2}$.

Remark: Although the problem is fundamentally different, its solution is similar in spirit to a recently-proposed personalization scheme known as Ditto (Li et al., 2021). FedHDP differs from Ditto in a number of major ways. First, The server-side aggregation in Ditto is the vanilla FedAvg; however, in the proposed solution the server-side aggregation is no longer FedAvg, but rather a new aggregation rule which utilizes the privacy choices made by clients. Second, Ditto is designed for robustness against malicious clients; hence, the performance on malicious clients is not considered. That is not the case in the proposed setup, where measuring the performance for all types of clients, i.e., clients with different privacy levels, is needed, and improving their performance is desired across all sets of clients. Third, the server in Ditto is unaware of the status of the clients, i.e., whether or not they are malicious; while in the proposed setup the server is aware of the privacy choices made by clients and can be utilized during training to improve the performance of the model.

3.5 Privacy-Utility Tradeoff

Earlier in this section, we have shown that for the problem of federated point estimation, the global estimate benefited greatly from the introduced setup of heterogeneous differential privacy. A better global estimate would enable better performance on clients' devices in the federated point estimation setup, even when no personalization is utilized. However, a question may arise on whether clients have a utility cost if they choose to remain private compared to the case where they opt out.

To answer this question, we argue that opting out helps the server to produce a better global estimate, in addition to helping clients to produce better personalized local estimates. In other words, clients that opt out can produce better personalized local estimates compared to the ones that remain private. To illustrate

the *motivation* of opting out for clients, we perform an experiment where we conduct the federated point estimation experiment for two scenarios. The first is the case where client c_k remains private, and the second is the case where c_k opts out of privacy and becomes non-private. For comparison, we provide the results of the experiments of FedHDP with the optimal value r^*, as well as Ditto when using the vanilla FedAvg. The results of these experiments are shown in Figure 2. We can see that if the client is non-private, they exhibit improvements in their estimates using the optimal value λ^* for both algorithms, but the proposed FedHDP with the optimal value r^* greatly outperforms the one with vanilla FedAvg. Additionally, in this problem, we can see that the optimal value of λ^*_{np} for non-private clients is always greater than or equal to the value λ^*_{p} for private clients, which is due to the value of r being less than or equal to 1. In other words, non-private clients have more influence on the global estimate, and hence, encouraging the local estimate to get closer to the global estimate in (Local FedHDP objective) is more meaningful compared to private clients. Furthermore, this experiment illustrates an important trade-off between privacy and utility for each client, where opting out of privacy improves performance, while maintaining privacy incurs degraded performance.

3.6 Extension to Federated Linear Regression and More Complex Models

The extension of the federated point estimation to federated linear regression with two privacy levels along with discussions of its optimality are presented in Appendix A. In this extended setup, we have two subsets of clients $\mathcal{C}_1$ and $\mathcal{C}_2$, each having its own privacy requirements γ_1^2 and γ_2^2, respectively. We perform an analysis of the new setup and show the optimality of the proposed solution. The federated point estimation problem with private and non-private client subsets is a special case of the considered federated linear regression analysis.

Note that the analysis for this simplified setup is considered a first step towards showing the success of the proposed algorithm. Although it does not provide any guarantees beyond the considered federated point estimation and federated linear regression, it gives us some insights into the different factors that can affect the algorithm and its performance. We will see similar trends for more complex setups such as deep models in the experimental evaluation section.

4 FedHDP: Federated Learning with Heterogeneous Differential Privacy

Now that we have been able to find a Bayes optimal solution in the simplified federated point estimation setup, we build upon the ingredients we used to build a general solution for federated learning with heterogeneous differential privacy. We formally present the FedHDP algorithm and elaborate on its hyperparameters. The FedHDP algorithm that is designed to take advantage of the aforementioned heterogeneous privacy setup is described in Algorithm 1. Similarly to the simplified setting, FedHDP utilizes differential privacy with adaptive clipping, upweighting of non-private clients at the server side, and a simple form of personalization.

First, the notations for the variables used in the algorithm are introduced. The set of N clients $\mathcal{C}$ is split into subsets containing clients grouped according to their desired privacy levels, denoted by $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_l$. Let the number of clients in the subset $\mathcal{C}_i$ be denoted by $N_i = |\mathcal{C}_i|$. The rest of the hyperparameters in the algorithm are as follows: the noise multipliers $\boldsymbol{z}, z_b$, the clipping sensitivity S, the learning rate at clients η, the personalized learning rate at clients η_p, quantile κ, and factor η_b. Also, the superscript $(\cdot)^t$ is used to denote a parameter during the t-th training round.

During round t of training, no additional steps are required for the clients during the global model training. Clients train the received model using their local data followed by sending back their clipped updates $\Delta\boldsymbol{\theta}_j^t$ along with their clipping indicator b_j^t to the server. The server collects the updates from clients and performs a *two-step aggregation process*. During the first step, the updates from the clients in each subset $\mathcal{C}_i$ are passed through a (ϵ_i, δ_i) differentially private averaging function to produce $\triangle\tilde{\boldsymbol{\theta}}_i^t$. In the second step of the aggregation the outputs of the previous averaging functions are combined to produce the next iteration of the model. In this step, the server performs a weighted average of the outputs. The weights for this step are chosen based on the number of clients in each subset in that round, the privacy levels, as well as other parameters. This part resembles the weighted averaging considered in the aforementioned federated point estimation problem in Section 3.3. In general, the goal is to give more weight for updates from clients with less strict privacy requirement compared to the ones with stricter privacy requirements. The output of this step is $\triangle\boldsymbol{\theta}^t$, which is then added to the previous model state to produce the next model state.

Algorithm 1 FedHDP: Federated learning with heterogeneous DP

Inputs: model parameters $\boldsymbol{\theta}^0$, sensitivity S^0, learning rate η, personalized learning rate η_p, noise multipliers $\boldsymbol{z}, z_b$, quantile κ, and factor η_b.
Outputs: $\boldsymbol{\theta}^T, \{\boldsymbol{\theta_j}\}_{j\in[N]}$
At server:
for round $t = 0, 1, 2, ..., T-1$ **do**
  $\mathcal{C}^t \leftarrow$ Sample N^t clients from $\mathcal{C}$
  for client c_j in $\mathcal{C}^t$ **in parallel do**
    $\triangle\boldsymbol{\theta}_j^t, b_j^t \leftarrow$ *ClientUpdate*$(\boldsymbol{\theta}^t, c_j, S^t)$
  end for
  for $j \in [l]$ **in parallel do**
    $N_j^t \leftarrow |\mathcal{C}_j^t|, \quad z_j^t \leftarrow z_j \frac{S^t}{N_j^t}$
    $\triangle\tilde{\boldsymbol{\theta}}_j^t \leftarrow \frac{1}{N_j^t}\sum_{c_i\in\mathcal{C}_j^t} \triangle\boldsymbol{\theta}_i^t + \mathcal{N}(\mathbf{0}, (z_j^t)^2\mathbf{I})$
  end for
  $\triangle\boldsymbol{\theta}^t \leftarrow \sum_{i\in[l]} w_i^t \triangle \tilde{\boldsymbol{\theta}}_i^t$
  $\boldsymbol{\theta}^{t+1} \leftarrow \boldsymbol{\theta}^t + \triangle\boldsymbol{\theta}^t$
  $S^{t+1} \leftarrow S^t e^{-\eta_b\left((\frac{1}{N^t}\sum_{i\in\mathcal{C}^t} b_i^t + \mathcal{N}(0, z_b^2\frac{1}{N^t}^2)) - \kappa\right)}$
end for

At client c_j:
ClientUpdate$(\boldsymbol{\theta}^0, c_j, S)$:
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^0$
$\boldsymbol{\theta}_j \leftarrow \boldsymbol{\theta}^0$ (if not initialized)
$\mathcal{B} \leftarrow$ batch the client's data $\boldsymbol{D}_j$
for epoch $e = 1, 2, ..., E$ **do**
  for B in $\mathcal{B}$ **do**
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta\nabla f_j(\boldsymbol{\theta}, B)$
    $\boldsymbol{\theta}_j \leftarrow \boldsymbol{\theta}_j - \eta_p(\nabla f_j(\boldsymbol{\theta}_j, B) + \lambda(\boldsymbol{\theta}_j - \boldsymbol{\theta}^0))$
  end for
end for
$\triangle\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \boldsymbol{\theta}^0$
$b \leftarrow \mathbb{1}_{\|\triangle\boldsymbol{\theta}\|_2 \le S}$
return Clip$(\triangle\boldsymbol{\theta}, S), b$ to server

Clip$(\boldsymbol{\theta}, S)$:
  return $\boldsymbol{\theta} \times \frac{S}{\max(\|\boldsymbol{\theta}\|_2, S)}$ to client

To further elaborate on the averaging weights, let us reconsider the simple setup where we have only two subsets of clients, i.e., $\mathcal{C}_1$ and $\mathcal{C}_2$, with DP parameters (ϵ_1, δ_1) and (ϵ_2, δ_2), respectively. Also suppose that the second subset has stricter privacy requirements, i.e., $\epsilon_1 \ge \epsilon_2$ and $\delta_1 \ge \delta_2$ The weights w_1^t and w_2^t during round t can be expressed as follows $w_1^t = \frac{N_1^t}{N_1^t + rN_2^t}$ and $w_2^t = \frac{rN_2^t}{N_1^t + rN_2^t}$. In general, we desire the value of r be bounded as $0 \le r \le 1$ in FedHDP to use the less-private clients' updates more meaningfully. The first factor to consider when choosing r is related to the desired privacy budget, lower privacy budgets requires more noise to be added, leading to a lower value of r. This intuition was verified in the simplified setting in the previous section. Another factor that is more difficult to quantify is the heterogeneity between the less-private set of clients and the private set of clients. To illustrate this intuition we give the following example. Suppose that the model is being trained on the MNIST dataset where each client has samples of only one digit. Consider two different scenarios: the first is when each of the less-private clients have a digit drawn uniformly from all digits, and the second is when all of the less-private clients have the same digit. It can be argued that the ratio r, when every other hyperparameter is fixed, should be higher in the second scenario compared to the first; since contributions from the more-private clients are more significant to the overall model in the second scenario than the first. This will be experimentally verified in the experiments section presented later.

Then, clients need to train personalized models to be used locally. In the *personalization process*, each client simultaneously continues learning a local model when participating in a training round using the local dataset and the most recent version of the global model received during training and the appropriate value of λ. It is worth noting that the personalization step is similar in spirit to the personalized solution to the federated point estimation problem in Section 3.4. Furthermore, the server keeps track of the privacy loss due to the clients' participation in each round by utilizing the moment accountant method (Abadi et al., 2016) for each set of clients to provide them with tighter bounds on their privacy loss.

5 Experimental Evaluation

Thus far, we showed that FedHDP achieves Bayes optimal performance on a class of linear problems. In this section, we present the results of a number of more realistic experiments to show the utility gain of the proposed FedHDP algorithm with fine-tuned hyperparameters compared to the baseline DP-FedAvg algorithm. Additionally, we compare the performance against another baseline in FedHDP which considers the same privacy guarantees, but applies uniform averaging, i.e., $r = 1$, instead of a weighted averaging

Table 1: Summary of the results of experiments on *synthetic datasets*: We compare the performance of the baseline algorithms against FedHDP with tuned hyperparameters. The variance of the performance metric across clients is between parenthesis.

nonIID MNIST dataset, $(3.6, 10^{-4})$-DP								
Setup		Global model				Personalized local models		
Algorithm	**hyperparameters**	$Acc_g\%$	$Acc_{\mathrm{g,p}}\%$	$Acc_{\mathrm{g,np}}\%$	$\triangle_{\mathrm{g}}\%$	$Acc_{\mathrm{l,p}}\%$	$Acc_{\mathrm{l,np}}\%$	$\triangle_{\mathrm{l}}\%$
Non-Private	$\lambda_{\mathrm{np}}=0.005$	93.8	-	93.75(0.13)	-	-	99.98(0.001)	-
DP-FedAvg	$\lambda_{\mathrm{p}}=0.005$	88.75	88.64(0.39)	-	-	99.97(0.002)	-	-
FedHDP	$r=0.01$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	92.48	92.43(0.30)	93.30(0.21)	0.88	99.94(0.001)	99.94(0.001)	0.0
FedHDP	$r=1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	87.71	87.55(0.42)	88.35(0.34)	0.8	99.97(0.001)	99.93(0.001)	-0.04
Skewed nonIID MNIST dataset, $(3.6, 10^{-4})$-DP								
Non-Private	$\lambda_{\mathrm{np}}=0.005$	93.67	-	93.62(0.15)	-	-	99.98(0.001)	-
DP-FedAvg	$\lambda_{\mathrm{p}}=0.005$	88.93	88.87(0.35)	-	-	99.98(0.001)	-	-
FedHDP	$r=0.1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	90.36	89.96(0.37)	97.45(0.01)	7.49	99.97(0.001)	99.76(0.003)	-0.21
FedHDP	$r=0.9$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	87.96	87.69(0.56)	92.97(0.04)	5.28	99.98(0.001)	99.96(0.001)	-0.02
FedHDP	$r=1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	88.25	88.05(0.39)	89.98(0.05)	1.93	99.97(0.001)	99.85(0.001)	-0.11

at the server. The experiments consider the case where two privacy levels are presented to each client to choose from, to be private or non-private. The experiments show that FedHDP outperforms the baseline algorithms with the right choice of the hyperparameters r, λ in terms of the global model accuracy, as well as in terms of the average personalized local model accuracy.

5.1 Setup

The experiments are conducted on multiple federated datasets, synthetic and realistic. The synthetic datasets are manually created to simulate extreme cases of data heterogeneity often exhibited in federated learning scenarios. The realistic federated datasets are from TFF (Google, 2019), where such datasets are assigned to clients according to some criteria. The synthetic dataset is referred to as the non-IID MNIST dataset, and the number of samples at a client is fixed across all clients. Each client is assigned samples randomly from the subsets of samples each with a single digit between $0-9$. A skewed version of the synthetic dataset is one where non-private clients are sampled from the clients who have the digit 7 in their data. In the non-IID MNIST dataset, we have $2,000$ clients and we randomly sample 5% of them for training each round. The realistic federated datasets are the FMNIST and FEMNIST from TFF datasets. The FMNIST and FEMNIST datasets contain $3,383$ and $3,400$ clients, respectively, and we sample $\sim 3\%$ of them for training each round. TensorFlow Privacy (TFP) (Google, 2018) is used to compute the privacy loss, i.e., the values of (ϵ, δ), incurred during the training phase.

It is worth noting that computing the optimal values of r, λ_{np}, and λ_{p} for non-convex models such as neural networks is not an easy task. To resolve this issue in this section, we treat them as hyperparameters to be tuned. In practice, we cannot compute these parameters analytically, and hence we choose these parameters via grid search on the validation set. Refer to the appendix for an extended description of the used models and their parameters, as well as an extended version of the results.

5.2 Results

In this part, we provide the outcomes of the experiments on the datasets mentioned above. In these experiments, we provide results for an opt-out rate of 5% of the total client population. Clients that opt out are picked randomly from the set of all clients but fixed for a fair comparison across all experiments. The exception for this assumption is for the skewed non-IID MNIST dataset, where clients that opt out are sampled from the clients who have the digit 7. All other hyperparameters are fixed. To evaluate the performance of each algorithm, we measure the following quantities for each dataset:

1. Acc_{g}: the average test accuracy on the *server* test dataset using the global model.
2. $Acc_{\mathrm{g,p}}$, $Acc_{\mathrm{g,np}}$: the average test accuracy of all *private* and *non-private* clients using the global model on their local test datasets, respectively.
3. $Acc_{\mathrm{l,p}}$, $Acc_{\mathrm{l,np}}$: the average test accuracy of all *private* and *non-private* clients using their personalized local models on their local test datasets, respectively.

Table 2: Summarized results of experiments on *realistic federated datasets*: We compare the performance of the baseline algorithms against FedHDP with the hyperparameters that perform best. The variance of the performance metric across clients is between parenthesis.

FMNIST dataset, $(0.6, 10^{-4})$-DP								
Setup		Global model				Personalized local models		
Algorithm	**hyperparameters**	$Acc_g\%$	$Acc_{\mathrm{g,p}}\%$	$Acc_{\mathrm{g,np}}\%$	$\triangle_{\mathrm{g}}\%$	$Acc_{\mathrm{l,p}}\%$	$Acc_{\mathrm{l,np}}\%$	$\triangle_{\mathrm{l}}\%$
Non-Private	$\lambda_{\mathrm{np}}=0.05$	89.65	-	89.35(1.68)	-	-	94.53(0.59)	-
DP-FedAvg	$\lambda_{\mathrm{p}}=0.05$	77.61	77.62(2.55)	-	-	90.04(1.04)	-	-
FedHDP	$r=0.01$, $\lambda_{\mathrm{p}}=0.05$, $\lambda_{\mathrm{np}}=0.005$	86.88	85.36(1.89)	90.02(1.28)	4.66	93.76(0.68)	95.94(0.41)	2.18
FedHDP	$r=1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.005$	75.87	75.77(2.84)	74.41(2.8)	−1.36	90.45(1.02)	92.32(0.8)	1.87
FEMNIST dataset, $(4.1, 10^{-4})$-DP								
Non-Private	$\lambda_{\mathrm{np}}=0.25$	81.66	-	81.79(1.38)	-	-	84.46(0.89)	-
DP-FedAvg	$\lambda_{\mathrm{p}}=0.05$	75.42	75.86(1.82)	-	-	74.69(1.29)	-	-
FedHDP	$r=0.1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.05$	76.52	77.91(1.67)	83.9(1.27)	5.99	77.9(1.22)	79.15(0.99)	1.25
FedHDP	$r=0.01$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.25$	74.86	77.31(2.18)	86.73(0.98)	9.42	81.19(1.02)	84.68(0.78)	3.49
FedHDP	$r=1$, $\lambda_{\mathrm{p}}=\lambda_{\mathrm{np}}=0.05$	75.12	75.87(1.65)	78.59(1.58)	2.72	74.67(1.34)	75.95(1.12)	1.28

4. $\triangle_{\mathrm{g}}$, $\triangle_{\mathrm{l}}$: the gain in the average performance of *non-private* clients over the *private* ones using the global model and the personalized local models on their local test datasets, respectively; computed as $\triangle_{\mathrm{g}} = Acc_{\mathrm{g,np}} - Acc_{\mathrm{g,p}}$ and $\triangle_{\mathrm{l}} = Acc_{\mathrm{l,np}} - Acc_{\mathrm{l,p}}$.

A summary of the results, shown in Table 1 and Table 2, provides the best performance for each experiment along with their corresponding hyperparameters. More detailed results are shown in the appendix. If different values of the hyperparameters in FedHDP yield two competing results, such as one with better global model performance at the server and one with better personalized models at the clients, we show both.

We can see from Tables 1 and 2 that FedHDP allows the server to learn better global models while allowing clients to learn better personalized local models compared to the other baselines, i.e., DP-FedAvg as well as the FedHDP with $r = 1$. For example, the gain due to FedHDP compared to the DP-FedAvg in terms of global model performance is up to 9.27%. For personalized local models, the gain for clients due to FedHDP compared to DP-FedAvg is up to 9.99%. Additionally, we can also see the cost in the average performance in personalized local models between clients who choose to opt out of privacy and clients who choose to remain private. This demonstrates the advantage of opting out, which provides clients with an incentive to opt out of differential privacy to improve their personalized local models, for example, non-private clients can gain up to 3.49% on average in terms of personalized local model performance compared to private clients. It is worth mentioning that opting out can also improve the global model's performance on clients' local data. We observe that there is up to 12.4% gain in the average performance of non-private clients in terms of the accuracy of the global model on the local data compared to the one of baseline DP-FedAvg. Similar trends can be observed for the other baseline.

6 Conclusion

In this paper, we considered a new aspect of heterogeneity in federated learning setups. We proposed a new setup for privacy heterogeneity between clients where privacy levels are no longer fixed for all clients. In this setup, the clients choose their desired privacy levels according to their preferences and inform the server about the choice. We provided a formal treatment for the federated point estimation problem and showed the optimality of the proposed solution on the central server as well as the personalized local models in such setup. Moreover, we have observed that personalization becomes necessary whenever data heterogeneity is present, or privacy is required, or both. We proposed a new algorithm called FedHDP for the considered setup. In FedHDP, the aim is to employ differential privacy to ensure the privacy level desired by each clients are met, and we proposed a two-step aggregation scheme at the server to improve the utility of the model. We also utilize personalization to improve the performance at clients. Finally, we provided a set of experiments on synthetic and realistic federated datasets considering the opt-out of privacy setup. We showed that FedHDP outperforms the baseline private FL algorithm in terms of the global model as well as the personalized local models performance, and showed an the cost of requiring stricter privacy parameters in such scenarios in terms of the gap in the average performance at clients. Finally, broader impacts of this work are discussed in Appendix D.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Maruan Al-Shedivat, Jennifer Gillenwater, Eric Xing, and Afshin Rostamizadeh. Federated learning via posterior averaging: A new perspective and practical algorithms. *arXiv preprint arXiv:2010.05273*, 2020.

Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous differential privacy. *Journal of Privacy and Confidentiality*, 7(2):127–158, 2016.

Mário S Alvim, Miguel E Andrés, Konstantinos Chatzikokolakis, Pierpaolo Degano, and Catuscia Palamidessi. Differential privacy: on the trade-off between utility and information leakage. In *International Workshop on Formal Aspects in Security and Trust*, pp. 39–54. Springer, 2011.

Galen Andrew, Om Thakkar, H Brendan McMahan, and Swaroop Ramaswamy. Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871*, 2019.

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019.

Brendan Avent, Aleksandra Korolova, David Zeber, Torgeir Hovden, and Benjamin Livshits. BLENDER: Enabling local search with a hybrid differential privacy model. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pp. 747–764, 2017.

Amos Beimel, Aleksandra Korolova, Kobbi Nissim, Or Sheffet, and Uri Stemmer. The power of synergy in differential privacy: Combining a small curator with local randomizers. *arXiv preprint arXiv:1912.08951*, 2019.

Alberto Bietti, Chen-Yu Wei, Miroslav Dudik, John Langford, and Steven Wu. Personalization improves privacy-accuracy tradeoffs in federated learning. In *International Conference on Machine Learning*, pp. 1945–1962. PMLR, 2022.

Sebastian Caldas, Jakub Konečny, H Brendan McMahan, and Ameet Talwalkar. Expanding the reach of federated learning by reducing client resource requirements. *arXiv preprint arXiv:1812.07210*, 2018.

Flavio P Calmon, Ali Makhdoumi, and Muriel Médard. Fundamental limits of perfect privacy. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 1796–1800. IEEE, 2015.

Luca Corinzia, Ami Beuret, and Joachim M Buhmann. Variational federated multi-task learning. *arXiv preprint arXiv:1906.06268*, 2019.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

Canh T Dinh, Nguyen H Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. *arXiv preprint arXiv:2006.08848*, 2020.

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

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1322–1333, 2015.

Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

Google. TensorFlow Privacy, 2018. URL http://tensorflow.org/privacy.

Google. TensorFlow Federated, 2019. URL http://tensorflow.org/federated.

Xinran Gu, Kaixuan Huang, Jingzhao Zhang, and Longbo Huang. Fast federated learning in the presence of arbitrary device unavailability. *Advances in Neural Information Processing Systems*, 34:12052–12064, 2021.

Samuel Horvath, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. *Advances in Neural Information Processing Systems*, 34:12876–12889, 2021.

Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

Zach Jorgensen, Ting Yu, and Graham Cormode. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*, pp. 1023–1034. IEEE, 2015.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020.

Mikhail Khodak, Maria-Florina Balcan, and Ameet Talwalkar. Adaptive gradient-based meta-learning methods. *arXiv preprint arXiv:1906.02717*, 2019.

Muah Kim, Onur Günlü, and Rafael F Schaefer. Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2650–2654. IEEE, 2021.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. *International Conference on Machine Learning*, 2021.

Hessam Mahdavifar, Ahmad Beirami, Behrouz Touri, and Jeff S Shamma. Global games with noisy information sharing. *IEEE Transactions on Signal and Information Processing over Networks*, 4(3):497–509, 2018.

Ali Makhdoumi, Salman Salamatian, Nadia Fawaz, and Muriel Médard. From the information bottleneck to the privacy funnel. In *2014 IEEE Information Theory Workshop (ITW 2014)*, pp. 501–505. IEEE, 2014.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

Lalitha Sankar, S Raj Rajagopalan, and H Vincent Poor. Utility-privacy tradeoffs in databases: An information-theoretic approach. *IEEE Transactions on Information Forensics and Security*, 8(6):838–852, 2013.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18. IEEE, 2017.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet Talwalkar. Federated multi-task learning. *arXiv preprint arXiv:1705.10467*, 2017.

Shuang Song, Kamalika Chaudhuri, and Anand Sarwate. Learning from data with heterogeneous noise using SGD. In *Artificial Intelligence and Statistics*, pp. 894–902. PMLR, 2015.

Lichao Sun, Jianwei Qian, Xun Chen, and Philip S Yu. LDP-FL: Practical private aggregation in federated learning with local differential privacy. *arXiv preprint arXiv:2007.15789*, 2020.

Stacey Truex, Ling Liu, Ka-Ho Chow, Mehmet Emre Gursoy, and Wenqi Wei. LDP-Fed: Federated learning with local differential privacy. In *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, pp. 61–66, 2020.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H Yang, Farhad Farokhi, Shi Jin, Tony QS Quek, and H Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

Hangyu Zhu and Yaochu Jin. Multi-objective evolutionary federated learning. *IEEE transactions on neural networks and learning systems*, 31(4):1310–1322, 2019.