Personalized Federated Learning with Differential Privacy

Rui Hu, *Student Member, IEEE,* Yuanxiong Guo, *Senior Member, IEEE,* Hongning Li, *Member, IEEE,* Qingqi Pei, *Senior Member, IEEE,* and Yanmin Gong, *Member, IEEE*

***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 training 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. In addition, the convergence property and privacy guarantee of the proposed approach are rigorously analyzed. Experimental results on realistic mobile sensing data demonstrate that the proposed approach is robust to user heterogeneity and offers a good 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 several issues in applying federated learning to the real world. First of all, 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, a single global model cannot capture such differences, since the same word from different users may convey different sentiments due to various personal opinions and language using habits. 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 learning efficiency. The network condition, data size and computation capability of different devices are different and even time-varying, which may result in the delay, dropout or poor quality of the updates. Third, federated learning does not provide a rigorous privacy guarantee for participants. The aggregation 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

R. Hu and Y. Gong are with the Department of Electrical and Computer Engineering, University of Texas at San Antonio, Texas, 78249 (e-mail: {rui.hu, yanmin.gong}@utsa.edu). Y. Guo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, Texas, 78249 (e-mail: yuanxiong.guo@utsa.edu). H. Li and Q. Pei are with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an, China, 710071 (e-mail: {hnli@, qqpei@mail.}xidian.edu.cn).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.2991416, IEEE Internet of Things Journal

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

In summary, the main contributions of this paper are as follows.

- To our knowledge, this is the first work that rigorously analyzes the personalized federated learning with differential privacy in a heterogeneous IoT setting.
- We propose a novel differentially private federated learning scheme for collaboratively training multiple personalized machine learning models from the data stored across smart devices in IoT.
- We perform rigorous privacy analysis considering the heterogeneity of the IoT devices and convergence analysis for our proposed approach.
- We conduct extensive evaluations based on real-world datasets, verify the effectiveness of the proposed approach, and observe the trade-off among model accuracy and privacy empirically.

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

II. Preliminaries

In this section, we first briefly describe the basics of moments accountant and their properties. The basic idea of moments accountant is to bound the privacy loss by bounding the log moment of the privacy loss. For neighboring databases $\mathcal{A}$ and $\mathcal{A}'$, the randomized mechanism $\mathcal{M}$ and auxiliary input au, the privacy loss incurred by observing o is defined as:

$$z(o;\mathcal{M},au,\mathcal{A},\mathcal{A}') := \log\left(\frac{P_r[\mathcal{M}(au,\mathcal{A})=o]}{P_r[\mathcal{M}(au,\mathcal{A}')=o]}\right).$$

Here, we take the privacy loss $z(o;\mathcal{M},au,\mathcal{A},\mathcal{A}')$ as a random variable because the mechanism $\mathcal{M}$ is randomized.

Accordingly, given the randomized mechanism $\mathcal{M}$ and any positive integer γ, the γ-th moment of the privacy loss $\mu_{\mathcal{M}}(\gamma;au,\mathcal{A},\mathcal{A}')$ is defined as the log of the moment generating function evaluated at γ:

$$\mu_{\mathcal{M}}(\gamma;au,\mathcal{A},\mathcal{A}') := \log \mathbb{E}_{o\sim\mathcal{M}(\mathcal{A})}[\exp(\gamma z(o;\mathcal{M},au,\mathcal{A},\mathcal{A}'))].$$

Then, the moments accountant is defined as the upper bound of $\mu_{\mathcal{M}}(\gamma;au,\mathcal{A},\mathcal{A}')$ over all possible auxiliary information au and neighboring databases $\mathcal{A}$ and $\mathcal{A}'$ as

$$\mu_{\mathcal{M}}(\gamma) := \max_{au,\mathcal{A},\mathcal{A}'} \mu_{\mathcal{M}}(\gamma;au,\mathcal{A},\mathcal{A}').$$

Two important properties of moments accountant are provided below, which will be used compute the moments accountant of an adaptive mechanism and convert the moments accountant into a differential privacy guarantee.

Theorem 1 (Composability [5]). *Suppose that an adaptive mechanism $\mathcal{M}_{1:k}$ consists of a sequence of randomized mechanisms$(\mathcal{M}_1,\mathcal{M}_2,\cdots,\mathcal{M}_k)$ where $\mathcal{M}_i$ takes the dataset $\mathcal{A}$ and the output of $\mathcal{M}_{i-1}$ as its inputs. If the moments accountant of $\mathcal{M}_i$ is $\mu_{\mathcal{M}_i}(\gamma)$, $\forall i\in[k]$, then for any positive integer γ and any output $(o_1,\cdots,o_{k-1})$,*

$$\mu_{\mathcal{M}_{1:k}}(\gamma) = \sum_{i=1}^{k}\mu_{\mathcal{M}_i}(\gamma),$$

where o_i for $i<k$ is the output of mechanism $\mathcal{M}_i$ and $\mu_{\mathcal{M}_{1:k}}(\gamma)$ is conditioned on these $k-1$ outputs.

Theorem 2 (Tail bound [6]). *For any $\epsilon>0$, mechanism $\mathcal{M}$ is (ϵ,δ)-differentially private for*

$$\delta = \min_{\gamma}\exp(\mu_{\mathcal{M}}(\gamma)-\gamma\epsilon).$$

III. 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

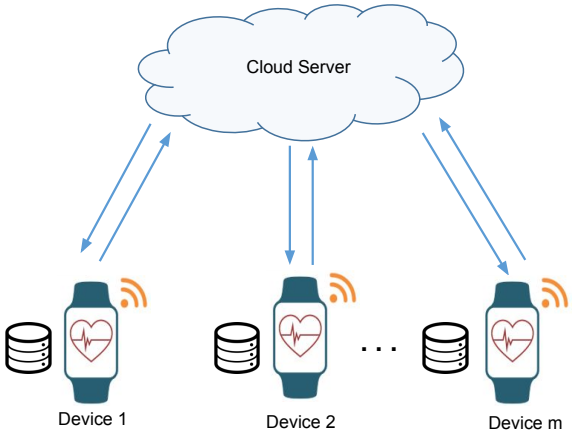


Figure 1: The overall architecture of federated learning systems.

or participants) senses the physical world continuously and stores 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 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 devices. 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 $\mathbf{X} := \text{diag}(\mathbf{X}_1, \cdots, \mathbf{X}_m) \in \mathbb{R}^{md\times n}$. Then the personalized federated learning can be formulated as the following multi-task learning problem [7]:

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

where $\mathbf{\Omega} \in \mathbb{R}^{m\times m}$ is the task covariance matrix that models the relationship between different devices, $\lambda > 0$ 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 [8].

IV. Privacy-Preserving Distributed Framework

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

A. Threat Model and Design Goals

We assume the information sent through the network is well-protected during the transmission and the adversary here can be the "honest-but-curious" central server or users in the system. By observing the received updates, it is possible for the server or malicious users to recover the training data using model inversion 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 or malicious users 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) [9]. 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 Gaussian mechanism [9], 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 [9] 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. Proposed Privacy-Preserving Scheme

Although it is hard to optimize all the unknown variables of problem (1) simultaneously, problem (1) can be solved by an alternating optimization procedure [8] since the objective is separable with respect to $\mathbf{W}$ and $\mathbf{\Omega}$. Specifically, we alternatively update $\mathbf{W}$ with fixed $\mathbf{\Omega}$ and then update $\mathbf{\Omega}$ with fixed $\mathbf{W}$ until convergence. In what follows, we present the details of these two steps.

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

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

which has an analytical solution $\mathbf{\Omega}^*$ [8], i.e.,

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

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

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

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

Since the overall dataset $\{\mathcal{A}_t\}_{t=1}^m$ 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:

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

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 $\widetilde{\mathbf{\Omega}} := \mathbf{\Omega} \otimes \mathbf{I}_{d\times d} \in \mathbb{R}^{md\times md}$. We assume that ℓ_t^* is convex and differentiable with $|\ell_t^{*\prime}(z)| \leq 1$ 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

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

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 variable 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:

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

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 chose 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 $\xi \in (0, 1]$, correction parameter β, inner global iteration number K, and outer global iteration number H.

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: $\Delta\boldsymbol{\alpha}_t \leftarrow \mathrm{argmin}_{\Delta\boldsymbol{\alpha}_t} \mathcal{G}_t^{\beta}(\Delta\boldsymbol{\alpha}_t; \mathbf{w}_t, \boldsymbol{\alpha}_t)$;
5: $\boldsymbol{\alpha}_t \leftarrow \boldsymbol{\alpha}_t + \xi\Delta\boldsymbol{\alpha}_t$;
6: $\Delta\boldsymbol{\alpha}_t \leftarrow \Delta\boldsymbol{\alpha}_t + \mathbf{b}_t$;
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 $\boldsymbol{\Omega} \leftarrow \frac{(\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}}{\mathrm{tr}((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}})}$ with the most recent $\mathbf{W}$ and send the block $\widetilde{\boldsymbol{\Omega}}_t$ back to device t;
14: **end for**

Algorithm 1 outlines our privacy-preserving scheme 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 update $\Delta\mathbf{u}_t$ to the server. Then the server concatenates all $\Delta\mathbf{u}_t$ as $\Delta\mathbf{u}$ 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 represent the L_2-sensitivity of $\Delta\boldsymbol{\alpha}_t$ and $\Delta\mathbf{u}_t$, respectively. Note that H is the number of outer global iterations and K is number of inner global iterations, which should be determined beforehand.

In this paper, we consider a heterogeneous setting where devices have different network conditions, computation capabilities and battery capacities. As described in Algorithm 1, devices need to compute and share their local updates for many iterations. During this process, devices may drop out if they run out of their resources (e.g. running out of power or getting disconnected from the network), which is named as node dropping. In this case, dropped devices are unable to share their computation results to the server. Here, we assume that a device will not always drop out, which means the device will recover and re-join the training process if it dropped out before. Besides, at each global iteration (either inner or outer global iteration), devices perform multiple local iterations in a given time period before sending their computation results to the server. However, as devices have different computation capabilities, they will perform different numbers of local iterations. We name this phenomenon as device variability. In the following, we provide the rigorous analysis of the convergence rate and the privacy loss of our algorithm, considering the node dropping and device variablity.

V. Convergence Analysis

In this section, we analyze the convergence properties of our proposed Algorithm 1. 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. As it is easy to optimize $\boldsymbol{\Omega}$ given $\mathbf{W}$, we focus on analyzing the convergence of updating $\mathbf{W}$ in the rest of this section. Following the discussion of 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:*

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

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.

According to Definition 1, if a device drops out at iteration k, then $\theta_t^k = 1$, otherwise $\theta_t^k \in [0, 1)$. Since a device will not always drop out, we have the probability of node dropping for any device $P(\theta_t^k = 1) < p_{max}$ with $0 \leq p_{\max} < 1$. In addition, due to the device variability, at iteration k, the local updates of devices will have different qualities. Generally, if a device has the computation capability to perform more local iterations, it will achieve higher quality of update. Here, in Assumption 1, we assume that the update of each device at each iteration will be more accurate than the previous one on average, which is a customary assumption for gradient descent-based algorithms [7].

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 and the expectation of θ_t^k under previous values be $\Theta_t^k = \mathbb{E}(\theta_t^k|\mathcal{I}_k)$. We assume that $\mathbb{E}(\theta_t^k|\mathcal{I}_k, \theta_t^k < 1) \leq \Theta_{\max}$ with $0 \leq \Theta_{\max} < 1$.*

Based on Assumption 1, we derive Theorem 3 and Theorem 4 which characterize the convergence of our privacy-preserving algorithm with respect to smooth and non-smooth loss functions, respectively. Before that, we first introduce some lemmas used for both smooth and non-smooth cases.

To make the local dual objective approximate to the global dual objective with respect to varying $\Delta\boldsymbol{\alpha}_t$, we choose β as follows:

Lemma 1. *For any dual variable $\boldsymbol{\alpha} \in \mathbb{R}^n$ and the change of it $\Delta\boldsymbol{\alpha} = (\Delta\boldsymbol{\alpha}_1, \cdots, \Delta\boldsymbol{\alpha}_m)^T$, aggregation parameter $\xi \in (0, 1]$ and correction parameter β, when*

$$\beta \geq \xi \max_{\boldsymbol{\alpha} \in \mathbb{R}^n} \frac{\|\mathbf{X}\boldsymbol{\alpha}\|_{\widetilde{\boldsymbol{\Omega}}}^2}{\sum_{t=1}^m \|\mathbf{X}_t\boldsymbol{\alpha}_t\|_{\widetilde{\boldsymbol{\Omega}}_t}^2}, \tag{9}$$

it holds that

$$\mathcal{D}(\boldsymbol{\alpha} + \xi\Delta\boldsymbol{\alpha}) \leq (1-\xi)\mathcal{D}(\boldsymbol{\alpha}) + \xi \sum_{t=1}^m \mathcal{G}_t^{\beta}(\Delta\boldsymbol{\alpha}_t; \mathbf{w}_t, \boldsymbol{\alpha}_t).$$

Proof: The proof is similar to the proof of Lemma 3 in [10]. ∎

Here we choose $\beta = \xi m$ to ensure that the inequality (9) is always satisfied [10]. Next, we show that the expected objective gap is bounded in Lemma 2 and Lemma 3.

Lemma 2. *For the loss function ℓ_t with its conjugate function ℓ^* is convex and $|\ell_t^{*\prime}| \leq 1$ and any $s \in (0, 1]$,*

$$\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^k + \xi \sum_{t=1}^m \Delta\boldsymbol{\alpha}_t^k)|\mathcal{I}_k] \geq \xi(1-\overline{\Theta})(sG(\boldsymbol{\alpha}^k) + \sum_{t=1}^m J_t), \tag{10}$$

with

$$\begin{cases} G(\boldsymbol{\alpha}^k) := \sum_{t=1}^m \sum_{i=1}^{n_t} \ell_t^*(-\alpha_t^i) + \ell_t(\mathbf{w}_t(\boldsymbol{\alpha})^T\mathbf{x}_t^i, y_t^i) \\ \qquad\qquad + \mathbf{w}_t(\boldsymbol{\alpha})^T\mathbf{x}_t^i\alpha_t^i, \\ J_t := \frac{\mu(1-s)s}{2}\|\mathbf{u}_t - \boldsymbol{\alpha}_t^k\|^2 - \frac{\beta s^2}{4\lambda}\|\mathbf{X}_t(\mathbf{u}_t - \boldsymbol{\alpha}_t^k)\|_{\widetilde{\boldsymbol{\Omega}}_t}^2, \end{cases}$$

where $\mathbf{u}_t = \partial\ell_t(\mathbf{w}_t^T\mathbf{x}_t^i, y_t^i)$ and $\overline{\Theta} = p_{max} + (1-p_{max})\Theta_{max}$.

Proof: The proof is given in Appendix E. ∎

Lemma 3. *For the loss function ℓ_t with its conjugate function ℓ^* is convex and $|\ell_t^{*\prime}| \leq 1$ and any $s \in (0, 1]$,*

$$\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^{k+1})|\mathcal{I}_k] \geq (1-\xi)\xi(1-\overline{\Theta})sG(\boldsymbol{\alpha}^k) + R, \tag{11}$$

with

$$R := \begin{cases} 0, \text{ if } \ell_t \text{ is } (1/\mu)\text{-smooth} \\ -\frac{(1-\xi)}{\lambda}\xi(1-\overline{\Theta})s^2L^2\beta\sum n_t^2, \text{if } \ell_t \text{ is } L\text{-Lipschitz} \end{cases}$$

Here, $G(\boldsymbol{\alpha}^k) = D(\boldsymbol{\alpha}^k) - (-P(\mathbf{W}(\boldsymbol{\alpha}^k)))$ and $\overline{\Theta} = p_{max} + (1-p_{max})\Theta_{max}$.

Proof: The proof is given in Appendix F. ∎

A. Convergence rate of smooth cases

Based on above lemmas, we obtain the convergence rate of our privacy-preserving algorithm for smooth loss functions in Theorem 3.

Theorem 3 (Convergence rate for smooth losses)**.** *Assume the loss function ℓ_t is $(1/\mu)$-smooth such that ℓ_t^* is μ-strongly convex and $|\ell_t^{*\prime}(z)| \leq 1$ for all z. Under Assumption 1, there exist a constant $s \in (0, 1)$ such that*

$$K \geq \frac{1}{(1-\xi)\xi(1-\overline{\Theta})s}\log\left(\frac{n}{\epsilon_G}\right) \tag{12}$$

will satisfy that $\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^K) - \mathcal{D}(\boldsymbol{\alpha}^*) \leq \epsilon_G$. Here, $\overline{\Theta} := p_{max} + (1-p_{max})\Theta_{max}$.

Proof: We first define the duality gap as $G(\boldsymbol{\alpha}) = D(\boldsymbol{\alpha}) - (-P(\mathbf{W}(\boldsymbol{\alpha})))$, and for each sub-problem (7), we choose the correction parameter β and the aggregation parameter ξ according to Lemma 1. To prove Theorem 3 and 4, we first show that $\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^{k+1})|\mathcal{I}_k]$ has a lower bound in Lemma 3 since

$$\begin{aligned}\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^{k+1}) - \mathcal{D}(\boldsymbol{\alpha}^*)|\mathcal{I}_k] &= \mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^*) \\ &\quad - \mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^{k+1})|\mathcal{I}_k]. \end{aligned} \tag{13}$$

Now we derive the upper bound of $\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^{k+1}) - \mathcal{D}(\boldsymbol{\alpha}^*)|\mathcal{I}_k]$ based on above theorems and Equation (13). Following proof is similar to the proof of Theorem 1 in [7]. Since

$$\begin{aligned}&\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^{k+1}) - \mathcal{D}(\boldsymbol{\alpha}^*)|\mathcal{I}_k] \\ &\leq \mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^*) - (1-\xi)\xi(1-\overline{\Theta})sG(\boldsymbol{\alpha}^k) - R \\ &\leq (1 - (1-\xi)\xi(1-\overline{\Theta})s)(\mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^*)) - R, \end{aligned} \tag{14}$$

by recursively applying above inequality, we arrive at

$$\begin{aligned}\mathbb{E}[\mathcal{D}(\boldsymbol{\alpha}^{k+1}) - \mathcal{D}(\boldsymbol{\alpha}^*)|\mathcal{I}_k] &\leq -\frac{R}{(1-\xi)\xi(1-\overline{\Theta})s} \\ &\quad + ((1 - (1-\xi)\xi(1-\overline{\Theta})s))^{k+1}(\mathcal{D}(\boldsymbol{\alpha}^0) - \mathcal{D}(\boldsymbol{\alpha}^*)). \end{aligned} \tag{15}$$

If the loss function ℓ_t is $(1/\mu)$-smooth, $R = 0$ by Lemma 3. Therefore if we denote by $\epsilon_G^k = \mathcal{D}(\boldsymbol{\alpha}^k) - \mathcal{D}(\boldsymbol{\alpha}^*)$ we have that

$$\begin{aligned}\mathbb{E}[\epsilon_G^k|\mathcal{I}_k] &\leq ((1 - (1-\xi)\xi(1-\overline{\Theta})s))^k\epsilon_G^0 \\ &\leq n\exp -k(1-\xi)\xi(1-\overline{\Theta})s. \end{aligned} \tag{16}$$

Here we have $(\mathcal{D}(\boldsymbol{\alpha}^0) - \mathcal{D}(\boldsymbol{\alpha}^*)) \leq n$ by Lemma 10 in [11]. Then, the right hand side will be smaller than some ϵ_G if

$$k \geq \frac{1}{(1-\xi)\xi(1-\overline{\Theta})s}\log\left(\frac{n}{\epsilon_G}\right). \tag{17}$$

Now Theorem 3 follows. ∎

B. Convergence rate of non-smooth cases

Here, we derive the convergence rate of our privacy-preserving algorithm for non-smooth loss functions in Theorem 4.

Theorem 4 (Convergence rate for non-smooth losses)**.** *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, $\overline{\Theta} := p_{max} + (1 - p_{max})\Theta_{max}$.

Proof: The proof of Theorem 4 is similar to the proof of Theorem 3. According to Lemma 2 and Lemma 3, when the loss function ℓ_t is L-Lipschitz, we follow the derivation of Lemma Theorem 9 in [11] and finally obtain that when

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

it holds that $\mathbb{E}(\mathcal{D}(\overline{\boldsymbol{\alpha}}) - \mathcal{D}(\boldsymbol{\alpha}^*)) \leq \epsilon_D$ at the averaged iterate $\overline{\boldsymbol{\alpha}} = \frac{1}{K-K_0} \sum_{k=K_0+1}^{K} \boldsymbol{\alpha}^k$. Then Theorem 4 follows by plugging in parameters $\beta \geq m\xi$. ∎

VI. Privacy Analysis

In this section, we rigorously analyze the end-to-end privacy guarantee of our personalized federated learning scheme. 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 for each access to the training data and accumulate this cost for the whole training process. Recently, some advanced composition theorems ([6], [9], [12]–[14]) 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. When using the composability property, Algorithm 1 is $(K\epsilon, K\delta)$-differential private after K iterations. However, by the strong composition theorem presented in [12] and [13], Algorithm 1 will be $(\epsilon\sqrt{K \log 1/\delta}, K\delta)$-differential private, and this bound can be further tightened by combining with the privacy amplification theorem proposed in [14] 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 [6], 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.

The application of moments accountant to our proposed scheme is not straightforward. Due to the device variability, some devices will run more local iterations (where data is randomly sampled to perform updating at each local iteration) so that more data are accessed at a global iteration, and thus more privacy is leaked. Besides, due to the node dropping, dropped devices are unable to upload their computation results to the server at a certain global iteration and thus no privacy loss occurred at that global iteration. In order to count the overall privacy loss, we let q denote the data sampling probability at each global iteration, which captures the sampling probability of data after multiple local iterations for each device. Then we let p denote the node active probability which equals to $1 - P(\theta_t^k = 1)$ where $P(\theta_t^k = 1)$ represents the probability of node dropping. We first account the privacy loss incurred at each global iteration. Before that, we analyze the L_2-sensitivity of $\Delta\boldsymbol{\alpha}_t$ and $\Delta\mathbf{u}_t$ that calibrate the size of Gaussian noises added to them. To calculate the sensitivities, we first show that the local objective is strongly convex.

Corollary 1. *If ℓ^* is convex and differentiable with $|\ell^{*\prime}(z)| \leq 1$ for all z, then function $\operatorname{argmin}_{\Delta\boldsymbol{\alpha}_t} \mathcal{G}_t^{\beta}(\Delta\boldsymbol{\alpha}_t, \mathcal{A})$ is $\frac{\beta n_t}{2\lambda}$-strongly convex.*

Proof: The proof is given in Appendix A. ∎

Based on Corollary 1, we can estimate the L_2-sensitivity of $\Delta\boldsymbol{\alpha}_t$ and $\Delta\mathbf{u}_t$, whose values are given in Corollary 2 and Corollary 3, respectively.

Corollary 2. *If ℓ^* is convex and differentiable with $|\ell^{*\prime}(z)| \leq 1$ 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 $\frac{1}{\beta n_t}(8\lambda + \beta)$.*

Proof: The proof is given in Appendix B. ∎

Corollary 3. *When $\Delta\boldsymbol{\alpha}_t$ is known, the L_2-sensitivity of $\Delta\mathbf{u}_t$ is at most $2\xi\|\Delta\boldsymbol{\alpha}_t\|_2$.*

Proof: The proof is given in Appendix C. ∎

Next, using the results in Corollary 2 and Corollary 3, we account the privacy loss incurred at each global iteration in Lemma 4. First, we obtain the moments accountant of each global iteration of Algorithm 1, as given in Corollary 4.

Corollary 4. *If $\sigma_1 \geq 1$ and the data sampling probability $q < \frac{1}{16\sigma_1}$. Then, for any integer $\gamma \in (0, \sigma_1^2 \ln \frac{1}{q\sigma_1})$, each global iteration of Algorithm 1 satisfies*

$$\mu_{\mathcal{M}_{1:2}}(\gamma) \leq \frac{\gamma(\gamma+1)}{2\sigma_2^2} + \frac{q^2\gamma(\gamma+1)}{(1-q)\sigma_1^2} + O(q^3\gamma^3/\sigma_1^3).$$

Proof: The proof is given in Appendix D. ∎

Lemma 4 (Per-iteration privacy loss)**.** *Assume s_1 and s_2 are the sensitivity 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 data sampling probability q, Algorithm 1 is $(\sqrt{q^2\epsilon_1^2 + \epsilon_2^2}, \delta)$-DP for device t at each global iteration if*

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

Proof: We compute the moments accountant of each iteration of our algorithm according to the definitions and then obtain the moments accountant for the whole algorithm

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.2991416, IEEE Internet of Things Journal

by Theorem 1. Finally, we can derive the privacy guarantee of our algorithm by converting the moments accountant to a (ϵ, δ)-differentially privacy guarantee according to Theorem 2.

By Corollary 4, we have the moments accountant of iteration k that is $\mu_{\mathcal{M}_k}(\gamma) \leq \frac{q^2\gamma^2}{\sigma_1^2} + \frac{\gamma^2}{\sigma_2^2}$. By Theorem 2, we can convert $\mu_{\mathcal{M}_k}(\gamma)$ into the differentially privacy guarantee of each global iteration of our algorithm. Then, we have following optimization problem to be solved.

$$\log \delta = \min_{\gamma}(q^2\gamma^2/\sigma_1^2 + \gamma^2/\sigma_2^2 - \gamma\epsilon)$$
$$\text{subject to} \quad \gamma \leq \sigma_1^2 \log(\frac{1}{q\sigma_1}), \quad \gamma \in \mathbb{Z}^+,$$
$$q \leq \frac{1}{16\sigma_1}, \quad \sigma_1 \geq 1.$$

Given fixed σ_1 and σ_2 and δ, if we could find some values of $\epsilon > 0$, $c_0 \in (0, 1)$ and a positive integer γ so that

$$(\frac{q^2}{\sigma_1^2} + \frac{1}{\sigma_2^2})\gamma^2 \leq c_0\gamma\epsilon \text{ and } \log\delta \geq (c_0 - 1)\gamma\epsilon,$$

each iteration of Algorithm 1 can be (ϵ, δ)-differentially private. The above two inequalities are equivalent to

$$\frac{\log 1/\delta}{\epsilon(1 - c_0)} \leq \frac{c_0\epsilon}{\frac{q^2}{\sigma_1^2} + \frac{1}{\sigma_2^2}}.$$

We can see that the positive integer γ exists if $\frac{\log(1/\delta)}{\epsilon(1-c_0)} \leq \frac{c_0\epsilon}{q^2/\sigma_1^2 + 1/\sigma_2^2}$, which is equivalent to the following condition

$$\epsilon^2 \geq \frac{\log(1/\delta)(q^2/\sigma_1^2 + 1/\sigma_2^2)}{c_0(1 - c_0)}.$$

It is easy to verify that there exists a constant c_1 such that

$$\epsilon^2 = c_1 \log(1/\delta)(q^2/\sigma_1^2 + 1/\sigma_2^2).$$

Corollary 4 follows by plugging $\sigma_1 = \sqrt{2\ln(1.25/\delta)}/\epsilon_1$ and $\sigma_2 = \sqrt{2\ln(1.25/\delta)}/\epsilon_2$ into above equality. ■

Now, we account the privacy loss of the whole algorithm given the node active probability p. We give the conclusion in Theorem 5 in terms of a privacy guarantee for Algorithm 1, which helps us to allocate the Gaussian noise directly.

Theorem 5 (End-to-end privacy loss)**.** *There exist constants c_1 and c_2 so that given the node active probability 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 $r \geq 0$.

Proof: Assume each iteration of Algorithm 1 is (ϵ, δ)-differentially private, and the iteration number is K. Given the node active probability p, we have that Algorithm 1 is $(\epsilon p\sqrt{K}, \delta)$-differentially private using the moments accountant. Then the result follows by an application of Lemma 4 with ϵ_2 being represented by $\epsilon_2 = r\epsilon_1$. ■

VII. Evaluation

In this section, we evaluate our approach against device heterogeneity and different privacy budgets. We first show the convergence properties of our approach, and then study the impact of device heterogeneity and the trade-off between accuracy and privacy.

A. Experimental Setting

We evaluate our approach on the HAR dataset (Human Activity Recognition Using Smartphones Data Set) [15]. It is collected by monitoring six different activities (walking, walking upstairs, walking downstairs, sitting, standing and laying) of 30 individuals, using the accelerometer and gyroscope embedded in the mobile phone. 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. We train models for each individual and predict between sitting and other activities using 75% of the 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 [16]. Since SDCA samples the data point with probability $1/n_t$ at each iteration, the data sampling probability q is n_{iter}/n_t where n_{iter} is the local iteration number of device t. The node active probability $p = 1 - P(\theta_t^k = 1)$ where $P(\theta_t^k = 1)$ is the probability of node dropping of device t at global iteration k.

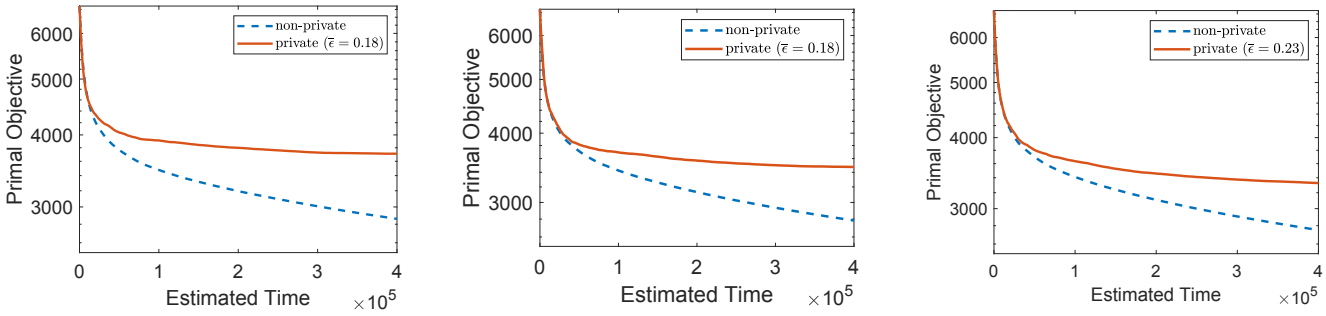
We evaluate our approach in both homogeneous and heterogeneous scenarios. In the homogeneous scenario, the server waits for all devices to upload their updates at each time, and no device will drop out during the process. In the heterogeneous scenario, all devices have to upload their updates in a fixed global clock cycle at each time, and each device will drop out with a certain probability $P(\theta_t^k = 1)$. We simulate the device variability via varying the local iteration numbers of each device. 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. Besides, in each scenario, we compare our approach with the baseline approach, i.e., the non-private personalized federated learning scheme, where no noise is added to the updates.

For each experiment, the number of inner global iterations K is set as 2000 and the number of outer global iterations H is set as 10. Besides, we use a 5-fold cross-validation to choose the best hyperparameters λ and γ. We train and test all models for 10 times and report the average results.

B. Numerical Results

1) Convergence Properties of Our Approach: We evaluate the convergence properties of our approach in both the homogeneous and heterogeneous scenarios considering device variability and node dropping. In our approach, we achieve (ϵ, δ)-differential privacy for each user using Algorithm 1.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.2991416, IEEE Internet of Things Journal



(a) (b) (c)

Figure 2: Convergence properties of our approach: (a) homogeneous setting; (b) heterogeneous setting with device variability ($\tau = 0.5$); and (c) heterogeneous setting with device variability ($\tau = 0.5$) and node dropping ($P(\theta_t^k = 1) = 0.2$).

Specifically, we set $\epsilon = 8$ and $\delta = 10^{-3}$ and calculate the privacy budget of each global iteration $\bar{\epsilon}$ for each experiment.

We first compare the learning progress of our approach with the baseline approach (i.e., the non-private personalized federated learning scheme) under homogeneous and heterogeneous scenarios as shown in Figure 2. Specifically, Figure 2a shows the change of the primal objective value with respect to the overall running time (which is proportional to the iteration number) in the homogeneous scenario. We can observe that our approach will converge quickly. However, due to the addition of random noise at each iteration in our approach, it will only converge to a suboptimal value compared with the non-private approach, which matches the intuition that differential privacy guarantee comes with a utility loss. Similar results are observed under the heterogeneous scenarios, which are shown in Figure 2b (with device variability) and Figure 2c (with both device variability and dropping).

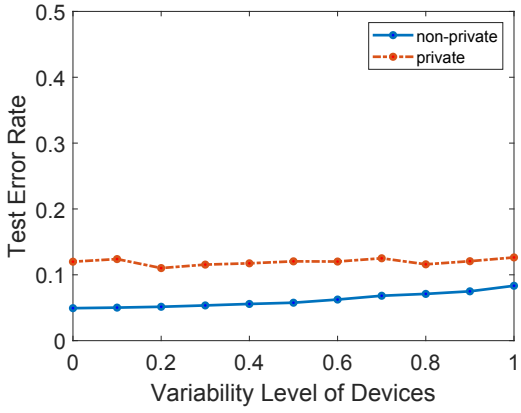


Figure 3: Impact of device variability.

2) Impact of Device Heterogeneity: In this set of experiments, we evaluate the test error rate of the learned models in our approach under different device heterogeneity settings. We first study the impact of device variability by measuring the test error rate with respect to different variability levels of devices τ as shown in Figure 3. As we can observe from the figure, our approach is robust to the device variability and the error rate of the learned model is almost stable even when the variability level increases. In comparison, the non-private baseline approach will be affected more by the device variability. The reason is that in our approach, the randomness introduced by the device variability reduces the size of noises and thus eliminates the impact of privacy on the accuracy.

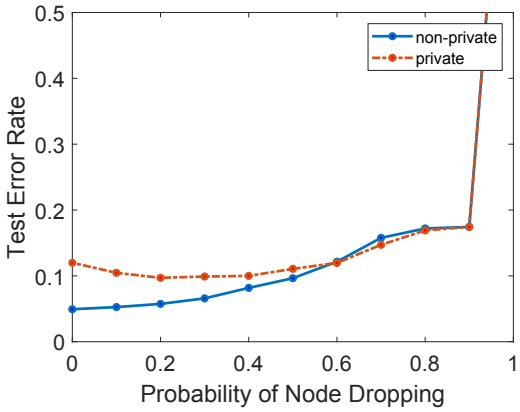


Figure 4: Impact of the drop-out of devices during the training.

Then, we study the impact of node dropping by assigning the probability $P(\theta_t^k = 1)$ to be a random number within $[0, 1]$ as shown in Figure 4. The results show that the error rate of the learned models in our approach first decreases to 0.097 when $P(\theta_t^k = 1) = 0.2$ and then increases as more nodes start to drop out. In our approach, the randomness introduced by the node dropping reduces the size of noises which means the test error rate will decrease. According to the test error rate of the baseline approach, the test error rate increases as the probability of node dropping increases. Thus, there exists an optimal point, i.e., when $P(\theta_t^k = 1) = 0.2$, which generates the minimum test error rate. We can see that node dropping does not always make things worse but brings us benefits sometimes. Therefore, sometimes extra dropping of updates will be needed in order to achieve better accuracy of models while preserving the privacy.

3) Trade-off between Accuracy and Privacy: In the last set of experiments, we measure the error rate of learned models corresponding to different privacy budgets, and in each case we set $\tau = 1$ and $P(\theta_1^k = 1)$ as the optimal probability that

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2020.2991416, IEEE Internet of Things Journal

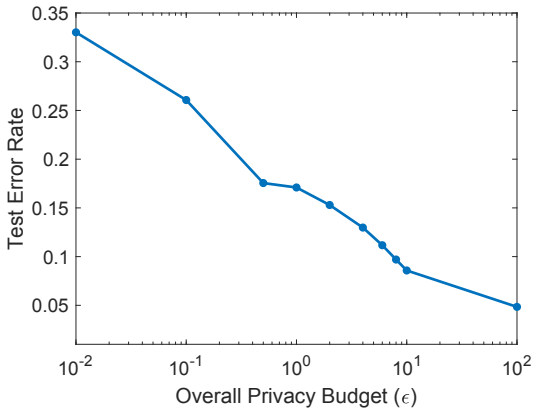


Figure 5: Trade-off between accuracy and privacy.

minimizes the error rate. Here the privacy budget ϵ indicates the overall end-to-end privacy loss for one user and smaller ϵ implies higher privacy. As shown in Figure 5, by increasing the value of ϵ from 0.01 to 100 with $\delta = 10^{-3}$, the corresponding test error rate keeps decreasing, matching the intuition that higher privacy corresponds to lower utility.

VIII. Related Work

Federated learning uses multiple devices to collaboratively train a shared model in an iterative manner while keeping all the data on devices. Specifically, all devices update a model downloaded from a central server independently using their own data and then upload the updates to the server to improve the shared model. Most of the work in federated learning has focused on the consensus problems [1], [2], [17]–[22] which are aimed to learn one global model distributedly. In contrast, we tackle the case where multiple personalized models are trained collaboratively based on relationships among all participants, which is known as multi-task learning.

Multi-task learning can be generally categorized into two categories based on how they capture relationships amongst tasks. The first category (e.g., [7], [8], [23], [24]) assumes that the relationships are not known beforehand and can be learned from the datasets of tasks. On the other hand, the second category (e.g., [25], [26]) assumes that a clustered, sparse, or low-rank structure between the tasks is known a priori. In this paper, we focus on the first category, which is more general and the relationships amongst tasks may not be known beforehand in practice. Moreover, different from the traditional multi-task learning approaches where all learning tasks are performed on a single machine, we consider a federated learning setting where learning tasks are performed on different edge devices and a cloud server will coordinate the learning process. Since devices are heterogeneous in practice, the training process becomes much more complex and challenging. Besides, some works recently studied the problem of collaborative learning of personalized models, however, they did not address any privacy issue [7], [24], [27]. Although some private and personalized learning scheme with a fully decentralized architecture like [28] has been proposed, the architecture with central servers will be more efficient especially for applications that are large-scale and require high system agility. Moreover, they did not consider the device heterogeneity in the real world.

For a distributed system coordinated by a central server, privacy issue arises when an "honest-but-curious" server or device has access to the data or models. There exist several kinds of attacks in addition to the direct access of raw data: reconstruction attacks which recover training data from learned knowledge [3], model inversion attacks which create adversarial example that resemble those used to create the model based on the responses received from that model [29], and membership inference attacks which determine if the sample was a member of the training set through querying the model [4].

Differential privacy [9] is especially effective in preventing membership inference attacks and reconstruction attacks. The differentially private approaches in machine learning can be categorize according to the object it perturbs: one is to directly add noises to the training data [30], another is to add noises to output of training at each iteration or at the end [6]. But many of these approaches are not designed for a distributed system where data are stored on local devices. Therefore, these privacy guarantees are achieved for the whole dataset without including the personalized privacy concern. In this paper, we make our distributed personalized learning process to be differentially private by perturbing the output of training at each iteration and achieve personalized differential privacy for each user under the consideration of device heterogeneity in the real world.

IX. Conclusion

In this paper, we have studied the problem of learning multiple 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 evaluated our approach on real mobile sensing data, showing the impact of device heterogeneity and the trade-off between privacy and accuracy.

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 is supported in part by the Key Program of NSFC-Tongyong Union Foundation under Grant U1636209.

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 *IEEE Symposium on Security and Privacy*, 2017, pp. 3–18.

[5] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," *arXiv preprint arXiv:1610.05755*, 2016.

[6] 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.

[7] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4424–4434.

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

[9] 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.

[10] C. Ma, V. Smith, M. Jaggi, M. I. Jordan, P. Richtárik, and M. Takáč, "Adding vs. averaging in distributed primal-dual optimization," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, 2015, pp. 1973–1982.

[11] V. Smith, S. Forte, C. Ma, M. Takáč, M. I. Jordan, and M. Jaggi, "CoCoA: A general framework for communication-efficient distributed optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 8590–8638, 2017.

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

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

[14] 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.

[15] 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.

[16] 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.

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

[18] Y. Guo and Y. Gong, "Practical collaborative learning for crowdsensing in the internet of things with differential privacy," in *2018 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2018, pp. 1–9.

[19] Y. Gong, Y. Fang, and Y. Guo, "Private data analytics on biomedical sensing data via distributed computation," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 13, no. 3, pp. 431–444, 2016.

[20] ——, "Privacy-preserving collaborative learning for mobile health monitoring," in *2015 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2015, pp. 1–6.

[21] R. Hu, Y. Gong, and Y. Guo, "Cpfed: Communication-efficient and privacy-preserving federated learning," *arXiv preprint arXiv:2003.13761*, 2020.

[22] R. Hu, Y. Guo, E. P. Ratazzi, and Y. Gong, "Differentially private federated learning for resource-constrained internet of things," *arXiv preprint arXiv:2003.12705*, 2020.

[23] L. Jacob, J.-p. Vert, and F. R. Bach, "Clustered multi-task learning: A convex formulation," in *Advances in neural information processing systems*, 2009, pp. 745–752.

[24] S. Liu, S. J. Pan, and Q. Ho, "Distributed multi-task relationship learning," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 937–946.

[25] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 109–117.

[26] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Advances in neural information processing systems*, 2007, pp. 41–48.

[27] J. Wang, M. Kolar, and N. Srebro, "Distributed multi-task learning with shared representation," *arXiv preprint arXiv:1603.02185*, 2016.

[28] P. Vanhaesebrouck, A. Bellet, and M. Tommasi, "Decentralized collaborative learning of personalized models over networks," in *Artificial Intelligence and Statistics*, 2017, pp. 509–517.

[29] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1322–1333.

[30] C. Dwork, K. Talwar, A. Thakurta, and L. Zhang, "Analyze gauss: optimal bounds for privacy-preserving principal component analysis," in *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, 2014, pp. 11–20.

[31] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. Mar, pp. 1069–1109, 2011.

Rui Hu (S'18) received the B.Eng. degree in electrical engineering from Jinan University, China, in 2017. She is currently a PhD student in electrical and computer engineering at the University of Texas at San Antonio. Her research interest includes security and privacy in machine learning, social networks, and cyber-physical systems. She received the Dorrough Distinguished Graduate Fellowship and Graduate Student Professional Development Awards in 2018 and 2019, respectively.

Yuanxiong Guo (SM'19) received the B.Eng. degree in electronics and information engineering from the Huazhong University of Science and Technology, China, in 2009, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida in 2012 and 2014, respectively. He is an Assistant Professor in the Department of Information Systems and Cyber Security at the University of Texas at San Antonio. His research interests include data analytics, security, and privacy.

Hongning Li received the B.S. degree in School of Science and M.S. degree in School of Telecommunications Engineering from Xidian University on 2007 and 2010, respectively, and the Ph.D. degree in School of Computer Science and Technology from Xidian University, Xi'an, China, in 2014. Her research interests include wireless networks and security, security and privacy in cognitive radio networks.

Qingqi Pei received his B.S., M.S. and Ph.D. degrees in Computer Science and Cryptography from Xidian University, in 1998, 2005 and 2008, respectively. He is now a Professor and member of the State Key Laboratory of Integrated Services Networks, also a Professional Member of ACM and Senior Member of IEEE, Senior Member of Chinese Institute of Electronics and China Computer Federation. His research interests focus on digital contents protection and wireless networks and security.

Yanmin Gong received the B.Eng. degree in electronics and information engineering from Huazhong University of Science and Technology, China, in 2009, the M.S. degree in electrical engineering from Tsinghua University, China, in 2012, and the Ph.D. degree in electrical and computer engineering from the University of Florida in 2016. She is an Assistant Professor in the Department of Electrical and Computer Engineering at the University of Texas at San Antonio. Her research interests include security and privacy, wireless networks, and machine learning.