Privacy-Preserving Personalized Federated Learning

Rui Hu*, Yuanxiong Guo*, Hongning Li†, Qingqi Pei†, and Yanmin Gong*
*University of Texas at San Antonio, TX, USA 78249
†Xidian University, Xi'an, China, 710071
Emails: {rui.hu, yuanxiong.guo}@utsa.edu, {hnli, qqpei}@xidian.edu.cn, yanmin.gong@utsa.edu

***Abstract*—To provide intelligent and personalized services on smart devices, machine learning techniques have been widely used to learn from data, identify patterns, and make automated decisions. Machine learning processes typically require a large amount of representative data that are often collected through crowdsourcing from end users. However, user data could be sensitive in nature, and learning machine learning models on these data may expose sensitive information of users, violating their privacy. Moreover, to meet the increasing demand of personalized services, these learned models should capture their individual characteristics. This paper proposes a privacy-preserving approach for learning effective personalized models on distributed user data while guaranteeing the differential privacy of user data. Practical issues in a distributed learning system such as user heterogeneity are considered in the proposed approach. Moreover, the convergence property and privacy guarantee of the proposed approach are rigorously analyzed. Experiments on realistic mobile sensing data demonstrate that the proposed approach is robust to high user heterogeneity and offer a trade-off between accuracy and privacy.**

I. Introduction

SMART devices equipped with sensing, communications, computing, and/or control capabilities, such as smartphones, wearable devices, and in-vehicle sensing devices, are becoming extremely popular nowadays. These devices generate, collect, store and analyze an unprecedented amount of data as they interact with the physical world, which can provide intelligent and personalized services to people. For instance, smart watches can record their users' physical activities and mental conditions for health monitoring at any time, and smart insoles can track the body temperature, motion and heart rate of their users to help them stay injury-free and run better.

For these smart devices to provide intelligent services, machine learning techniques need to be applied to learn powerful predictive models on the collected data. A common practice to learn predictive models from these crowdsourced data is to first collect data from all devices in a cloud server and then train a global model. However, it may be risky to store the privacy-sensitive data in a cloud server which may not be fully trustworthy. Moreover, as the data volume increases, the cost and latency of uploading all the raw data to a distant cloud server increase as well. On the other hand, a device may choose to learn a local model on its own data without sharing data with other devices. Such local models often perform poorly due to the limited training data size. Hence, how to benefit from data sharing without violating user privacy in learning predictive models from distributed data is a challenge. *Federated learning* [1] has been proposed recently as a promising approach to solve the challenge. In federated learning, all devices update the global model downloaded from the cloud server with their own data and only send the updates back to the server for aggregation. By sharing only the learned updates rather than the raw data, federated learning both achieves high communication efficiency and reduces privacy risks while obtaining effective predictive models.

Although promising, there remain issues in applying federated learning to the real world. First, the model obtained through federated learning is a shared model that extracts the common knowledge of all participants without capturing personal inclinations [2]. For instance, when learning the sentiment of users on their personal messages, since the same word from different users may convey different sentiments due to various personal opinions and language using habits, a single global model cannot capture such differences. However, since people with close relationships are likely to have similar habits, it will be beneficial to allow the learning tasks of all users to learn from each other based on their relationships. Also known as *multi-task learning*, this kind of method allows personalized models to be learned, which could both benefit from the collective data and keep personal characteristics. Second, in a federated learning system with lots of participants, the device heterogeneity has a large impact on the training efficiency. The network condition, data size and computation capability of different devices are various, leading to the delay, dropout or poor quality of the updates. Third, federated learning does not provide a rigorous privacy guarantee for participants. The server in federated learning is assumed to be fully trusted to coordinate the training. However, the server can easily violate the privacy of participants by observing their updates as shown in recent attacks [3], [4].

To address the aforementioned issues, we propose a novel federated learning scheme that provides an effective personalized model for each participant under the device heterogeneity while guaranteeing differential privacy of their data. In our proposed scheme, the personalized model of a participant is learned based on not only its own local data but also the shared updates computed from other participants' data. We provide differential privacy guarantee for shared updates by adding certain amounts of noises before releasing them. At the heart of our scheme is a new iterative algorithm that solves the multi-task learning optimization problem in a distributed and privacy-preserving way. The iterative algorithm can learn optimal personalized models and the relationship between them simultaneously. Since the algorithm is an iterative process and

978-1-7281-5089-5/20/$31.00 ©2020 IEEE

would consume the privacy budget at each iteration, we further use moments accountant to characterize the end-to-end privacy loss after multiple iterations.

The rest of the paper is organized as follows: We describe the problem setting in Section II and develop distributed solution to achieve personalized federated learning with differential privacy guarantee in Section III. Next, we analyze the privacy guarantee of the proposed solution in Section IV. Then, we provide the convergence rate the proposed solution in Section V. In Section VI, we evaluate our scheme through numerical experiments. Finally, related literature is reviewed in Section VII, and conclusions are made in Section VIII.

II. Problem Setting

We consider a federated learning system as shown in Figure 1. In the system, a group of smart devices (a.k.a., users or participants) will sense the physical world continuously and store the collected data in their local databases. Each device has some embedded computing capabilities capable of training a local model. A cloud server will coordinate the collaboration among devices to improve their models from others' data. Let

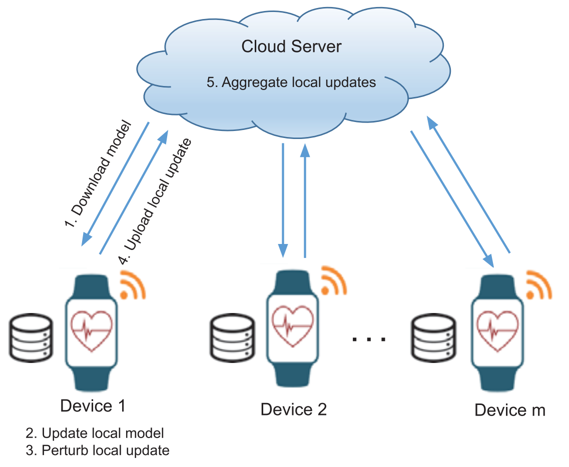


Figure 1: The overall diagram of our system.

m denote the total number of devices in the system, each needing to learn a personalized model. Each device t has a local training dataset $\mathcal{A}_t = (\mathbf{X}_t, \mathbf{y}_t)$, where the i-th column of the matrix $\mathbf{X}_t$ denotes a feature vector $\mathbf{x}_t^i \in \mathbb{R}^d$, and the i-th element of the vector $\mathbf{y}_t$ denotes the corresponding label y_t^i that takes continuous value for regression problems and categorical value for classification problems. Let n_t be the total number of training samples in device t's database, and therefore $\mathbf{X}_t \in \mathbb{R}^{d\times n_t}$. We assume that the feature vector $\|\mathbf{x}_t^i\|_2 \leq 1$ which can be enforced through normalization. Denote by $\mathbf{w}_t \in \mathbb{R}^d$ the model parameters of device t and by $\mathbf{W} := [\mathbf{w}_1, \ldots, \mathbf{w}_m] \in \mathbb{R}^{d\times m}$ the collective model parameters of all users. We use $n := \sum_{t=1}^T n_t$ to denote the total number of all data points and represent the overall feature data matrix as $\mathbf{X} := \mathrm{diag}(\mathbf{X}_1, \cdots, \mathbf{X}_m) \in \mathbb{R}^{md\times n}$. Then the personalized federated learning can be formulated as the following multi-task learning problem [5]:

$$\min_{\mathbf{W},\mathbf{\Omega}} \mathcal{P}(\mathbf{W},\mathbf{\Omega}) := \sum_{t=1}^{m}\sum_{i=1}^{n_t} \ell_t(\mathbf{w}_t^T\mathbf{x}_t^i, y_t^i) + \lambda\,\mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T)$$
$$\text{s.t.} \quad \mathbf{\Omega} \succeq 0, \quad \mathrm{tr}(\mathbf{\Omega}) = 1, \qquad (1)$$

where $\mathbf{\Omega} \in \mathbb{R}^{m\times m}$ is the task covariance matrix that models the relationship between different tasks, $\lambda > 0$ is the regularization parameter, and $\ell_t(\cdot)$ is the convex loss function corresponding to device t's learning task. In the above optimization problem, the first term of the objective function measures the empirical loss of all training samples, and the second term measures the learning task relationship between devices. Note that $\mathcal{P}(\mathbf{W},\mathbf{\Omega})$ is jointly convex with respect to $\mathbf{W}$ and $\mathbf{\Omega}$ under our assumptions as proved in [6].

III. Private-Preserving Distributed Framework

In this section, we propose a distributed framework to solve problem (1) with rigorous privacy guarantee. We first describe the attack model and design goals and then propose a privacy-preserving algorithm to solve the problem.

A. Attack Model and Design Goals

We assume the information sent through the network is well-protected during the transmission and the cloud server is "honest-but-curious". By observing the received updates, it is possible for the server to recover the training data using reconstruction attack [3] or infer whether a sample is in the training dataset with membership inference attack [4]. The goal of our design is to ensure that the server cannot learn much additional information of user samples from the received messages under any auxiliary information and attack. We design our privacy-preserving algorithm in the framework of differential privacy (DP) [7]. A differentially private algorithm provides a strong guarantee that the presence of an individual record in the dataset will not significantly change the output of the algorithm. Specifically, we use the notion of (ϵ,δ)-DP, which is suitable for the iterative algorithm due to its composability property.

In this paper, we achieve (ϵ,δ)-DP for each user using the Gaussian mechanism [7], which provides privacy guarantee through adding Gaussian noise to the uploaded local update. The size of noise is calibrated by the update's *sensitivity* which captures how much a single individual's data changes the value of this update in the worst case. Given any function $f : \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}$ with L_2-sensitivity s_f, the Gaussian mechanism on f is $\mathcal{M}(\mathcal{A}) := f(\mathcal{A}) + \mathcal{N}(0, s_f^2\sigma^2)$, where $\mathcal{N}(0, s_f^2\sigma^2)$ is a normal distribution with mean 0 and standard variance $s_f\sigma$. It has been proved in [7] that the Gaussian mechanism $\mathcal{M}$ achieves (ϵ,δ)-DP if $\sigma \geq \sqrt{2\log(1.25/\delta)}/\epsilon$ with $\epsilon \in (0,1)$.

B. Privacy-Preserving Algorithm

Although it is hard to optimize all the unknown variables of problem (1) simultaneously, since the objective is separable with respect to $\mathbf{W}$ and $\mathbf{\Omega}$, problem (1) can be solved by an alternating optimization procedure [6]. Specifically, we alternatively update $\mathbf{W}$ with fixed $\boldsymbol{\Omega}$ and then update $\boldsymbol{\Omega}$ with fixed $\mathbf{W}$ at each iteration until convergence. In what follows, we present the details of these two steps.

1) Optimize $\boldsymbol{\Omega}$ *with Fixed* $\mathbf{W}$*:* When $\mathbf{W}$ is fixed, the corresponding subproblem becomes minimizing the following:

$$\begin{aligned} \min_{\boldsymbol{\Omega}} \quad & \mathcal{P}(\boldsymbol{\Omega}) := \lambda \operatorname{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T) \\ \text{s.t.} \quad & \boldsymbol{\Omega} \succeq 0, \operatorname{tr}(\boldsymbol{\Omega}) = 1, \end{aligned} \tag{2}$$

which has an analytical solution $\boldsymbol{\Omega}^*$ [6], i.e.,

$$\boldsymbol{\Omega}^* := \frac{(\mathbf{W}^T\mathbf{W})^{1/2}}{\operatorname{tr}((\mathbf{W}^T\mathbf{W})^{1/2})}. \tag{3}$$

We can see that $\boldsymbol{\Omega}^*$ can be computed from the latest $\mathbf{W}$ without requiring any user data, so this step can be preformed efficiently at the cloud server side.

2) Optimize $\mathbf{W}$ *with Fixed* $\boldsymbol{\Omega}$*:* When $\boldsymbol{\Omega}$ is fixed, the subproblem becomes:

$$\min_{\mathbf{W}} \mathcal{P}(\mathbf{W}) := \sum_{t=1}^{m}\sum_{i=1}^{n_t} \ell_t(\mathbf{w}_t^T\mathbf{x}_t^i, y_t^i) + \lambda \operatorname{tr}(\mathbf{W}\boldsymbol{\Omega}^{-1}\mathbf{W}^T). \tag{4}$$

Since the overall dataset $\{\mathcal{A}_t\}_{t=1,\ldots,m}$ is distributed across devices, a distributed and parallel algorithm without requiring expensive raw data transfer is highly desirable.

Towards that goal, we use the block dual coordinate descent considering the fact that the dual of $\mathcal{P}(\mathbf{W})$ has a better separability property. By taking the conjugate dual of $\mathcal{P}(\mathbf{W})$, we obtain the following dual problem:

$$\min_{\boldsymbol{\alpha}} \mathcal{D}(\boldsymbol{\alpha}) := \sum_{t=1}^{m}\sum_{i=1}^{n_t} \ell_t^*(-\alpha_t^i) + \frac{1}{4\lambda}\|\mathbf{X}\boldsymbol{\alpha}\|_{\widetilde{\boldsymbol{\Omega}}}^2, \tag{5}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^n$ is a column vector of all dual variables with the $(\sum_{\tau=1}^{t-1} n_\tau + i)$-th element α_t^i corresponding to the training sample $(\mathbf{x}_t^i, y_t^i)$, ℓ_t^* is the conjugate function of ℓ_t, i.e., $\ell_t^*(-\alpha) = \max_v\{-\alpha v - \ell(v)\}$, and $\widetilde{\boldsymbol{\Omega}} := \boldsymbol{\Omega} \otimes \mathbf{I}_{d\times d} \in \mathbb{R}^{md\times md}$. We assume that ℓ_t^* is convex and differentiable with $|\ell_t^{*\prime}(z)| \le 1$ for all z.

Due to the convexity of problem (4), we have $\mathcal{P}(\mathbf{W}^\star) = \mathcal{D}(\boldsymbol{\alpha}^\star)$, and hence the optimal primal variables can be derived from the optimal dual variables as

$$\mathbf{w}(\boldsymbol{\alpha}) := \frac{1}{2\lambda}\widetilde{\boldsymbol{\Omega}}\mathbf{X}\boldsymbol{\alpha}, \tag{6}$$

where $\mathbf{w}(\boldsymbol{\alpha}) \in \mathbb{R}^{md}$ is a column vector formed by concatenating m blocks of primal variables, with the t-th block vector $\mathbf{w}_t(\boldsymbol{\alpha})$ being the primal variables of device t.

In the following, we develop an iterative search algorithm to solve the dual problem (5). Specifically, given the current solution $\boldsymbol{\alpha}$ and $\mathbf{w}$, we define the following sub-problem for device t at each iteration:

$$\begin{aligned} \min_{\Delta\boldsymbol{\alpha}_t} \mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t; \mathbf{w}_t, \boldsymbol{\alpha}_t) := & \sum_{i=1}^{n_t} \ell_t^*(-\alpha_t^i - \Delta\alpha_t^i) \\ & + \mathbf{w}_t^T\mathbf{X}_t\boldsymbol{\Delta\alpha}_t + \frac{\beta}{4\lambda}\|\mathbf{X}_t\boldsymbol{\Delta\alpha}_t\|_{\widetilde{\boldsymbol{\Omega}}_t}^2, \end{aligned} \tag{7}$$

where $\boldsymbol{\alpha}_t \in \mathbb{R}^{n_t}$ is the t-th block vector of $\boldsymbol{\alpha}$ representing the dual variables of device t, $\widetilde{\boldsymbol{\Omega}}_t \in \mathbb{R}^{d\times d}$ refers to the t-th diagonal block of $\widetilde{\boldsymbol{\Omega}}$, and $\beta > 0$ is the correction parameter.

Note that in the traditional block dual coordinate descent, each local update minimizes the global objective based on all updated coordinates. However, in our approach, each local update minimizes the local objective based on the previous values of local coordinates, which can be executed in parallel and decrease the training efficiency. To compensate for such differences, the correction parameter β needs to be chosen carefully to ensure the sum of the local objectives of all devices approximately equal to the global objective $\mathcal{D}$.

Algorithm 1 Privacy-Preserving Algorithm

```
Input: Datasets $\{\mathcal{A}_t, t = 1, \ldots, m\}$, aggregation parameter
    $\xi \in (0, 1]$, and correction parameter $\beta$.
Initialize: $\boldsymbol{\alpha} \leftarrow \mathbf{0}$, $\mathbf{w} \leftarrow \mathbf{0}$, and $\boldsymbol{\Omega} \leftarrow (1/m)\mathbf{I}$
 1: for $h = 1$ to $H$ do
 2:   for $k = 1$ to $K$ do
 3:     for all devices $t = 1, \cdots, m$ in parallel do
 4:       $\Delta\boldsymbol{\alpha}_t \leftarrow \operatorname{argmin}_{\Delta\boldsymbol{\alpha}_t} \mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t; \mathbf{w}_t, \boldsymbol{\alpha}_t)$;
 5:       $\boldsymbol{\alpha}_t \leftarrow \boldsymbol{\alpha}_t + \xi\Delta\boldsymbol{\alpha}_t$;
 6:       $\Delta\boldsymbol{\alpha}_t = \Delta\boldsymbol{\alpha}_t + \mathbf{b}_t$;
 7:       $\Delta\mathbf{u}_t \leftarrow \xi\mathbf{X}_t\Delta\boldsymbol{\alpha}_t + \mathbf{p}_t$;
 8:       return $\Delta\mathbf{u}_t$ to the server;
 9:     end for
10:     update $\mathbf{w} \leftarrow \mathbf{w} + \frac{1}{2\lambda}\widetilde{\boldsymbol{\Omega}}\Delta\mathbf{u}$ in the server;
11:     send the updated block $\mathbf{w}_t$ back to device $t$;
12:   end for
13:   update $\boldsymbol{\Omega} \leftarrow \frac{(\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}}{\operatorname{tr}((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}})}$ with the most recent $\mathbf{W}$ and
      send the block $\boldsymbol{\Omega}_t$ back to device $t$;
14: end for
```

Algorithm 1 outlines our privacy-preserving algorithm using Gaussian mechanism. Our algorithm contains two parts: (i) update $\mathbf{W}$ (line 2–12); and (ii) update $\boldsymbol{\Omega}$ (line 13). In part (i), each device first solves its own local subproblem (7) and uploads its local parameter update $\Delta\mathbf{u}_t$ to the server. Then the server aggregates $\Delta\mathbf{u}_t$ and updates the global parameters $\mathbf{W}$, which are sent back to the corresponding device. In part (ii), the server updates $\boldsymbol{\Omega}$ using the most recent $\mathbf{W}$ and sends the block result $\widetilde{\boldsymbol{\Omega}}_t$ to the corresponding device. This process iterates multiple rounds until convergence. The noise vectors $\mathbf{b}_t$ and $\mathbf{p}_t$ in line 6 and line 7 are drawn independently from the Gaussian distributions $\mathcal{N}(0, s_1^2\sigma_1^2)$ and $\mathcal{N}(0, s_2^2\sigma_2^2)$ and used to achieve (ϵ_1, δ_1)-DP and $(\epsilon_2.\delta_2)$-DP, respectively. Here, s_1 and s_2 refer to the L_2-sensitivities of $\Delta\boldsymbol{\alpha}_t$ and $\Delta\mathbf{u}_t$, whose values are given in Corollary 1 and Corollary 2, respectively. Due to the limited space, all the proofs of corollaries, lemmas and theorems in this paper are included in supplement.

Corollary 1. *If ℓ^* is convex and differentiable with $|\ell^{*\prime}(z)| \le 1$ for all z, the L_2-sensitivity of $\operatorname{argmin}_{\Delta\boldsymbol{\alpha}_t} \mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t; \mathbf{w}_t, \boldsymbol{\alpha}_t)$ is at most $\frac{1}{\beta n_t}(8\lambda + \beta)$.*

Corollary 2. *When $\Delta\boldsymbol{\alpha}_t$ is known, the L_2-sensitivity of $\Delta\mathbf{u}_t$ is at most $2\xi\|\Delta\boldsymbol{\alpha}_t\|_2$.*

IV. Privacy Analysis

For a mechanism that achieves (ϵ, δ)-DP, the corresponding privacy loss will be bounded by ϵ with probability at least $1-\delta$. The composability property of differential privacy enables us to account the privacy loss at each access to the training data and accumulate this cost for the whole training process. Recently, some advanced composition theorems ([7]–[11]) have been proposed to achieve tighter analysis of the privacy loss for multiple iterations.

Assume that each iteration of Algorithm 1 is (ϵ, δ)-differentially private, by the composability property Algorithm 1 is $(K\epsilon, K\delta)$-differential private after K iterations. While by the strong composition theorem presented in [8] and [9], Algorithm 1 will be $(\epsilon\sqrt{K\log 1/\delta}, K\delta)$-differential private. And this can be further tightened by combining with the privacy amplification theorem proposed in [10] which makes the composed mechanism to be $(O(q\epsilon), O(q\delta))$-differentially private if the training data at each iteration is a random sample from the dataset with sampling probability q. Recently, a stronger method known as moments accountant has been proposed in [11], which saves a $\sqrt{\log 1/\delta}$ factor of the ϵ part and Kq factor of the δ part. In the following, we analyze the privacy loss of our algorithm using the moments accountant.

However, the application of moments accountant in our scenario is not straightforward. Due to the device variability, some devices will run more local iterations and thus use more data. This is analogous to the data sampling, i.e. powerful devices have higher probability. Similarly, some devices will not upload any data to the server because of the node dropping, and this can be mapped to the sampling process of dataset, i.e. devices with poor network connections have lower probability. According to the amplification theorem we mentioned before, the privacy loss can be tightened if we leverage these existing system uncertainties caused by device heterogeneity. Using q to denote the sampling probability of data due to device variability, we first account the privacy loss incurred at each global iteration in Lemma 1.

Lemma 1 (Privacy Loss at Each Iteration)**.** *Assume s_1 and s_2 are the sensitivities of $\Delta\boldsymbol{\alpha}_t$ and $\Delta\mathbf{u}_t$ respectively, $\delta_1 = \delta_2 = \delta \in (0,1)$, and $\epsilon_1, \epsilon_2 \in (0,1)$. Given the sampling probability of data q, Algorithm 1 is $(\sqrt{q^2\epsilon_1^2 + \epsilon_2^2}, \delta)$-DP for device t at each global iteration if*

$$\sigma_1 \geq \frac{\sqrt{2\ln\frac{1.25}{\delta}}}{\epsilon_1}, \quad \sigma_2 \geq \frac{\sqrt{2\ln\frac{1.25}{\delta}}}{\epsilon_2}.$$

Now, we account the privacy loss of the whole algorithm given the sampling probability of dataset p caused by node dropping. We give the conclusion in Lemma 2 in terms of a privacy guarantee for Algorithm 1, which helps us to allocate the Gaussian noise directly.

Lemma 2 (Overall Privacy Loss)**.** *There exist constants c_1 and c_2 so that given the sampling probability of dataset p and the number of global iteration K, for any $\epsilon < c_1 p^2 K$, Algorithm 1 is (ϵ, δ)-differential private for any $\delta > 0$ if we choose*

$$\sigma_1 \geq c_2 \frac{p\sqrt{(q^2+r^2)K\log(1/\delta)}}{\epsilon},$$
$$\sigma_2 \geq c_2 \frac{p\sqrt{(q^2+r^2)K\log(1/\delta)}}{r\epsilon},$$

where the parameter r refers to the ratio of privacy budgets at step 6 and step 7, i.e. $\epsilon_2 = r\epsilon_1$ with $r \geq 0$.

V. Convergence Analysis

Since problem (1) is jointly convex with respect to $\mathbf{W}$ and $\boldsymbol{\Omega}$, the alternating optimization is guaranteed to converge to the optimal solution. Since it is easy to optimize $\boldsymbol{\Omega}$, we focus on the convergence of updating $\mathbf{W}$ in the rest of this section. Following the discussion of the device heterogeneity, we first introduce an approximation parameter to quantify the quality of each update.

Definition 1 (Quality of Update)**.** *At each iteration k, we define the quality measurement of the solution calculated by device t to its subproblem as:*

$$\theta_t^k = \frac{\mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t^k; \mathbf{w}_t^k, \boldsymbol{\alpha}_t^k) - \mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t^\star; \mathbf{w}_t^k, \boldsymbol{\alpha}_t^k)}{\mathcal{G}_t^\beta(\mathbf{0}; \mathbf{w}_t^k, \boldsymbol{\alpha}_t^k) - \mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t^\star; \mathbf{w}_t^k, \boldsymbol{\alpha}_t^k)}, \tag{8}$$

where $\theta_t^k \in [0,1]$ and $\Delta\boldsymbol{\alpha}_t^\star$ is the exact minimizer of $\mathcal{G}_t^\beta(\Delta\boldsymbol{\alpha}_t^k; \mathbf{w}_t^k, \boldsymbol{\alpha}_t^k)$. $\theta_t^k = 0$ refers that the update is the exact solution, and $\theta_t^k = 1$ indicates that the update of model t makes no progress at iteration k.

To provide convergence guarantees, we assume that a device will not drop all the time, and that the update at each iteration will be better than the previous one on average in the following sense.

Assumption 1. *Let $\mathcal{I}_k = (\boldsymbol{\alpha}^k, \boldsymbol{\alpha}^{k-1}, \cdots, \boldsymbol{\alpha}^1)$ be a vector of previous dual variables until iteration k, the expectation of θ_t^k under previous values is $\Theta_t^k = \mathbb{E}(\theta_t^k|\mathcal{I}_k)$. We assume that $P(\theta_t^k = 1) \leq p_{\max}$ with $0 \leq p_{\max} < 1$ and $\mathbb{E}(\theta_t^k|\mathcal{I}_k, \theta_t^k < 1) \leq \Theta_{\max}$ with $0 \leq \Theta_{\max} < 1$.*

Based on Assumption 1, we derive the following theorem which characterizes the convergence of our privacy-preserving algorithm with respect to L-Lipschitz loss functions.

Theorem 1. *Assume the loss function ℓ_t is L-Lipschitz. Under Assumption 1 when*

$$K \geq K_0 + \left\lceil \frac{1}{(1-\xi)(1-\overline{\Theta})} \max\left(1, \frac{2mL^2\sum n_t^2}{\lambda n^2 \epsilon_G}\right) \right\rceil,$$
$$K_0 \geq k_0 + \left\lceil \frac{2}{(1-\xi)(1-\overline{\Theta})} \left(\frac{4mL^2\sum n_t^2}{\lambda n^2 \epsilon_G} - 1\right) \right\rceil,$$
$$k_0 \geq \max\left(0, \left\lceil (1-\xi)(1-\overline{\Theta}) \log \frac{(\mathcal{D}(\boldsymbol{\alpha}^0) - \mathcal{D}(\boldsymbol{\alpha}^*))}{mL^2\sum n_t^2/2\lambda n^2} \right\rceil \right),$$

it holds that $\mathbb{E}(\mathcal{D}(\overline{\boldsymbol{\alpha}}) - \mathcal{D}(\boldsymbol{\alpha}^*)) \leq \epsilon_G$ at the averaged iterate $\overline{\boldsymbol{\alpha}} = \frac{1}{K-K_0}\sum_{k=K_0+1}^K \boldsymbol{\alpha}^k$. Here, $\overline{\Theta} := p_{max} + (1 - p_{max})\Theta_{max}$.

VI. Evaluation

We evaluate our algorithm on the HAR dataset (Human Activity Recognition Using Smartphones Data Set) [12]. It is collected by monitoring six different activities of 30 individuals, using the accelerometer and gyroscope embedded in their mobile phones. The dataset includes 10299 instances in total with 561 features, and 210-306 instances per individual. All data is normalized locally by l_2-normalization. For each participant, we use 75% of their data for training and 25% for testing. We use the hinge loss $\ell(u) = \max(0, 1 - yu)$ as the loss function. It is L-Lipschitz, and its dual is $\ell^*(-\alpha) = -\alpha y$ with $\alpha y \in [0, 1]$. We use the Stochastic Dual Coordinate Ascent (SDCA) as the local solver which selects one coordinate to update randomly at each iteration [13]. For each experiment, we use grid search to choose the best regularization parameter λ and the best ratio r of privacy budgets at step 6 and step 7. The maximum global iteration number is 2000 by default.

A. System performance

We evaluate our system performance in the heterogeneous scenario considering device variability and node dropping, in which both private and non-private cases are studied. In the non-private case, no noises are added on the updates in Algorithm 1. In the private case, Gaussian noises are added as shown in Algorithm 1 to achieve (ϵ, δ)-DP where $\epsilon = 8$ and $\delta = 10^{-3}$. Since we use SDCA as the local solver which samples the data point with probability $1/n_t$ at each iteration, the sampling probability of data $q = n_{iter}/n_t$ where n_{iter} is the local iteration number of device t. The sampling probability of dataset $p = 1 - P(\theta_t^k = 1)$. Thus, in each scenario, we can calculate the privacy budget and the size of noise per iteration by Lemma 1 and Lemma 2. In the heterogeneous scenario, all devices have to upload their updates in a fixed global clock cycle at each time and device will drop with certain probability $P(\theta_t^k = 1)$. More precisely, we simulate the device variability via varying the local iteration numbers of devices. We use $\tau \in [0, 1]$ to measure the device variability level. The local iteration numbers of devices are uniformly distributed between $(1 - \tau)n_{\min}$ and $n_{\min}$ where $n_{\min}$ is the minimum number of local data points across devices.

From Figure 2, we can see that our algorithm converges in the heterogeneous scenario where the device variability level $\tau = 0.5$ and the node dropping probability $P(\theta_t^k = 1) = 0.2$. Note that the estimated time is the overall running time approximately calculated by the total iteration number of the slowest device. In the private case, we calculate the average privacy budget per global iteration $\bar{\epsilon} = 0.23$. Due to the randomness introduced by the Gaussian noise, the non-private algorithm outperforms the private algorithm. In what follows, we are going to estimate the trade-off between privacy and system performance in terms of test error rate.

B. Trade-off between privacy and performance

Since the device heterogeneity and the privacy guarantee influence the system performance simultaneously, we first study the impact of device heterogeneity on the system performance,

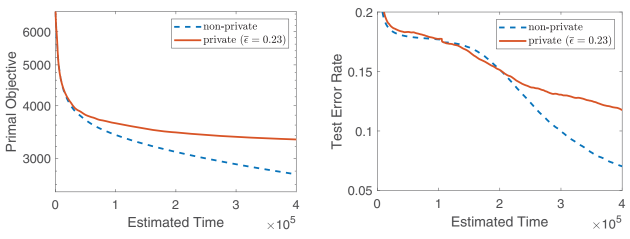
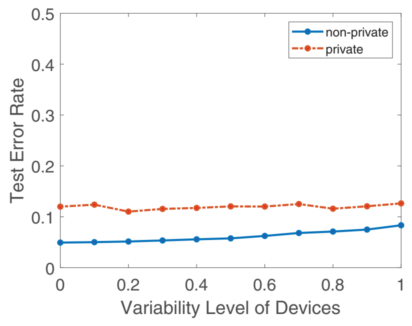


Figure 2: System performance in the heterogeneous scenario considering device variability ($\tau = 0.5$) and node dropping ($P(\theta_t^k = 1) = 0.2$).

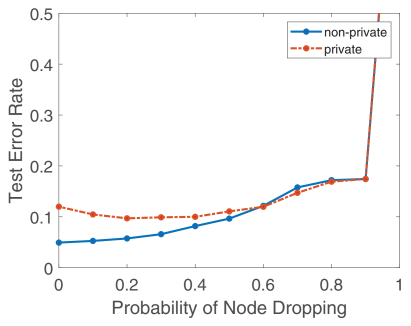
based on which we calculate the trade-off between privacy and system performance. In Figure 3(a), we show the impact of device variability on the system performance. We change the distribution of local iteration numbers via varying the device variability level in the range of $[0, 1]$. Here, we do not consider the node dropping. We test the error rate in both non-private and private case with the same privacy setting in Figure 2. The result shows that as the level of device variability increases, the system performance decreases in the non-private case but does not change much in the private case. Due to the device variance, the local iteration numbers of some devices is smaller than $n_{\min}$ which means the the sampling probability of data $q = n_{iter}/n_t$ on these devices decrease. By Theorem 2, the size of noises is reduced. However, since the device variability will degrade the system performance at the same time, the impact of device variability becomes negligible.

In Figure 3(b), we show the impact of node dropping on the system performance. We vary the node dropping probability from 0 to 1 and test the error rate for the non-private and private case. Here, we do not consider the device variability and we use the same privacy setting in Figure 2 for the private case. The result shows that our system is robust for frequent node dropping. When the node dropping probability is 1, devices will drop all the time and hence the algorithm will not converge. In the non-private case, as the node dropping probability increases, the system performance decreases significantly. However, in the private case, we can see that the error rate first decreases and then increases. There exists an optimal node dropping probability that minimizes the error rate. The randomness introduced by the node dropping both reduces the size of Gaussian noises and increases the error rate. And when $P(\theta_1^k = 1) \leq 0.2$, its effect on the size Gaussian noise is greater than its effect on the error rate so that the error rate decreases. Hence, properly increasing the node dropping probability can help us to achieve better system performance. According to the impact of device heterogeneity on the system performance, we finally calculate the trade-off between the privacy and error rate.

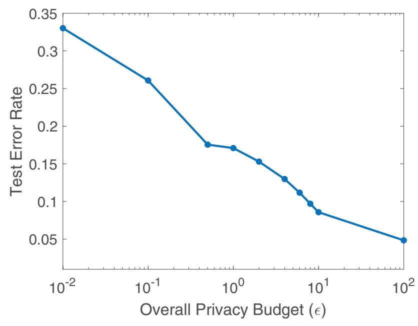
Since the device variability does not have much influence on the privacy and error rate, we set $\tau = 0$. We set the initial node dropping probability as $P(\theta_1^k = 1) = 0.1$ and ϵ vary from 0.01 to 100 with $\delta = 10^{-3}$. For each ϵ, we search the optimal node dropping probability and then update the node



(a) Impact of device variability



(b) Impact of node dropping



(c) Tradeoff between accuracy and privacy

Figure 3: Impact of device heterogeneity and noises on models' accuracy

dropping probability as the optimal probability if the optimal probability is greater than 0.1. The result shows that the error rate varies from about 0.3 to 0.15 when $\epsilon \in (0.01, 1)$ and the error rate is less than 0.15 when ϵ is greater than 1.

VII. Related Work

Most of the work in federated learning focuses the consensus problems [1], [2] with the goal of learning one global model distributedly. In contrast, we tackle the case where the goal is to learn multiple personalized models collaboratively based on relationships among all participants. The relationship among participants can either be learned from the data or be determined from some prior knowledge like the real relationship between devices. recent studies investigate the privacy issue of collaborative learning of personalized models. For example, differentially private multi-task learning is proposed in [14], but they did not consider their learning in a federated setting. Private and personalized learning scheme with a fully decentralized architecture has been proposed in [15], but the architecture with central servers will be more efficient especially for applications that are large-scale and require high system agility and also they did not consider the device heterogeneity. In this paper, we make our distributed personalized learning process to be differentially private and provide rigorous analysis on privacy loss and convergence rate.

VIII. Conclusion

In this paper, we have studied the problem of learning personalized classifiers collaboratively in a privacy-preserving manner. We have considered privacy in the (ϵ, δ)-differential privacy model and provided a privacy-preserving algorithms for the personalized federated learning. We bound the privacy loss by exploiting the existing system uncertainty caused by the device heterogeneity. The proposed approach is robust to device heterogeneity and the perturbation of noises. We have verified the effectiveness of our proposed approach on real mobile sensing data.

Acknowledgment

The work of R. Hu and Y. Gong is supported by the U.S. National Science Foundation under grant CNS-1850523. The work of H. Li and Q. Pei is supported in part by the Key Program of NSFC-Tongyong Union Foundation under Grant U1636209, the National Natural Science Foundation of China under Grant 61902292.

References

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

[2] J. Konecnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[3] M. Al-Rubaie and J. M. Chang, "Reconstruction attacks against mobile-based continuous authentication systems in the cloud," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2648–2663, 2016.

[4] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Security and Privacy (SP), 2017 IEEE Symposium on*, 2017, pp. 3–18.

[5] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4427–4437.

[6] Y. Zhang and D.-Y. Yeung, "A convex formulation for learning task relationships in multi-task learning," *arXiv preprint arXiv:1203.3536*, 2012.

[7] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[8] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, 2010, pp. 51–60.

[9] C. Dwork and G. N. Rothblum, "Concentrated differential privacy," *arXiv preprint arXiv:1603.01887*, 2016.

[10] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

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

[12] J. Reyes-Oritz, D. Anguita, A. Ghio, L. Oneto, and X. Parra, "Human activity recognition using smartphones data set," *UCI Machine Learning Repository; University of California, Irvine, School of Information and Computer Sciences: Irvine, CA, USA*, 2012.

[13] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss minimization," *Journal of Machine Learning Research*, vol. 14, no. Feb, pp. 567–599, 2013.

[14] L. Xie, I. M. Baytas, K. Lin, and J. Zhou, "Privacy-preserving distributed multi-task learning with asynchronous updates," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1195–1204.

[15] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," in *International Conference on Artificial Intelligence and Statistics*, 2017.