Decentralized Differentially Private Without-Replacement Stochastic Gradient Descent

Richeng Jin, *Member, IEEE*, Xiaofan He, *Senior Member, IEEE*, Huaiyu Dai, *Fellow, IEEE*

***Abstract*—While machine learning has achieved remarkable results in a wide variety of domains, the training of models often requires large datasets that may need to be collected from different individuals. As sensitive information may be contained in the individual's dataset, sharing training data may lead to severe privacy concerns. Therefore, there is a compelling need to develop privacy-aware machine learning methods, for which one effective approach is to leverage the generic framework of differential privacy. Considering that stochastic gradient descent (SGD) is one of the most commonly adopted methods for large-scale machine learning problems, a decentralized differentially private SGD algorithm is proposed in this work. Particularly, we focus on SGD without replacement due to its favorable structure for practical implementation. Both privacy and convergence analysis are provided for the proposed algorithm. Finally, extensive experiments are performed to demonstrate the effectiveness of the proposed method.**

I. Introduction

With the rapid development of wireless sensor networks and smart devices, it is nowadays becoming easier to collaboratively collect data from multiple devices for data processing and analysis. For example, as an important emerging application, health monitoring systems have drawn a lot of attention (e.g., see [1] and the references therein). In a health monitoring system, wearable sensors are used to collect the patients' health data, which are later utilized to develop disease prediction models through machine learning techniques. Considering the size of the systems and the sensitivity of the collected data, there is a compelling need to design efficient decentralized data processing methods. Compared to centralized data processing, the decentralized approaches mainly have two advantages. Firstly, decentralization can offer better scalability by exploiting local computational resource of the smart devices. Secondly, considering that data collected from individuals (e.g., medical and financial records) are sensitive and private, decentralized processing is able to avoid direct data sharing between individual devices and the (possibly) untrusted central node, leading to improved privacy.

Due to its simplicity and scalability, stochastic gradient descent (SGD) has been extensively studied in the literature [2]. SGD admits decentralized implementation by allowing the individuals to compute and share the gradients derived from their local training samples, and hence is suitable for various collaborative learning applications. However, sharing the local gradients may jeopardize the privacy of the users, since an adversary may be able to infer the private local data (e.g., the health information) from the shared gradients [3]. With such consideration, differential privacy [4] has been incorporated into SGD to guarantee a quantifiable level of privacy.

Various differentially private SGD algorithms have been proposed, among which one of the most popular approaches is adding noise to the gradients in the training process (e.g., [5]–[10] and the references therein). Most of the existing works adopt the commonly used independent and identically distributed (i.i.d.) sampling method [11], in which the training examples are sampled in an i.i.d fashion during each training iteration. Nonetheless, in practical implementations of SGD algorithms, without-replacement sampling is often easier and faster, as it allows sequential data access [12]. More specifically, let i_t and $[n]$ denote the index of the training sample used at time t and the whole training dataset, respectively. The mathematical description for i.i.d. sampling is $P(i_t = j) = 1/n, \forall j \in [n]$; for without-replacement sampling, it is $P(i_t = j) = 1/(n-t+1), \forall j \in [n]/\{i_1, \cdots, i_{t-1}\}$. It has been shown that without-replacement sampling is strictly better than i.i.d. sampling after sufficiently many passes over the dataset under smoothness and strong convexity assumptions [13]. [14] considers without-replacement sampling for differentially private SGD. However, it adds noise to the trained models and assumes that the data is held centrally, which cannot be generalized to the decentralized setting directly.

With such consideration, in this work, a decentralized without-replacement sampling SGD algorithm with both privacy and convergence guarantees are proposed. We consider a scenario in which multiple nodes with limited numbers of training samples aim to learn a global model over the whole dataset (i.e., all the training samples from the nodes). It is assumed that each node has two models: a local model that is only available to itself and a global model that is known to the public. At each iteration, a node decides to update either the local model or the global model. To fulfill privacy-aware decentralized SGD, each node adds noise when it updates the global model. As a result, the global model is not necessary better than the local model, especially in the high privacy requirement settings. Therefore, we leverage the deep-Q learning framework [15] to help determine whether each node updates the global model or not during each iteration.

The remainder of this paper is organized as follows. Section II reviews preliminaries and notations used in this work.

R. Jin is with the Zhejiang–Singapore Innovation and AI Joint Research Lab and the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, 310000 (e-mail: richengjin@zju.edu.cn). X. He is with the Electronic Information School, Wuhan University, Wuhan, China, 430000 (e-mail: xiaofanhe@whu.edu.cn). H. Dai is with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC, USA, 27695 (e-mail: hdai@ncsu.edu).

The problem is formulated and presented in Section III. Section IV presents the proposed algorithm, and its effectiveness is examined through simulations in Section V. Conclusions and future works are presented in Section VI.

II. Preliminaries and Notations

In this section, we start by reviewing some important definitions and existing results.

A. Machine Learning and Stochastic Gradient Descent

Suppose that there is a training data set with n training instances (i.e., $\mathcal{S} = \{(x_1, y_1), \cdots, (x_n, y_n)\}$) i.i.d. sampled from a sample space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is a space of feature vectors and $\mathcal{Y}$ is a label space. Let $\mathcal{W} \subseteq \mathbb{R}^d$ be a hypothesis space equipped with the standard inner product and 2-norm $||\cdot||$. The goal is to learn a good prediction model $h(w) \in \mathcal{F} : \mathcal{X} \to \mathcal{Y}$ which is parameterized by $w \in \mathcal{W}$. The prediction accuracy is measured by a loss function $f : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$. Given a hypothesis $w \in \mathcal{W}$ and a training sample $(x_i, y_i) \in \mathcal{S}$, we have a loss $f(w, (x_i, y_i))$. SGD [2] is a popular optimization algorithm, which aims to minimize the empirical risk $F(w) = \frac{1}{n}\sum_{i=1}^{n} f(w, (x_i, y_i))$ over the training dataset $\mathcal{S}$ of n samples and obtain the optimal hypothesis $w^* = \arg\min_w F(w)$. For simplicity, let $f_i(w) = f(w, (x_i, y_i))$ for fixed $\mathcal{S}$. In each iteration, given a training sample (x_t, y_t), SGD updates the hypothesis w_t by:

$$w_{t+1} = G_{f_t,\eta_t} = w_t - \eta_t \nabla f_t(w_t), \tag{1}$$

in which $\nabla f_t(w_t) = \nabla f(w_t, (x_t, y_t))$ is the gradient and η_t is the learning rate. We will denote G_{f_t,η_t} as G_t for ease of presentation.

In order to perform the convergence analysis later, some basic properties of loss functions are defined as follows.

Definition 1. *Let $f : \mathcal{W} \to \mathbb{R}$ be a function: f is convex if for any $u, v \in \mathcal{W}$, $f(u) \geq f(v) + \langle \nabla f(v), u - v \rangle$; f is L-Lipschitz if for any $u, v \in \mathcal{W}$, $||f(u) - f(v)|| \leq L||u - v||$; f is γ-strongly convex if for any $u, v \in \mathcal{W}$, $f(u) \geq f(v) + \langle \nabla f(v), u - v \rangle + \frac{\gamma}{2}||u - v||^2$; f is μ-smooth if for any $u, v \in \mathcal{W}$, $\nabla f(u) - \nabla f(v) \leq \mu||u - v||$.*

Example: **Logistic Regression**. The above three parameters (L, γ, μ) can be derived by analyzing the specific loss function. Here, we give an example using the popular L_2-regularized logistic regression model with the L_2 regularization parameter $\lambda \leq 0$, which can also be found in [14]. Assuming that each feature vector is normalized before processing, i.e., $||x|| \leq 1$, the loss function (for L_2-regularized logistic regression model) on a sample (x, y) with $y \in \{+1, -1\}$ is defined as follows:

$$f(w, (x, y)) = \ln(1 + \exp(-y < w, x >)) + \frac{\lambda}{2}||w||^2. \tag{2}$$

If $\lambda > 0$, the loss function $f(w, (x, y))$ is strongly convex. Suppose the norm of the hypothesis is bounded by R, i.e., $||w|| \leq R$, then it can be proved that $L = 1 + \lambda R, \mu = 1 + \lambda$ and $\gamma = \lambda$. If $\lambda = 0$, the loss function is only convex, and we can deduce that $L = \mu = 1$ and $\gamma = 0$.

We now introduce some important properties of gradient descent updates that will be used in the convergence and privacy analyses of the proposed algorithm.

Definition 2. *Let $G : \mathcal{W} \to \mathcal{W}$ be an operator that maps a hypothesis to another hypothesis. G is ρ-expansive if $sup_{w,w'} \frac{||G(w) - G(w')||}{||w - w'||} \leq \rho$ and σ-bounded if $sup_w ||G(w) - w|| \leq \sigma$.*

Lemma 1. *[14] Assume that f is μ-smooth, if f is convex, then for any $\eta \leq \frac{2}{\mu}$, $G_{f,\eta}$ is 1-expansive; if f is γ-strongly convex, then for $\eta \leq \frac{1}{\mu}$, $G_{f,\eta}$ is $(1 - \eta\gamma)$-expansive.*

Lemma 2. *Suppose that f is L-Lipschitz, then the gradient update $G_{f,\eta}$ is (ηL)-bounded.*

Lemma 3 (Growth Recursion [16])**.** *Fix any two sequences of updates $G_1, \cdots, G_T$ and $G'_1, \cdots, G'_T$. For $t = 1, \cdots, T$, let $w_0 = w'_0$, $w_t = G_t(w_{t-1})$ and $w'_t = G'_t(w'_{t-1})$. Then $||w_0 - w'_0|| = 0$ and for $0 \leq t \leq T$*

$$||w_t - w'_t|| \leq \begin{cases} \rho||w_{t-1} - w'_{t-1}||, \text{if } G_t = G'_t \text{ is } \rho\text{-expansive.} \\ min(\rho, 1)||w_{t-1} - w'_{t-1}|| + 2\sigma_t, \text{if } G_t \text{ is} \\ \rho\text{-expansive; } G_t \text{ and } G'_t \text{ are } \sigma_t\text{-bounded.} \end{cases} \tag{3}$$

B. Differential Privacy

In this subsection, we start by reviewing the definition of differential privacy, and then introduce the Gaussian mechanism that ensures (ϵ, δ)-differential privacy.

Definition 3. *A (randomized) algorithm A is said to be (ϵ, δ)-differentially private if for any neighboring datasets S, S', and any event $E \subseteq Range(A)$, $Pr[A(S) \in E] \leq e^{\epsilon} Pr[A(S') \in E] + \delta$, in which $Range(A)$ is the codomain that consists of all the possible outputs of A.*

Theorem 1. *[17] Let q be a deterministic query that maps a dataset to a vector in $\mathbb{R}^d$. For $c^2 \geq 2ln(1.25/\delta)$, adding Gaussian noise sampled according to*

$$\mathcal{N}(0, \sigma^2); \sigma \geq \frac{c\Delta_2(q)}{\epsilon} \tag{4}$$

ensures (ϵ, δ)-differential privacy for $\epsilon \in (0, 1)$, in which $\Delta_2(q) = max_{S \sim S'} ||q(S) - q(S')||$ is the L_2-sensitivity.

III. Problem Formulation

In this work, a network consisting of M computational nodes is considered. Each node in the network has a local dataset of $\frac{n}{M}$ training samples, and the set of all the training samples from all the nodes in the network form the global training dataset. The goal of the nodes is to collaboratively learn a hypothesis w that minimizes the empirical risk $F(w) = \frac{1}{n}\sum_{i=1}^{n} f(w, (x_i, y_i))$ over the whole training dataset. It is assumed that each node stores two models: a local model (i.e., a local hypothesis w^L) that is only known to itself and a global model (i.e., a global hypothesis w^G) that is shared among all the nodes in the network. All the nodes know the index of the last node that updates the global model and are able to contact it directly. At each iteration, a node randomly samples a mini-batch of training examples from its own local dataset without replacement and determines whether to use and update the global model or not. If a node decides not to update the global model, it simply updates its own local

model; otherwise, it first contacts the last node that has updated the global model and fetches the latest global model. Then it updates the global model using its local model and training samples through the SGD method. Since the global model is publicly known, one can infer the training sample (x, y) in (2) given the loss function f, previous global model w_t and the updated global model w_{t+1}, which leads to privacy concerns and deters the nodes from collaborating. Therefore, each node will add noise to the gradients for privacy preservation.

When the nodes update the global model, they need to contact the other nodes to obtain the latest global model, which induces communication overhead and latency during message passing. Moreover, adding noise may also induce accuracy degradation. Since each node will also learn a local model that is updated without privacy concerns, the local model may sometimes be better than the global model, especially when the privacy requirement is high (i.e., small ϵ). In this sense, each node has to learn a control policy to determine whether to update the global model or not at each iteration.

IV. Deep-Q Learning based Collaborative Decentralized Differentially Private SGD

In this section, a deep-Q learning based collaborative training scheme is proposed. More specifically, the model learning process is modeled as a Markov Decision Process (MDP) [18], in which the collaborative nodes are the agents, the current local models and the loss are the states, and the action for each node is whether updating the global model or not. Reinforcement learning (RL) based methods are commonly used to solve such MDP problems in practice due to two advantages: 1) RL methods do not require prior knowledge of the underlying system dynamics, and 2) the designer is free to choose reward metrics that best match the desired controller performance [19]. There have been some works that employ RL as the controller of optimization algorithms. For example, [20] uses RL to determine the step size of neural network training. Inspired by the success of deep RL methods [15], a deep-Q network is adopted to control the behavior (i.e., updating the local model or the global model) of the nodes.[1]

The deep-Q learning algorithm is presented in Algorithm 2, in which the nodes act as the agents, and the states of the environment are defined by the local models. There are two possible actions for each node: updating the local model or the global model. The basic idea of deep-Q learning is to approximate the action-value (Q) function in traditional Q-learning by a deep neural network. Since RL is known to be unstable when a nonlinear function approximator (i.e., neural network) is used to represent the Q-function, similar to [15], two neural networks are created for each node. The first network θ_t includes all the updates in the training while the second (target) network θ' retrieves the Q values and is periodically updated to imitate the first network. Experience replay is also adopted. After each action, the experience

[1]The deep-Q based method is our first attempt to explore the possibility of using RL as a controller to guide the learning process of the collaborative nodes. The optimization of the controller remains an interesting future work.

Algorithm 1 Deep-Q Learning based Collaborative Decentralized Differentially Private SGD

1. Require: initial vector $w_0^{L_1}, \cdots, w_0^{L_M}, w_0^G$, size of local mini-batch b, number of nodes M, total number of training data samples n, number of iterations T.
2. for $t = 0, 1, \cdots, T$ do
3. for local nodes m:
4. if update, run Algorithm 2 and obtain action $a_t^m \in \{Local, Global\}$
   - If Local, obtain the mini-batch $D_m(t)$, compute the gradient $\nabla f_{D_m(t)}(w_t^{L_m})$ and update its weights $w_{t+1}^{L_m} = w_t^{L_m} - 2\eta_t^{L_m} \nabla f_{D_m(t)}(w_t^{L_m})$
   - If Global, fetch the w_t^G from the latest global model, obtain the mini-batch $D_m(t)$, and compute $\nabla f_{D_m(t)}(w_t^G)$, add noise N_t to the gradient and then update w_{t+1}^G and $w_{t+1}^{L_m}$ according to the following rule

$$w_{t+1}^G = \frac{w_t^G + w_t^{L_m}}{2} - \eta_t^{L_m}(\nabla f_{D_m(t)}(w_t^G) + N_t), \quad (5)$$

$$w_{t+1}^{L_m} = w_{t+1}^G. \quad (6)$$

5. end if
6. end for
7.end for

(transition) is stored in the replay memory as a tuple of $\langle state, action, reward, nextstate \rangle$. During each iteration, a random mini-batch of transitions is sampled and used to update θ_t. For each transition $(s_j, a_j^m, r_j, s_{j+1})$, the target network is used to compute the approximated target value $y_j = r_j + \gamma_{DQ} \max_{a'} \hat{Q}_m(s_{j+1}, a', \theta')$. Based on the current network θ_t and the state s_t, the action a_t^m is determined.

To this end, a Deep-Q learning based collaborative differentially private SGD algorithm (i.e., Algorithm 1) is proposed. For node m, given the training samples $sample_t^m$ and the current local model $w_t^{l_m}$ at time t, it obtains the current state $s_t = [w_t^{L_m}, f(w_t^{L_m}, sample_t^m)]$ and determines to update the global model or the local model via the deep-Q network. After updating the local (or global) model, the updated loss $f(w_{t+1}^{l_m}, sample_t^m)$ is used to update the deep-Q network.

Note that in Algorithm 1, the privacy concern only exists when the nodes update the global model. In (5), there are two terms that may lead to privacy leakage: $w_t^{L_m}$ and $\nabla f_{D_m(t)}(w_t^G)$. Suppose that the latest time that node m updates the global model is $t-j-1$ and therefore $w_{t-j}^{L_m} = w_{t-j}^G$ is publicly known, we have the following Lemma.

Lemma 4. *Suppose that the loss function f is L-Lipschitz, convex and μ-smooth, let $D_m(t-j:t) \triangleq \{D_m(t-j), \cdots, D_m(t)\}$, $D'_m(t-j:t) \triangleq \{D'_m(t-j), \cdots, D'_m(t)\}$ be two neighboring datasets differing at only one sample located in the i-th mini-batch. For Algorithm 1 with $\eta_t^{L_m} \leq \frac{1}{2\mu}, \forall t$, we have*

$$sup_{D_m(t-j:t) \sim D'_m(t-j:t)} ||w_{t+1} - w'_{t+1}|| \leq \max_{k \in [t-j,t]} \frac{2\eta_k^{L_m} L}{b}. \quad (8)$$

Algorithm 2 Deep-Q Learning Algorithm for node m with input $sample_t^m$ and $w_t^{L_m}$

1. Require: replay memory RM_m, action-value function Q_m with weights θ_t, target action-value function $\hat{Q}_m$ with weights $\theta^{'}$, the previous action of the node a_{t-1}^m, the previous loss f_{t-1}.
2. Given the training sample $sample_t^m$, set the current state $s_t = [w_t^{L_m}, f(w_t^{L_m}, sample_t^m)]$ and the previous state $s_{t-1} = [w_{t-1}^{L_m}, f(w_{t-1}^{L_m}, sample_{t-1}^m)]$. Set the reward as $r_{t-1} = -f(w_t^{L_m}, sample_{t-1}^m)$.
3. Store transition $(s_{t-1}, a_{t-1}^m, r_{t-1}, s_t)$ in RM_m.
4. Sample random mini-batch of transitions from RM_m.

Set $y_j = \begin{cases} r_j, \text{ if terminates at step } j+1. \\ r_j + \gamma_{DQ} \max_{a'} \hat{Q}_m(s_{j+1}, a', \theta^{'}), \text{otherwise,} \end{cases}$ (7)

in which γ_{DQ} is the discounting factor. Perform a gradient descent step on $(y_j - Q_m(s_j, a_j, \theta_t))$ w.r.t the network parameter θ_t. In addition, reset $\hat{Q}_m = Q_m$ every C steps.
5. With probability p_{explr} select a random action a_t^m, otherwise select $a_t^m = \arg max_a Q(s_t, a, \theta_t)$.
6. Feed a_t^m to Algorithm 1.

Proof. See Appendix A. □

Theorem 2. *Suppose that the loss function f is L-Lipschitz, convex and μ-smooth, if the noise term $\eta_t^{L_m} N_t$ is sampled according to (4), with $\Delta_2(q) = ||w_{t+1} - w'_{t+1}||$ which is given by Lemma 4, then Algorithm 1 is (ϵ, δ)-differentially private.*

Proof. See Appendix B. □

The following theorem shows the convergence rate of Algorithm 1 for convex loss function f.

Theorem 3. *Suppose that the hypothesis space $\mathcal{W}$ has diameter R, the loss function f is convex and L-Lipschitz on $\mathcal{W}$, and $||\nabla f_i(w)||^2 \leq B^2, \forall w, i$. Let $p_{t,L}^{L_m}$ and $p_{t,G}^{L_m}$ denote the probabilities (given by the Deep-Q learning algorithm) that node m chooses to update the local model and global model, respectively. Then for any $1 \leq T \leq \frac{n}{b}$, if we run Algorithm 1 for T iterations with step size $\eta_t^{L_m} = \eta$, we have*

$$\begin{aligned} &\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(p_{t,L}^{L_m} w_t^{L_m} + p_{t,G}^{L_m} w_t^G) - F(w^*)] \\ &\leq p_{t,L}^{L_m} \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(w_t^{L_m})] + p_{t,G}^{L_m} \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(w_t^G)] - F(w^*) \\ &\leq \frac{(M+1)R^2}{4T\eta} + \eta B^2 + \frac{4\ln(1.25/\delta)\eta L^2}{b^2\epsilon^2} \frac{\sum_{m'} |p_{t,G}^{L_{m'}}|}{T} \\ &+ \frac{2(2+12\sqrt{2})RL}{3} \left[\frac{\sqrt{bT}}{n} + \frac{2}{\sqrt{n} + \sqrt{n - bT}} \right], \end{aligned} \tag{9}$$

in which $F(\cdot) = \frac{1}{n}\sum_{i=1}^{n} f(\cdot)$ is the empirical risk, and $\sum_{m'} |p_{t,G}^{L_{m'}}|$ is the expected total number of time instances that the nodes update the global model.

Proof. Please see Appendix C in [21]. □

Remark 1. *By properly selecting the step size η (e.g., $\eta \propto \frac{1}{\sqrt{n}}$), the convergence rate is $\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(p_{t,L}^{L_m} w_t^{L_m} + p_{t,G}^{L_m} w_t^G) - F(w^*)] \leq \mathcal{O}(\frac{1}{\sqrt{n}})$. In addition, according to the definition of w^*, $\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(w_t^G)] - F(w^*) \geq 0$ and therefore $p_{t,L}^{L_m}[\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(w_t^{L_m})] - F(w^*)] \leq \mathcal{O}(\frac{1}{\sqrt{n}})$. As a result, there exists a positive constant $p_L^{min} \leq p_{t,L}^{L_m}, \forall t, m$ such that $\mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} F(w_t^{L_m})] - F(w^*) \leq \mathcal{O}(\frac{1}{p_L^{min}\sqrt{n}})$, which indicates the convergence of the local models.*

For the convergence rate of Algorithm 1 with λ-strongly convex loss function f, we add the following assumption.

Assumption 1. *At each time instance $0 \leq t \leq T$, each node updates once (either the local model or the global model).*

Theorem 4. *Suppose that the loss function f is γ-strongly convex and L-Lipschitz, and $||\nabla f_i(w)||^2 \leq B^2, \forall w, i$. For any $1 \leq T \leq \frac{n}{bM}$, if we run Algorithm 1 for T iterations with step size given by $\eta_t^{L_m} = \frac{1}{a\gamma t}, \forall m$, in which $a = \min\{p_{t,L}^{L_1}, p_{t,G}^{L_1}, \cdots, p_{t,L}^{L_M}, p_{t,G}^{L_M}\} > 0$, we have*

$$\begin{aligned} &\sum_{m=1}^{M} \mathbb{E}[||w_{t+1}^{L_m} - w^*||^2] + \mathbb{E}[||w_{t+1}^G - w^*||^2] \\ &\leq \mathcal{O}(\frac{MB^2}{a^2 t} + \frac{MB^2 \log t}{a^2 bt} + \frac{ML^2 \ln(\frac{1.25}{\delta})}{a^2 b^2 \epsilon^2 t}). \end{aligned} \tag{10}$$

Proof. Please see Appendix D in [21] □

Remark 2. *Note that the parameter a may depend on the exploration rate p_{explr} in the deep-Q learning algorithm, which is initialized to be large and then annealed down to a small constant (e.g., 0.1). In particular, since there is a probability of p_{explr} with which a node will randomly select an action, we have $\frac{p_{explr}}{2} \leq a \leq 1 - \frac{p_{explr}}{2}$.*

V. Simulation Results

This section presents simulation results to evaluate the effectiveness of the proposed algorithms. In particular, two widely used public datasets are considered: Covertype and MNIST. MNIST is a computer vision dataset which consists of 70,000 28×28 pixel images of handwritten digits from 0 to 9 while Covertype is a larger dataset with 581,012 data points and a dimension of 54. Without loss of generality, we reduce the data samples in MNIST to 50 dimensions with principal component analysis (PCA) [22] in our simulation. In addition, the data of both datasets are normalized and projected on the surface of the unit ball before training. For the Deep-Q network, we build a 3-layer fully connected deep neural network for each node and choose the parameters according to [15]. The input layer consists of $d + 2$ neurons, where d is the dimension of the training samples; the hidden layer consists of 128 neurons and the output layer consists of 2 neurons. The activation functions of all the three layers are linear and the weights are initialized by performing Xavier initialization in Tensorflow. The exploration rate p_{explr} is set to 1 in the beginning and then

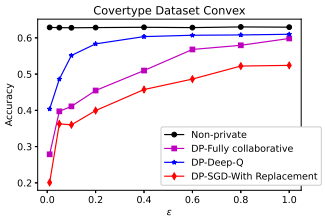

Fig. 1. Covertype Dataset (C)

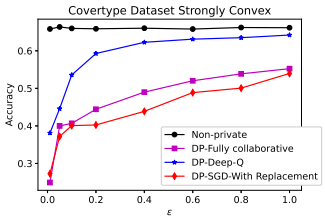

Fig. 2. Covertype Dataset (SC)

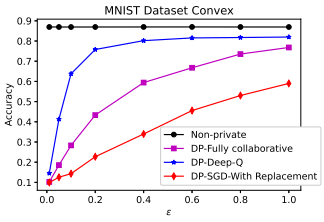

Fig. 3. MNIST Dataset (C)

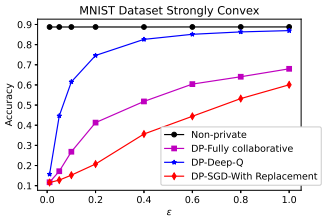

Fig. 4. MNIST Dataset (SC)

annealed down to 0.1 within $\frac{n}{2Mb}$ steps. The Adam optimizer is used to train the Deep-Q neural network with a learning rate of $\gamma_{DQ} = 0.01$; the mini-batch size and the size of the replay memory D_m are set to 10 and 20, respectively. In the simulation, the MNIST dataset is divided into a training subset of 60,000 samples and a testing subset of 10,000 samples while the Covertype dataset is divided into a training subset of 464,809 samples and a testing subset of 116,202 samples. Each node randomly draws $\frac{n}{M}$ samples from the training subset as its local training dataset. We build "one vs. all" multi-class logistic regression models for both datasets.[2] Then the nodes run the proposed algorithms (one pass over their local training dataset) to train the models, followed by the testing.

A. The Impact of Privacy Requirement

In this subsection, we investigate the impact of privacy requirement on the accuracy of the proposed algorithm. It is assumed that there are 10 collaborative nodes with 60,000 training samples for both datasets. The privacy parameter δ is set to $\frac{1}{n^2}$. We set $\eta_t = 0.1$ for the convex case. For the strongly convex case, the regularization parameter and the diameter of weights w is set to $\lambda = 0.0001$ and $R = 1/\lambda$, respectively. The mini-batch size is set to $b = 50$.

Figure 1 and Figure 2 show the classification accuracy of the proposed algorithm for the Covertype dataset in the convex and strongly convex scenarios, in which "C" and "SC" stand for "convex" and "strongly convex", respectively. More specifically, the simulation results of four scenarios are presented: the fully collaborative and noiseless case (denoted as "Noiseless"); the differentially private and fully collaborative case (i.e., the nodes update the global model with probability 1, denoted as "DP-Fully collaborative"); the differentially private and Deep-Q learning based algorithm (i.e., Algorithm 1, denoted as "DP-Deep-Q"); the baseline DP-SGD algorithm that adopts the i.i.d sampling strategy (denoted as "DP-SGD-With Replacement"). For "DP-SGD-With Replacement", we select a node to update a global model uniformly at random during each iteration. Each iteration is ensured to be $(\frac{\epsilon}{5}, \frac{\delta}{5})$ differentially private and the nodes stop updating the global model once their privacy budgets are depleted (i.e., the training samples have been visited 5 times). In addition, we use the same learning rate as that in [6] and set $\eta_t = \frac{1}{\sqrt{t}}$. It can be observed that Algorithm 1 outperforms both "DP-Fully collaborative" and "DP-SGD-With Replacement". While "DP-Fully collaborative" gives higher accuracy than "DP-SGD-With Replacement", another improvement of up to 10% in accuracy can be achieved by using the Deep-Q learning based algorithm in both convex and strongly convex scenarios. Similar results are observed on the MNIST dataset in Figure 3 and Figure 4.

[2]This means that 10 (7) binary models (one for each digit) are constructed and the output with the highest confidence is chosen as the prediction for the MNIST (Covertype) dataset.

TABLE I
THE ACCURACY OF PROPOSED ALGORITHMS FOR COVERTYPE DATASET

Number of nodes	1	5	10	20
Noiseless (C)	56.24%	61.90%	62.83%	64.10%
Fully Collaborative (C)	54.21%	58.63%	59.82%	60.22%
Algorithm 1 (C)	-	59.22%	60.65%	61.23%
Noiseless (SC)	62.79%	65.31%	65.96%	66.04%
Fully Collaborative (SC)	50.60%	52.60%	55.24%	57.73%
Algorithm 1 (SC)	-	59.27%	61.34%	62.37%

TABLE II
THE ACCURACY OF PROPOSED ALGORITHMS FOR MNIST DATASET

Number of nodes	1	5	10	20
Noiseless (C)	77.74%	85.49%	86.83%	87.69%
Fully Collaborative (C)	63.86%	74.04%	76.80%	78.17%
Algorithm 1 (C)	-	79.87%	80.52%	81.93%
Noiseless (SC)	84.80%	88.51%	88.76%	88.96%
Fully Collaborative (SC)	55.04%	65.63%	68.00%	73.9%
Algorithm 1 (SC)	-	75.39%	80.93%	82.97%

B. The Impact of the Number of Participating Nodes

In this subsection, we investigate the impact of the number of participating nodes. In particular, it is assumed that each node has 60,000 training samples for both datasets. Table I and Table II show the accuracy of the proposed algorithms in different scenarios for the Covertype dataset and the MNIST dataset with $\epsilon = 1$, respectively. It can be observed that as the number of participating nodes grows, the accuracy for both "DP-Fully collaborative" and Algorithm 1 increases since there are more training samples in total which can reduce the impact of the noise added at each iteration. In addition, Algorithm 1 is always better than "DP-Fully collaborative" and as the number of collaborative nodes grows, they are expected to approach the performance of the noiseless case. In the simulated scenarios, the accuracy degradation induced by privacy is within 6% and 4% for Algorithm 1 when there are 20 collaborative nodes for the MNIST dataset and the Covertype dataset, respectively.

VI. CONCLUSIONS AND FUTURE WORKS

In this work, the scenario in which multiple nodes (with limited training samples) collaboratively learn a global model

is studied. A decentralized differentially private without-replacement SGD algorithm is proposed, and both privacy and convergence analysis are provided. Extensive simulations are conducted to demonstrate the effectiveness of the proposed algorithm. Since we only consider the cases in which the objective functions are convex, differentially private non-convex optimization problems remain our future work.

APPENDIX A
PROOF OF LEMMA 4

Proof. Since node m updates the global model at time $t-j-1$, (5) can be written as follows:

$$w_{t+1}^G = \frac{w_t^G + w_{t-j}^{L_m} - \sum_{k=1}^{j} 2\eta_{t-k}^{L_m} \nabla f_{D_m(t-k)}(w_{t-k}^{L_m})}{2} - \eta_t^{L_m} \nabla f_{D_m(t)}(w_t^G) + N_t, \tag{11}$$

in which $D_m(t-k)$ is empty (i.e., $\nabla f_{D_m(t-k)}(w_{t-k}^{L_m}) = 0$) if node m does not update its local model at time $t-k$ either. Since $D_m(t-j:t)$ and $D'_m(t-j:t)$ differ at only the $i-th$ mini-batch, there are two possible cases.

case 1: ($i = t$) In this case, we have

$$||w_{t+1}^G - w_{t+1}^{G'}|| = \eta_t^{L_m} ||\nabla f_{D_m(t)}(w_t^G) - \nabla f_{D'_m(t)}(w_t^G)|| \leq \frac{2\eta_t^{L_m} L}{b}, \tag{12}$$

case 2: ($i \in [t-j, t)$) In this case,

$$||w_{t+1}^G - w_{t+1}^{G'}|| = \frac{1}{2} ||w_t^{L_m} - w_t^{L'_m}||, \tag{13}$$

in which $w_t^{L_m}$ and $w_t^{L'_m}$ are the local models of node m after j updates using the local mini-batches $D_m(t-j:t-1)$ and $D'_m(t-j:t-1)$, respectively. According to Lemma 1-3, when f_i's are convex, we have

$$||w_k^{L_m} - w_k^{L'_m}|| \leq \begin{cases} ||w_{k-1}^{L_m} - w_{k-1}^{L'_m}||, & \text{if } D_m(k-1) = D'_m(k-1). \\ ||w_{k-1}^{L_m} - w_{k-1}^{L'_m}|| + \frac{4\eta_{k-1}^{L_m} L}{b}, & \text{if } D_m(k-1) \neq D'_m(k-1). \end{cases}$$

As a result,

$$\frac{1}{2} ||w_t^{L_m} - w_t^{L'_m}|| \leq \max_{k \in [t-j,t)} \frac{2\eta_k^{L_m} L}{b}. \tag{14}$$

Combining (12) and (14), we have

$$||w_{t+1}^G - w_{t+1}^{G'}|| \leq \max_{k \in [t-j,t]} \frac{2\eta_k^{L_m} L}{b}. \tag{15}$$

□

APPENDIX B
PROOF OF THEOREM 2

Proof. Combing Lemma 4 and Theorem 1, it follows that each update step in Algorithm 1 is (ϵ, δ)-differentially private. Since each mini-batch is only visited once, Algorithm 1 is also (ϵ, δ)-differentially private over the whole dataset. □

REFERENCES

[1] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, "A guide to deep learning in healthcare," *Nature medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[2] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization." in *ICML*, 2012.

[3] H. Ren, J. Deng, and X. Xie, "Grnn: generative regression neural network—a data leakage attack for federated learning," *ACM TIST*, vol. 13, no. 4, pp. 1–24, 2022.

[4] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference*. Springer, 2006, pp. 265–284.

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

[6] I. Hegedus and M. Jelasity, "Distributed differentially private stochastic gradient descent: An empirical study," in *International Conference on Parallel, Distributed, and Network-Based Processing (PDP)*. IEEE, 2016, pp. 566–573.

[7] M. Gong, J. Feng, and Y. Xie, "Privacy-enhanced multi-party deep learning," *Neural Networks*, vol. 121, pp. 484–496, 2020.

[8] M. Kim, O. Günlü, and R. F. Schaefer, "Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication," in *IEEE (ICASSP)*, 2021, pp. 2650–2654.

[9] Z. Yu, J. Hu, G. Min, Z. Wang, W. Miao, and S. Li, "Privacy-preserving federated deep learning for cooperative hierarchical caching in fog computing," *IEEE Internet of Things Journal*, 2021.

[10] A. El Ouadrhiri and A. Abdelhadi, "Differential privacy for deep and federated learning: A survey," *IEEE Access*, vol. 10, pp. 22 359–22 380, 2022.

[11] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, no. Jan, pp. 165–202, 2012.

[12] O. Shamir, "Without-replacement sampling for stochastic gradient methods: Convergence results and application to distributed optimization," *arXiv preprint arXiv:1603.00570*, 2016.

[13] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, "Why random reshuffling beats stochastic gradient descent," *arXiv preprint arXiv:1510.08560*, 2015.

[14] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, and J. Naughton, "Bolt-on differential privacy for scalable stochastic gradient descent-based analytics," in *Proceedings of International Conference on Management of Data*. ACM, 2017, pp. 1307–1322.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[16] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," *arXiv preprint arXiv:1509.01240*, 2015.

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

[18] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[19] P. L. Ruvolo, I. Fasel, and J. R. Movellan, "Optimization on a budget: A reinforcement learning approach," in *NeurIPS*, 2009, pp. 1385–1392.

[20] C. Daniel, J. Taylor, and S. Nowozin, "Learning step size controllers for robust neural network training." in *AAAI*, 2016, pp. 1519–1525.

[21] R. Jin, X. He, and H. Dai, "Decentralized differentially private without-replacement stochastic gradient descent," *arXiv preprint arXiv:1809.02727*, 2023.

[22] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.