Topology-aware Differential Privacy for Decentralized Image Classification

Shangwei Guo, Tianwei Zhang, Guowen Xu, Han Yu, Tao Xiang, and Yang Liu

***Abstract*—Image classification is a fundamental artificial intelligence task that labels images into one of some predefined classes. However, training complex image classification models requires a large amount of computation resources and data in order to reach state-of-the-art performance. This demand drives the growth of distributed deep learning, where multiple agents cooperatively train global models with their individual datasets. Among such learning systems, decentralized learning is particularly attractive, as it can improve the efficiency and fault tolerance by eliminating the centralized parameter server, which could be the single point of failure or performance bottleneck.**

Although the agents do not need to disclose their training image samples, they exchange parameters with each other at each iteration, which can put them at the risk of data privacy leakage. Past works demonstrated the possibility of recovering training images from the exchanged parameters. One common defense direction is to adopt Differential Privacy (DP) to secure the optimization algorithms such as Stochastic Gradient Descent (SGD). Those DP-based methods mainly focus on standalone systems, or centralized distributed learning. How to enforce and optimize DP protection in decentralized learning systems is unknown and challenging, due to their complex communication topologies and distinct learning characteristics.

In this paper, we design TOP-DP, a novel solution to optimize the differential privacy protection of decentralized image classification systems. The key insight of our solution is to leverage the unique features of decentralized communication topologies to reduce the noise scale and improve the model usability. (1) We enhance the DP-SGD algorithm with this *topology-aware* noise reduction strategy, and integrate the time-aware noise decay technique. (2) We design two novel learning protocols (synchronous and asynchronous) to protect systems with different network connectivities and topologies. We formally analyze and prove the DP requirement of our proposed solutions. Experimental evaluations demonstrate that our solution achieves a better trade-off between usability and privacy than prior works. To the best of our knowledge, this is the first DP optimization work from the perspective of network topologies.

***Index Terms*—Decentralized Learning, Image Processing, Differential Privacy, Topology**

I. INTRODUCTION

DEEP Learning (DL) has become one of the most popular and powerful machine learning methods for the image classification task. To learn an accurate DL model, a common technique is Stochastic Gradient Descent (SGD), which iteratively approaches the ideal model by minimizing the empirical performance on a large number of training images. To accelerate the training process and protect data privacy, this SGD task can be distributed to multiple agents with their own training image sets to collaboratively learn a shared image classification model. This distributed learning [1], [2], [3] has gained a lot of popularity, especially in the edge computing [4], [5], [6].

Distributed learning can be divided into two categories: centralized and decentralized learning [7]. A centralized learning system utilizes a centralized parameter server to collect and aggregate estimates (i.e., model parameters) of agents at each iteration. In a decentralized system, agents interconnect based on a certain network topology and exchange estimates with their neighbors to reach consensus on the DL model. Distributed learning can prevent direct privacy leakage as each agent keeps its own private dataset locally at the training stage. However, it still faces the threats of indirect privacy leakage: the exchanged estimates among agents may contain information about their training sets. This gives honest-but-curious agents opportunities to compromise the data privacy of their neighbors. Past works have demonstrated the feasibility and severity of model inversion attacks [8], [9], [10], [11] and membership inference attacks [12], [13] in distributed learning.

To mitigate such privacy threats in distributed training, one promising solution is Differential Privacy (DP), which was originally introduced to preserve the privacy of individual data records in statistical databases [14]. A number of studies have then applied DP to SGD to enhance the privacy of DL training in different environments [15], [16], [17], [18], [19], [20]. Most existing DP-SGD algorithms adopt additive noise mechanisms by adding random noise to the estimates in every training iteration. There exists a trade-off between privacy and usability, determined by the noise scale added during training: adding too much noise can meet the privacy requirements, at the cost of huge drop in model accuracy. As a result, it is critical to identify the minimal amount of noise that can provide desired privacy protection, and also maintain acceptable model performance.

Two common approaches were devised to optimize the DP mechanism and balance the privacy-usability trade-off. The first one is to carefully restrict the sensitivity of randomized mechanisms. For example, Abadi et al. [16] bounded the influence of training samples on gradients by clipping each gradient in l_2 norm below a given threshold. Yu et al. [19] optimized the model accuracy by adding decay noise to the gradients over the training time since the learned models converge iteratively. The second approach is to precisely track the accumulated privacy cost of the training process using composition techniques such as the strong composition theorem [21] and moments account

T. Zhang is the corresponding author.

S. Guo and T. Xiang are with College of Computer Science, Chongqing University, Chongqing, China (email: {swguo, txiang}@cqu.edu.cn).

T. Zhang, G. Xu, H. Yu, and Y. Liu are with School of Computer Science and Engineering, Nanyang Technological University, Singapore (email: {tianwei.zhang, guowen.xu, han.yu, and yangliu}@ntu.edu.sg).

(MA) [16], [22], [23], [24], [25].

Those DP-SGD solutions have been well developed and evaluated in centralized learning systems. In contrast, privacy protection in the decentralized learning setting is less explored. There are distinct differences between these two systems. First, decentralized systems have more interactions and parameter exchanges in order to reach the consensus. Each agent receives parameters from multiple neighbors and broadcasts the update to them. Second, many decentralized systems are usually spontaneously organized and each agent is relatively independent. It is highly possible that certain nodes are offline due to the discrepancy of network bandwidth or unpredictable system faults. The asynchronous training mode [26], [27] thus becomes more prevalent with higher reliability and efficiency. Hence, we raise two questions: (1) *how can we design DP algorithms to support these unique features (e.g., decentralized topology, asynchronous training)?* (2) *How can we leverage these features to further optimize the DP solution and balance the privacy-usability trade-off?*

Prior works in differentially private decentralized learning mainly targeted the Alternating Direction Method of Multipliers (ADMM) algorithm with existing optimization techniques [28], [17], [18], [29]. They cannot be used with the mainstream SGD-based training tasks. Other differentially private decentralized leanring methods [18], [30], [31], [32], [33] are designed either using existing DP techniques or for a specific application. For example, [18] simply applied the standard DP technique (e.g., tracking accumulated privacy loss [21]) from the centralized setting to the decentralized one. These optmization techniques have been well studied, and seem to reach the performance limit. In contrast, the unique topology features were never considered.

In this paper, we present TOP-DP, a novel **Top**ology-aware **D**ifferential **P**rivacy approach for SGD-based training in decentralized systems. TOP-DP leverages network features of decentralized systems to optimize the randomized mechanism. The key idea is that each agent takes into account the injected noise from its neighbors when adding its own noise to the aggregated parameter. Such noise reuse can significantly reduce the actual noise scale added by each agent, but still satisfying the DP requirement. In addition, TOP-DP can also be integrated with the noise decay technique from the standalone training mode, to further optimize the DP protection in decentralized systems.

Based on this strategy, we design two new learning protocols to realize our optimization. The first one is for synchronous training mode. Different from existing styles, each agent calculates and sends different aggregated estimates to different neighbors. This can guarantee that each parameter exchange can always enjoy the maximal benefit from the topology-aware strategy. However, such advantage becomes minor when two connected agents share the same neighbors in a network topology. To copy with this corner case and save communication bandwidth, we introduce an asynchronous training protocol: at every iteration, each agent only pairs with one neighbor which is randomly picked to meet the noise reduction criterion. Then the parameter exchange between the pair can reduce the noise scale, and eliminate unnecessary communication costs in total.

We extensively validate the privacy and effectiveness of our proposed solution. From the theoretical view, we formally prove that each agent can guarantee differential privacy with significantly reduced noise. Empirically, we conduct comprehensive experiments to demonstrate that our solution outperforms prior works and techniques under various system configurations, datasets and DL models. We make the following contributions in this paper:

- To best of our knowledge, this is the first work that utilizes the network topology feature to enhance the usability of DP in distributed learning systems.
- We propose two novel learning protocols to achieve DP optimization for both synchronous and asynchronous training modes.
- We formally prove that our solution can guarantee the DP requirement for all the agents, and analyze its advantage under different decentralized settings.
- We conduct extensive experiments to show the superior of our method over prior works with various scenarios and image classification tasks.

The rest of this paper is organized as follows. Section III introduces formal definitions of decentralized systems, differential privacy and problem statement. Section IV presents the topology-aware and time-aware strategies for decentralized systems. Section V illustrates two learning protocols for synchronous and asynchronous training modes, respectively. Section VI presents our privacy analysis, and Section VII shows the experimental evaluations of our approach under various system settings. We review the related works in Section II, and conclude in Section VIII.

II. RELATED WORK

Differential privacy has been adopted to protect the individual privacy of training datasets and a large amount of DP-SGD algorithms have been proposed [20], [34]. We classify these algorithms into two categories: DP-SGD for standalone and distributed learning systems.

A. DP-SGD for Standalone Learning Systems

For standalone systems, there are commonly two possible ways to add random noise. The first one is to inject noise to the objective function. For instance, Chaudhuri et al. [15] perturbed the objective function before optimizing over classifiers and proved that the objective perturbation is DP if certain convexity and differentiability criteria hold. The sensitivity analysis methods in [15] relies on a strong convexity assumption. However, most objective function is non-convex. Phan et al. [35] attempted to use the objective perturbation by replacing the non-convex function with a convex polynomial function. To this end, a new convex polynomial function was introduced in [35] to approximate the non-convex one. However, this would change the learning protocol and, even worse, sacrifice the model's performance.

A simpler but more popular way is to add random noise to the gradients. Abadi et al. [16] achieved DP by adding

Gaussian noise to the gradients of each iteration. This approach restricts the sensitivity of randomized mechanisms, i.e., the influence of training data on gradients, by clipping each gradient in l_2 norm below a given threshold. Abadi et al. [16] also proposed MA to reduce the added noise by keeping track of a bound on the moments of the privacy loss during the training process. Yu et al. [19] focused on the DP problem during the sharing and publishing of pre-trained models. They optimized the model accuracy by adding decay noise to the gradients over the training time since the learned models converge iteratively. They improved the model usability by employing a generalization of concentrated DP, based on the observation that the privacy loss of an additive noise mechanism follows a sub-Gaussian distribution.

Another way to improve the model usability lies in precisely tracking the overall privacy cost of the training process. Shokri et al. [36] and Wei et al. [37] composed the additive noise mechanisms using the advanced composition theorem [21], leading to a linear increase in the privacy budget. In [16], [22], [23], [24], moments account (MA) was used to reduce the added noise by keeping track of a bound on the moments of the privacy loss during the training process. Other algorithms [38], [39], [19] were designed to improve the model usability using (zero) concentrated DP [40], based on the observation that the privacy loss of an additive noise mechanism follows a sub-Gaussian distribution. Recently, Asoode et al [41] proposed an optimal DP analysis to further reduce the scale of added noise during the training process.

B. DP-SGD for Centralized Learning Systems

Some works [36], [22], [23], [39], [24] applied the DP techniques from the standalone mode to the centralized learning systems to preserve the privacy of the training data for each agent. For example, Shokri et al. [36] proposed a privacy-preserving distributed learning algorithm by adding Laplacian noise to each agent's gradients to prevent indirect leakage. Kang et al. [24] adopted weighted aggregation instead of simply averaging to reduce the negative impact caused by uneven data scale.

In terms of the accumulated privacy loss, Kang et al. [24] employed MA to track the overall privacy cost of the training process. Wei et al. [37] perturbed agents' trained parameters locally by adding Gaussian noise before uploading them to the server for aggregation and bounded the sensitivity of the Gaussian mechanism by clipping. Shokri et al. [36] and Wei et al. [37] composed the additive noise mechanisms using the strong composition theorem [21], leading to a linear increase in the privacy budget.

Federated learning, as a typical example of centralized learning systems, has gained great popularity. A variety of works [42], [43], [44], [37], [45] have attempted to solve the privacy problem using the above DP techniques in such federated systems. For instance, Truex et al. [43] proposed a hybrid method that leverages both secure multiparty computation and differential privacy techniques to achieve privacy-preserving federated learning. Choudhury et al. [44] designed a federated learning system that enables to process sensitive health data with differential privacy protection. Wei et al. [45] provided user-level privacy protection for federated learning systems and improve the usability of trained models.

Some DP-SGD methods [45] for centralized learning systems, especially for federated learning systems, can be applied to decentralized systems as well. However, these methods are designed specifically for the corresponding centralized collaboration schemes, and do not well optimized for the decentralized setting. In contrast, our TOP-DP utilizes the unique features of decentralized systems and can significantly improve the usability of the trained models of all agents compared with state-of-the-art DP-SGD for federated learning.

C. DP-SGD for Decentralized Learning Systems

Several DP approaches [28], [17], [18], [29], [32], [31] were proposed for decentralized learning systems. However, they are mainly for the ADMM or gradient tracking algorithms, while there are very few solutions for the SGD algorithm in decentralized systems. Recently, motivated by the privacy leakage problem in big data analytics, Li et al. [18] proposed DP-SGD algorithms by adding Gaussian and Laplacian noise to the gradients. Zhou et al. [32] designed a differential privacy decentralized learning system for social recommendation systems. However, they track the accumulated privacy loss using the existing DP techniques such as the strong composition theorem, which limits the performance of learned models.

Similar as the centralized DP approaches, all those decentralized DP solutions (both for ADMM, SGD or other optimization algorithms) only apply existing DP techniques and focus on restricting the sensitivity of the optimization algorithm. Besides, they require the decentralized systems to be well synchronous. In contrast, our TOP-DP provides a novel optimization direction from the network topology. It can also be combined with other optimization algorithms such as ADMM and existing techniques (e.g., noise decay). The learning protocols in TOP-DP can be applied to different training modes efficiently.

III. BACKGROUND AND PROBLEM STATEMENT

In this section, we first formalize decentralized systems. Then, we present the threat model and the definition of DP for decentralized learning.

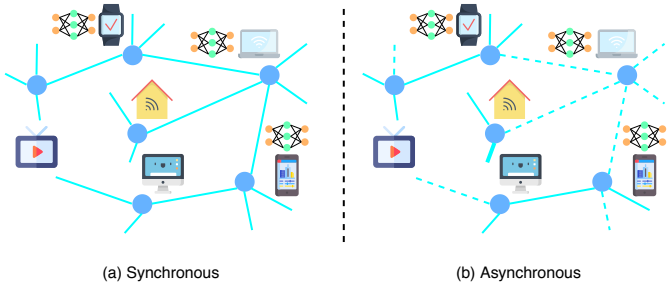
A. Decentralized Systems

We consider a decentralized system whose communication topology can be represented as an undirected graph: $\mathcal{G} = (V, E)$. V denotes a set of participates (or agents) in this decentralized network. E represents the set of communication links among the agents, with the following two properties:

1) $(i, j) \in E$ if and only if agent i can receive information from agent j;
2) $(j, i) \in E$ if $(i, j) \in E$.

We assume this undirected graph is fully connected, i.e., giving two arbitrary agents i and j, there always exists at least one path that connects them. This property can guarantee that information can be exchanged among all agents [46], [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/TCSVT.2021.3105723, IEEE Transactions on Circuits and Systems for Video Technology



(a) Synchronous (b) Asynchronous

Fig. 1. (a) Synchronous training and (b) asynchronous training in a decentralized learning system. Solid lines represent network connections with parameter exchanges between agents; dotted lines represent connections not used for parameter exchanges in certain iteration.

Let $x \in \mathbb{R}^d$ be the d-dimensional estimate vector of a DL model. Each agent $i \in V$ obtains a private training dataset D_i, consisting of independent and identically distributed (i.i.d.) data samples from a distribution D. Those agents train a shared model by solving the optimization problem [7], [1]:

$$\min_{x \in \mathbb{R}^d} \mathbb{E}_{\xi \sim D} l(x; \xi),$$

where ξ is a training data sample from D. During training, each agent i calculates its local estimate x_i, and exchanges x_i with its neighbors for parameter update. There are basically two training modes for this iterative process. In the *synchronous* mode (Fig. 1 (a)), each agent i needs to receive the estimates from all its neighbors before updating the model. In the *asynchronous* mode (Fig. 1 (b)), agent i exchanges parameters with only part of its neighbors for model update. This happens when the agent just wants to choose a smaller number of neighbors for lower communication and computation cost, or when some of its neighbors fail to respond due to unexpected system or network faults.

To adapt to both synchronous and asynchronous modes as well as maintaining the convergence rate, at each iteration, agent i (1) first collects estimates from its neighbor(s); (2) randomly selects a neighbor j^* from the participated neighbor(s); (3) utilizes the following update rule [26], [47] to aggregate estimates and calculate the local estimate:

$$x_i = \alpha x_i + (1-\alpha) x_{j^*} - \lambda g(x_i, \xi_i) \tag{1}$$

where $\alpha \in [0, 1]$ is a hyper-parameter determining the weight of the local estimate; λ is the learning rate; $g(x_i, \xi_i)$ is the stochastic gradient with $\xi_i \in D_i$. The gradient can also be replaced by a mini-batch of stochastic gradients [7], [26].

B. Threat Model

In a decentralized system, we assume the agents are *honest-but-curious*: all the agents agree on the proposed learning protocol and objective in advance. They will also strictly follow the steps of training and exchanging parameters during collaborative training. However, there exist some suspicious agents who attempt to passively steal the information and properties of their neighbors' datasets by analyzing the model parameters received at each iteration. We further assume that these suspicious agents will not collude to conduct the privacy attacks. Only connected neighbors are allowed to exchange information following the distributed training protocol.

For decentralized learning systems, agents are connected directly or indirectly. Our goal is to adopt DP to protect the training data privacy of all agents. DP is a rigorous mathematical framework to protect the privacy of individual records in a database when the aggregated information about this database is shared among untrusted parties [14]. Thus, we formally define decentralized learning with DP as follows:

Definition 1. *(DP of Decentralized Learning) A decentralized learning system is $\{(\epsilon_i, \delta_i)\}_{i \in V}$ differentially private if for each agent i, the randomized mechanism $\mathcal{M}_i : \mathcal{D}_i \to \mathcal{R}$ with domain $\mathcal{D}_i$ and range $\mathcal{R}$ satisfies (ϵ_i, δ_i)-DP, i.e., if for any two neighboring datasets D_i, D_i' and any subset of outputs $S \subseteq R$, the following property is held:*

$$Pr[\mathcal{M}(D_i) \in S] \leq e^{\epsilon_i} Pr[\mathcal{M}(D_i') \in S] + \delta_i. \tag{2}$$

$\mathcal{M}_i$ is restricted by two parameters: ϵ_i and δ_i. ϵ_i is the privacy budget of agent i to limit the privacy loss of training data. δ_i is a relaxation parameter that allows the privacy budget of $\mathcal{M}_i$ to exceed ϵ_i with probability δ_i. A decentralized learning system is differentially private if all agents are differentially private. Each agent can set its own privacy budget. Alternatively, the entire system can enforce a uniform privacy budget for all agents.

To achieve differentially private decentralized learning, a common and straightforward way is to use additive noise mechanisms at each iteration [20]. Specifically, we use Gaussian mechanism and denote σ_i as the noise parameter of agent i. At each iteration, agent i adds the Gaussian noise, $G_i = G(\sigma_i^2)$, to the updated local estimate to guarantee differential privacy (Eq. 3). Then, i sends $\widetilde{x}_i$ to its neighbors.

$$\widetilde{x}_i = \alpha \widetilde{x}_i + (1-\alpha) x_{j^*} - \lambda g(\widetilde{x}_i, \xi_i) + G_i. \tag{3}$$

IV. Optimization Strategies

As shown in Eq. 3, the random noise G_i added into the aggregated estimate must be large enough to satisfy the privacy requirement. However, adding too much noise can affect the model accuracy. So it is important to balance this trade-off. This section presents the strategies adopted in Top-DP to reduce the amount of noise for each agent to improve the usability of trained models, without violating the DP requirement. We start with a novel topology-aware noise reduction strategy. Then we extend time-aware noise decay to decentralized systems.

A. Strategy 1: Topology-aware Noise Reduction

Existing DP-SGD solutions all assume that the required noise scale only depends on the agents themselves. In decentralized systems, the communication topology can affect the amount of noise as well. Our topology-aware noise reduction strategy is able to reduce the noise scale of each agent when considering its connectivity with its neighbors. The key insight of our approach is that *the received estimates from other neighbors also contain certain noise, which can contribute*

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/TCSVT.2021.3105723, IEEE Transactions on Circuits and Systems for Video Technology

t

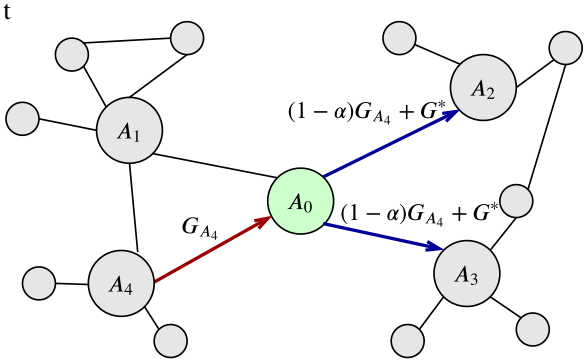


Fig. 2. An illustrative example of topology-aware noise reduction.

to the noise scale of the aggregated estimate, thus reducing the amount of noise added by the agent itself.

Fig. 2 gives an illustrative example. We consider an agent A_0 with four neighbors, where A_1 and A_4 are connected as well. When A_0 obtains all estimates of its neighbors, we assume it picks the estimate $\widetilde{x}_{A_4}$ of A_4 for aggregation with its own estimate and gradient. Since the received $\widetilde{x}_{A_4}$ also includes Gaussian noise G_{A_4}, then the aggregated estimate following Eq. 3 will have the corresponding random component $(1-\alpha)G_{A_4}$. As a result, when generating the estimate for agent A_2 or A_3, A_0 does not need to add the full-scale noise G_{A_0}. It only needs to inject the noise G^* such that

$$G_{A_0} = G^* + (1-\alpha)G_{A_4},$$

which can meet the DP requirement, but reduce the actual amount of noise.

It is worth noting that the noise scale G^* is not applicable when generating estimates for A_1 or A_4. For A_4, since it already knows its own parameter $\widetilde{x}_{A_4}$, then G_{A_4} is not random noise anymore. It is similar for A_1 as it receives $\widetilde{x}_{A_4}$ from A_4. Then for these two agents, we can pick another agent (e.g., A_2 or A_3) and generate a different estimate for them with still reduced noise scale. It is worth noting that our strategy allows the agent to send different estimates to different neighbors in one iteration, which is different from conventional distributed learning systems.

Formally, given an agent i, for each of its neighbors $j \in \mathcal{N}_i$, we define

$$\mathcal{N}_i^j = \mathcal{N}_i \backslash (j \bigcup \mathcal{N}_j),$$

which is the set of i's neighbors that are not connected to j (or j itself). For instance, in Fig. 2, we have $\mathcal{N}_{A_0}^{A_1} = \mathcal{N}_{A_0}^{A_4} = \{A_2, A_3\}$, $\mathcal{N}_{A_0}^{A_2} = \{A_1, A_3, A_4\}$ and $\mathcal{N}_{A_0}^{A_3} = \{A_1, A_2, A_4\}$. This also means that j can be used in the aggregation for all agents in $\mathcal{N}_i^j$ with the reduced noise scale. Then our goal is to find a minimal set $\widehat{\mathcal{N}_i}$, such that using the agents inside this set for aggregation can cover all the neighbor agents of i. Note that there can exist a neighbor j' that is connected to every neighbor in $\mathcal{N}_i$. Then we cannot find an non-adjacent neighbor to cover it, and should exclude it from $\mathcal{N}_i$. This process is described in Eq. 4. We will solve it heuristically in Section V-A.

$$\widehat{\mathcal{N}_i} = \underset{\mathcal{N} \subseteq \mathcal{N}_i}{\operatorname{argmin}} \Big(\bigcup_{j \in \mathcal{N}} \mathcal{N}_i^j = \mathcal{N}_i \backslash \big(\bigcup_{\mathcal{N}_i^{j'} = \emptyset} j' \big) \Big) \tag{4}$$

After identifying $\widehat{\mathcal{N}_i}$, for $\forall j \in \mathcal{N}_i$, if j is connected to every neighbor in $\mathcal{N}_i$ (i.e., $\mathcal{N}_i^j = \emptyset$), then agent i just sends the local estimate with full-scale noise to j. Otherwise, there exists at least one neighbor $k \in \widehat{\mathcal{N}_i}$ such that $j \in \mathcal{N}_i^k$. Then the noise scale from i to j, $G_i^j = G(\sigma_i^{j\,2})$, should satisfy Eq. 5(a) in order to guarantee the DP requirement against j, where G_i, G_k are the full-scale noise. According to the additivity of Gaussian distribution, we calculate the noise parameter σ_i^j via Eq. 5(b). With this reduced noise scale, agent i can update the estimate for agent j based on Eq. 5(c).

$$G_i = (1-\alpha)G_k + G_i^j \tag{5a}$$

$$\sigma_i^j = \sqrt{\sigma_i^2 - (1-\alpha)^2 \sigma_k^2} \tag{5b}$$

$$\widetilde{x}_i^j = \alpha \widetilde{x}_i + (1-\alpha)\widetilde{x}_k^i - \lambda g(\widetilde{x}_i, \xi_s) + G_i^j \tag{5c}$$

B. Strategy 2: Time-aware Noise Decay

Our topology-aware strategy can be combined with existing state-of-the-art techniques from other systems to enhance the optimization effects. We use the time-aware noise decay as an example. This technique was originally proposed in [19], to optimize the DP protection of model training in standalone systems. Here we apply this technique to decentralized systems. The key idea is that *the model converges and the norm of gradients decreases as the training iteration increases. Thus, the sensitivity of the Gaussian mechanism decreases, allowing us to inject less noise to the gradients*. Note that the training datasets are distributed in different agents, all agents in the decentralized system should reach a consensus on the noise decay schedule to tolerate the differences in the datasets.

Specifically, compared to the aggregation process in Eq. 5(c), our first modification is to clip the gradients in l_2 norm to bound their size at each training iteration. We follow the method from [16]: given a clipping threshold C, the clipped gradient vector $\bar{g}$ is bounded by C, as shown in Eq. 6(a).

Our second modification is to dynamically reduce the noise scale over the training time. Without loss of generality, we use step decay to reduce the noise scale every few epochs. Let $\sigma_{0,i}$ be the initial noise parameter of agent i. The noise parameter of agent i at the t-th iteration is shown in Eq. 6(b), where $\gamma \in (0,1)$ is the reduction factor and $period$ is the reduction step of noise decay.

$$\bar{g}(\widetilde{x}_i, \xi_s) = \frac{g(\widetilde{x}_i, \xi_s)}{\max(1, \frac{|g(\widetilde{x}_i, \xi_s)|}{C})} \tag{6a}$$

$$\sigma_{t,i} = \text{Decay}(\sigma_{0,i}, t) = \sigma_{0,i} \gamma^{\lfloor \frac{t}{period} \rfloor} \tag{6b}$$

V. Topology-aware Learning Protocols

With the topology-aware and time-aware strategies, we design two end-to-end decentralized learning protocols for synchronous and asynchronous modes, respectively.

A. Synchronous Topology-aware Protocol

In the synchronous mode, each agent requires the estimates from all its neighbors at each iteration. Then it solves Eq. 4 and calculates the parameters for different neighbors following Eqs. 5 and 6. We design a synchronous topology-aware protocol to support DL on decentralized topologies and Algorithm 1 illustrates the detailed communication and learning process of an agent i.

The algorithm takes as input the initial estimate x_0, initial noise parameter $\sigma_{0,i}$, learning rate λ, and number of iterations T. Before iteratively optimizing the shared model, agent i sends $\sigma_{0,i}$ to and receives $\{\sigma_{0,j}\}_{j\in\mathcal{N}_i}$ from its neighbors (Line 1). For $j \in \mathcal{N}_i$, agent i computes the neighbor set $\mathcal{N}_i^j$, including all neighbors that do not connect with j. Then, it updates the initial estimate and sends $\widetilde{x}_{0,i}$ to its neighbors (Lines 4-6). At the i-th iteration, it first computes the full-scale noise parameter $\sigma_{t,i}$ using the time-aware noise decay strategy (Line 8). Then, it computes the clipped gradient using a randomly selected sample ξ_s, generates estimates for all its neighbors and updates its local estimate using the proposed topology-aware noise reduction strategy.

To heuristically solve Eq. 4, agent i continuously selects an agent k from $\mathcal{N}_i$ exclusively until all agents are traversed or $\widehat{\mathcal{N}_i}$ is found. For $\forall j \in \mathcal{N}_i^k$, it computes the estimate $\widetilde{x}_i^j$ and sends it to j (Lines 10-17). The complexity of the approximate method is $O(|\mathcal{N}_i|d)$. Then, agent i randomly selects a neighbor $j^* \in \mathcal{N}_i$ and updates its local noised estimate (Lines 18-19). If there are still uncovered neighbors, i sends its local estimate to those neighbors (Lines 20-22). After T iterations, Algorithm 1 returns the final differentially private DL model.

Algorithm 1: Differentially private decentralized learning for agent i in the synchronous mode.

Input : Initial estimate x_0, initial budget $\sigma_{0,i}$, learning rate λ, number of iterations T

1 Send $\sigma_{0,i}$ to $\mathcal{N}_i$ and receive $\{\sigma_{0,j}\}_{j\in\mathcal{N}_i}$;
2 **foreach** $j \in \mathcal{N}_i$ **do**
3 $\quad \mathcal{N}_i^j \leftarrow \mathcal{N}_i \backslash (j \bigcup \mathcal{N}_j)$
4 $\bar{g}(x_0, \xi_s) \leftarrow$ Compute the clipped gradient;
5 $\widetilde{x}_i \leftarrow x_0 - \lambda \bar{g}(x_0, \xi_s) + G(\sigma_{0,i}^2 C^2)$;
6 Send $\widetilde{x}_i$ to its neighbors;
7 **for** $t \in [0, T)$ **do**
8 $\quad \sigma_{t,i} \leftarrow \texttt{Decay}(\sigma_{0,i}, t)$;
9 $\quad \bar{g}(\widetilde{x}_i, \xi_s) \leftarrow$ Compute the clipped gradient;
10 $\quad \mathcal{N}_i^* \leftarrow \mathcal{N}_i$ and $\mathcal{N}_i^\ddagger \leftarrow \mathcal{N}_i$;
11 $\quad$ **while** $\mathcal{N}_i^* \neq \emptyset$ *and* $\mathcal{N}_i^\ddagger \neq \emptyset$ **do**
12 $\quad\quad$ Randomly select $k \in \mathcal{N}_i^\ddagger$ and $\mathcal{N}_i^\ddagger \leftarrow \mathcal{N}_i^\ddagger \backslash k$;
13 $\quad\quad \sigma_{t,i}^j \leftarrow \sqrt{\sigma_{t,i}^2 - (1-\alpha)^2 \sigma_{t,k}^2}$, where $\sigma_{t,k} = \texttt{Decay}(\sigma_{0,k}, t)$;
14 $\quad\quad$ **foreach** $j \in \mathcal{N}_i^* \cap \mathcal{N}_i^k$ **do**
15 $\quad\quad\quad$ Update the estimate $\widetilde{x}_i^j \leftarrow \alpha\widetilde{x}_i + (1-\alpha)\widetilde{x}_k^i - \lambda\bar{g}(\widetilde{x}_i, \xi_s) + G({\sigma_{t,i}^j}^2 C^2)$;
16 $\quad\quad\quad$ Send $\widetilde{x}_i^j$ to agent j;
17 $\quad\quad \mathcal{N}_i^* \leftarrow \mathcal{N}_i^* \backslash \mathcal{N}_i^k$;
18 $\quad$ Randomly select an agent j^* from $\mathcal{N}_i$;
19 $\quad$ Update the local estimate $\widetilde{x}_i \leftarrow \alpha\widetilde{x}_i + (1-\alpha)\widetilde{x}_{j^*}^i - \lambda\bar{g}(\widetilde{x}_i, \xi_s) + G(\sigma_{t,i}^2 C^2)$;
20 $\quad$ **if** $\mathcal{N}_i^* \neq \emptyset$ **then**
21 $\quad\quad$ **foreach** $j \in \mathcal{N}_i^*$ **do**
22 $\quad\quad\quad$ $\widetilde{x}_i^j \leftarrow \widetilde{x}_i$ and send $\widetilde{x}_i^j$ to agent j ;
23 **return** $\widetilde{x}_i$

B. Asynchronous Topology-aware Protocol

Although the synchronous training in Algorithm 1 can realize the proposed strategies to improve the model usability, it still leaves some spaces for further optimization. First, each agent only selects part of the received parameters for update while discarding the rest. So it is not necessary to collect the estimates from all the neighbors, which can cause extra communication cost and waiting latency. Second, as introduced in Section IV-A, when agent j connects to every neighbor of agent i, i has to add full-scale noise to the parameter sent to j. The topology-aware optmization will lose effectiveness when there are a lot of such (i, j) pairs.

To overcome the above limitations, we design a novel topology-aware protocol for asynchronous training. At every iteration, each agent only pairs with one of its neighbors for parameter exchange and update. An extra checking is conducted to guarantee that the paired agents are qualified for the topology-aware noise reduction: two agents cannot be paired twice in two consecutive iterations. Otherwise, the aggregated parameter selected by one agent in the previous iteration is not a secret to the other agent, and full-scale noise has to be added in this iteration. Hence the topology-aware noise reduction cannot be applied. Specifically, during the training process, agent i randomly selects a neighbor j which is different from the paired neighbor in the previous iteration. Then i asks j's availability for parameter sharing. If j agrees to collaborate with i in this iteration, they exchange parameters with the reduced noise scale and update the models following Eq. 5. If agent i cannot find a qualified or available pair at this iteration, it will update its estimate by itself.

Algorithm 2 describes the detailed steps of our asynchronous learning protocol. Similar to Algorithm 1, it takes the same parameters as input, and updates the initial estimate (Lines 1-3). At the t-th iteration, agent i passively waits for pairing request from other neighbors. Meanwhile, it also actively searches in a random order for a neighbor that is not paired with it in the previous iteration (Lines 7-16). If the selected agent j is available or i receives a pairing request from j', i stops searching and pairs with $j_i^{t+1} = j$ (j'). Agent i sends $\widetilde{x}_i$ to and receives $\widetilde{x}_{j_i^{t+1}}$ from j_i^{t+1} (Line 18). Then, i adopts time-aware and topology-aware strategies to reduce the noise scale (Lines 19-20) and updates its estimates (Lines 21-22). Otherwise, i only utilizes the time-aware noise decay to update its estimate locally (Lines 24-26).

VI. Theoretical Analysis

We perform a formal analysis about Algorithms 1 and 2 from the aspects of privacy and efficiency.

Algorithm 2: Differentially private decentralized learning for agent i in the asynchronous mode.

Input : Initial estimate x_0, initial budget $\sigma_{0,i}$, learning rate λ, number of iterations T

Send $\sigma_{0,i}$ to $\mathcal{N}_i$ and receive $\{\sigma_{0,j}\}_{j\in\mathcal{N}_i}$;
$\bar{g}(x_0,\xi_s) \leftarrow$ Compute the clipped gradient ;
$\widetilde{x}_i \leftarrow x_0 - \lambda\bar{g}(x_0,\xi_s) + G(\sigma_{0,i}^2 C^2)$;
$j_i^0 \leftarrow None$;
for $t \in [0,T)$ **do**
  $\sigma_{t,i} \leftarrow \texttt{Decay}(\sigma_{0,i}, t)$;
  $\mathcal{N}^* \leftarrow \mathcal{N}_i / j_i^t$;
  while $\mathcal{N}^* \neq \emptyset$ **do**
    Randomly select $j \in \mathcal{N}^*$ and $\mathcal{N}^* \leftarrow \mathcal{N}^* \backslash j$;
    Ask if j is available for pairing up;
    if *j is available* **then**
      $j_i^{t+1} \leftarrow j$;
      $\mathcal{N}^* \leftarrow \emptyset$;
    if *receive pairing request from j'* **then**
      $j_i^{t+1} \leftarrow j'$;
      $\mathcal{N}^* \leftarrow \emptyset$;
  if j_i^{t+1} *is found* **then**
    Send $\widetilde{x}_i$ and receive $\widetilde{x}_{j_i^{t+1}}$ to/from j_i^{t+1} ;
    $\sigma_{t,j_i^{t+1}} \leftarrow \texttt{Decay}(\sigma_{0,j_i^{t+1}}, t)$;
    $\sigma_{t,i} \leftarrow \sqrt{\sigma_{t,i}^2 - (1-\alpha)^2\sigma_{t,j_i^{t+1}}^2}$;
    $\bar{g}(\widetilde{x}_i,\xi_s) \leftarrow$ Compute the clipped gradient;
    Update the local estimate $\widetilde{x}_i \leftarrow \alpha\widetilde{x}_i + (1-\alpha)\widetilde{x}^i_{j_i^{t+1}} - \lambda\bar{g}(\widetilde{x}_i,\xi_s) + G(\sigma_{t,i}^2C^2)$;
  else
    $\sigma_{t,i} \leftarrow \texttt{Decay}(\sigma_{0,i}, t)$;
    $\bar{g}(\widetilde{x}_i,\xi_s) \leftarrow$ Compute the clipped gradient;
    Update the local estimate $\widetilde{x}_i \leftarrow \alpha\widetilde{x}_i + (1-\alpha)\widetilde{x}^i_{j_i^{t+1}} - \lambda\bar{g}(\widetilde{x}_i,\xi_s) + G(\sigma_{t,i}^2C^2)$;
return $\widetilde{x}_i$

A. Proof of DP

First, we prove Algorithm 1 is differentially private by carefully choosing the initial noise parameters. We track the accumulated privacy loss of the training process using Rényi DP [48], which is a natural relaxation of DP based on the Rényi divergence and allows tighter analysis of tracking cumulative privacy loss and ensures a sublinear loss of privacy as a function of the number of iterations.

Theorem 1. *Let the number of iterations be T. For any decentralized system $\mathcal{G}$ and every agent $i \in V$, the randomized mechanisms in Algorithm 1 is (ϵ_i, δ_i)-DP if we choose*

$$\sigma_{0,i} \geq \frac{8\sqrt{T\log\frac{1}{\delta_i}\log\frac{1.25}{\delta_i}}}{\epsilon_i|D_i|} \tag{7}$$

Proof. We prove the theroem in the synchronous mode and ignore the time-aware noise decay strategy since it does not incur any additional privacy loss [19]. We clip the gradients in l_2 norm of C and assume the privacy budget ϵ_i' is the same at each iteration. According to the Gaussian mechanism [14], the update rule in Line 19 is (ϵ_i', δ_i)-DP at one iteration if we choose

$$\sigma_{0,i} \geq \frac{\sqrt{2\log\frac{1.25}{\delta_i}}}{\epsilon_i'|D_i|}.$$

Using Rényi composition theorem [48], our new update rule is (ϵ_i, δ_i)-DP after T iterations if we choose

$$\epsilon_i = 4\epsilon_i'\sqrt{2T\log\frac{1}{\delta_i}}.$$

Then, we have

$$\epsilon_i' = \frac{\epsilon_i}{4\sqrt{2T\log\frac{1}{\delta_i}}}.$$

Combining the above equations, we conclude that our update rule in Line 19 is (ϵ_i, δ_i)-DP if we choose $\sigma_{0,i}$ such that

$$\sigma_{0,i} \geq \frac{8\sqrt{T\log\frac{1}{\delta_i}\log\frac{1.25}{\delta_i}}}{\epsilon_i|D_i|} \tag{8}$$

We have proven that the local estimate of agent i is differentially private during the training process. Then, we prove that for $\forall j \in \mathcal{N}_i$, the estimates generated for j is also differentially private. Let $k, (k,j) \notin E$ be the selected agent for generating estimate for j. Since j, k are not directly connected, the noise of $\widetilde{x}_k^i$ can be used as a random component to guarantee the DP of i against j. Thus, because all agents generate noise independently, the noise scale for j should satisfy

$$G(\sigma_{0,i}) = G(\sigma_i^j) + (1-\alpha)G_{0,k} \tag{9}$$

According to the additivity of Gaussian distribution, the noise parameter for the estimate for j is

$$\sigma_i^j = \sqrt{\sigma_{0,i}^2 - (1-\alpha)^2\sigma_{0,k}^2}.$$

Therefore, in Algorithm 1, the estimates generated for the neighbors of agent i are also differentially private. □

The DP of Algorithm 2 can also be analyzed in a similar way. Note that an agent cannot pair with another agent twice in a row. Therefore, even the agents in a decentralized system are fully connected, the topology-aware noise reduction still works in such situation, where Algorithm 1 fails.

B. Efficiency Analysis of TOP-DP

Our protocols can reduce the noise and thus improve the usability of the trained models using the proposed TOP-DP algorithm when considering the communication topology. Here, we theoretically analyze the efficiency of TOP-DP by comparing the amount of added noise with and without TOP-DP. Without loss of generality, we assume

$$\sigma = \sigma_{t,i} = \sigma_{t,j} \text{ for } \forall i,j \in V \text{ and } (i,j) \in E.$$

Let $\sigma_{t,i}^j$ be the noise parameter of $\widetilde{x}_i^j$ at iteration t. According to the proposed topology-aware noise reduction strategy,

$$\begin{aligned}\sigma_{t,i}^j &= \sqrt{\sigma_{t,i}^2 - (1-\alpha)^2\sigma_{t,k}^2} \\ &= \sigma\sqrt{2\alpha - \alpha^2}.\end{aligned} \tag{10}$$

Compared with the full-scale noise parameter, the noise added to $\widetilde{x}_i^j$ is reduced by a factor of $\sqrt{2\alpha - \alpha^2}$. We can observe that $\sigma_{t,i}^j$ decreases as $\alpha \in (0, 1)$ decreases. When α approaches 0, the noise of the estimates that agent i sends to/receives from its neighbors would be significantly reduced. Thus, the usability of the trained models would be theoretically improved because of the decrease of the added noise.

In synchronous mode (Algorithm 1), agent i can always reduce the noise of the estimates for its neighbor j using the TOP-DP if there exists an agent that connects to agent i and cannot communicate with j directly, i.e.,

$$\mathcal{N}_i / \mathcal{N}_j \neq \emptyset.$$

For the asynchronous settings (Algorithm 2), TOP-DP works if it finds a pairing neighbor during the iteration. Therefore, the agents in both synchronous and asynchronous modes can theoretically improve the utility of their trained models using our TOP-DP.

VII. EXPERIMENTS

A. Implementation and Experimental Setup

Dataset and DNN model. We conduct experiments mainly on the MNIST dataset. It consists of a training set of 60k samples and a test set of 10k samples. We consider a fully connected network with a hidden layer of size 100 for image classification. We set a fading learning rate λ with the initial value of 0.05. Our solution is general and can be applied to other DNN tasks as well (as demonstrated in Section VII-E).

For the implementation of the decentralized system, we consider a network consisting of 30 agents, and each agent connects to others with the probability of 0.2 (connection rate). This decentralized system is guaranteed to be fully connected, i.e., there exists at least one path connecting two arbitrary agents. The training set of each agent is independent and identically distributed with the same size. In the synchronous mode, all 30 agents participate at each training iteration. In the asynchronous mode, we assume 10% of random agents will not be involved at each iteration.

Without loss of generality, the agents have same privacy budget (1.0) and relaxation hyper-parameter (10^{-5}). We assume the agents reach the consensus on the time-aware noise decay strategy, where γ and $period$ are 0.9 and 1000, respectively. We clip the gradients in l_2 norm of 4.0.

Baselines and metrics. We consider different decentralized learning algorithms in our experiments:

- *No Noise*: the agents exchange parameters without DP protection.
- *Li18*: the DP-SGD algorithm proposed by Li et al. [18].
- *Li18+MA*: we integrate Li18 with moments account [16] to track the accumulated privacy loss.
- *UDP*: the user-level DP-SGD algorithm proposed by Wei et al. [45].
- *Optimal*: the optimal DP analysis for SGD proposed by Asoodeh et al. [41].
- *Proposed*: our proposed learning protocols.

It is worth noting the first five solutions cannot be applied to the asynchronous mode directly. For fair comparisons, we modify their update rules as Eq. 3 to follow our learning protocol for asynchronous learning. For each algorithm, we measure the testing accuracy of each agent's model at every iteration during the training, and report the average accuracy.

B. Effectiveness of TOP-DP

We evaluate and compare the performance of those DP-SGD algorithms under different settings in both synchronous and asynchronous modes.

Epoch v.s. accuracy. Fig. 3 illustrates the trend of average testing accuracy in the training process with different α values. First, we observe that our proposed algorithm outperforms all baselines, and is closer to the No Noise case, for different α values and modes. Such advantage is more obvious with a smaller α, as the reduced noise is larger. Second, Li18+MA has higher performance than Li18 because of the usage of MA. With the new DP technique, Optimal outperforms UDP and Li18+MA in most settings. Different from our solution, the usability of the models from all baselines significantly decreases as α decreases. This is caused by the increase of the noise of the selected estimates. Third, the model training in synchronous mode converges slightly faster than the one in the asynchronous mode, since each agent can contribute to the model training to accelerate the process.

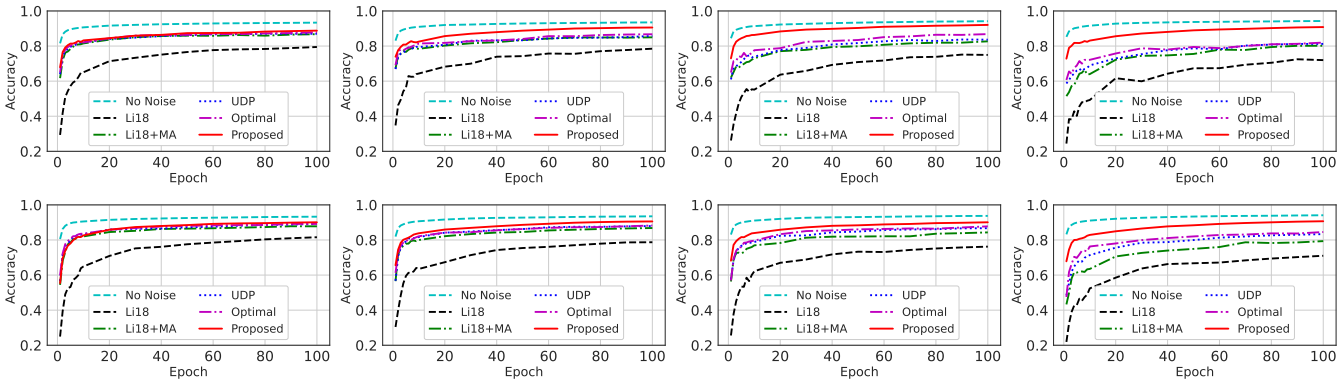
Privacy budget v.s. accuracy. We consider the impact of privacy budget on the model accuracy, as shown in Fig. 4. We can observe our solution can beat the other DP solutions for different privacy budgets. Besides, when the privacy budget decreases, the model usability decreases, as more noise is required to inject to the estimates. Meanwhile, the advantage of our solution also increases, as the amount of reduced noise increases as well. This indicates that our algorithm is more effective when a small privacy budget is needed.

C. Impact of System Configurations

Connection rate. We set the connection rate of the decentralized network as 0.2 in the previous experiments. Our proposed algorithm is effective under other connection rates as well. To validate this, we measure and compare the performance of different DP-SGD algorithms with the connection rates of 0.1 and 0.4. Without loss of generality, we consider the synchronous mode and set α as 0.25. Figure 5 shows the average accuracy of the agents as the training epoch increases. We observe that the performance of each algorithm does not change with different connection rates. The underlying reason may be that although the number of an agent's neighbors is changed with the connection rate, the agent still selects one estimate for updates at each iteration. Then the training result will not be changed either. As such, our proposed solution can exhibit advantages over prior works under various network connection rates.

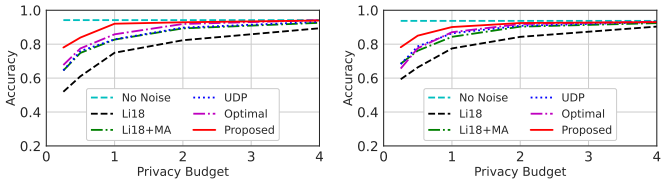
Number of agents. We now investigate the impact of the number of agents on the performance of decentralized learning systems. We conduct experiments on decentralized systems

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/TCSVT.2021.3105723, IEEE Transactions on Circuits and Systems for Video Technology



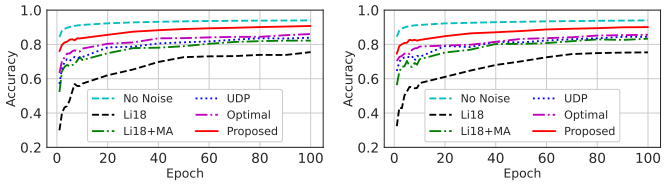
(a) $\alpha = 0.75$ (b) $\alpha = 0.5$ (c) $\alpha = 0.25$ (d) $\alpha = 0.125$

Fig. 3. The average accuracy of the agents with different α values under synchronous (first row) and asynchronous (second row) settings.



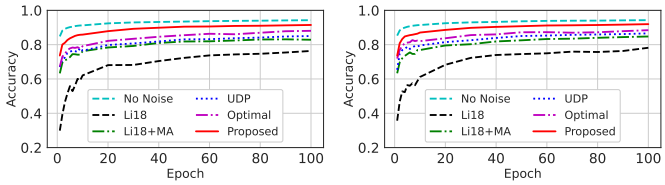
(a) Synchronous (b) Asynchronous

Fig. 4. The average accuracy of the agents as the privacy budget increases.



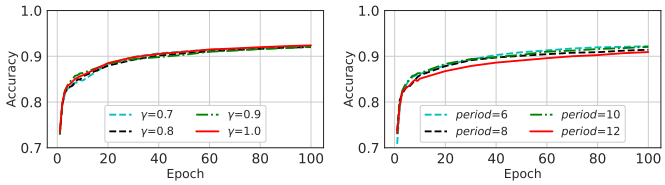
(a) Connection Rate = 0.1 (b) Connection Rate = 0.4

Fig. 5. The average accuracy of the agents with different connection rates under the synchronous setting.



(a) Number of Agents = 40 (b) Number of Agents = 50

Fig. 6. The average accuracy of the agents with different numbers of agents under the synchronous setting.



(a) Impact of γ (b) Impact of $period$

Fig. 7. The average accuracy of the agents with different parameters of the noise decay strategy under the synchronous setting.

with 40 and 50 agents in the synchronous mode. The experimental results are shown in Figure 6. We observe that the accuracy of the trained models only slightly increases with more agents involved, indicating that the number of agents has a small positive impact on the decentralized systems.

Parameters of the noise decay strategy. We evaluate the impact of the parameters of the noise decay strategy on our TOP-DP, i.e., γ and $period$. In our experiments, γ is set from 0.7 to 1.0 while $period$ varies from 8 to 12. Figure 7 illustrates the average performance of the trained models under different parameter settings. Two observations are drawn. First, both γ and $period$ have only limited impact on the performance of the decentralized learning systems, especially in Figure 7 (a). Second, as $period$ increases, the average accuracy of the trained models slightly decreases.

Network topology. We also evaluate our DP-SGD learning protocols on other typical network structures, such as the ring, star, tree, and mesh topologies. Figure 8 illustrates the comparisons of decentralized network structures with different connections. In each network topology, we set the number of total agents as 30. Figure 9 shows the learning curves of different DP-SGD algorithms for both synchronous and asynchronous modes.

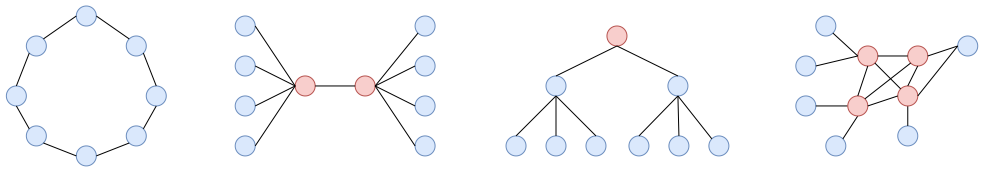
We observe that in the synchronous mode, the average accuracy scores of our learning protocals are significantly higher than other baselines, which is attributed to our proposed topology-aware strategy. The performance gap among Li18+MA, UDP, and Optimal is small due to the limitation of the corresponding DP optimizations. In the asynchronous mode, our protocol is slightly better than others, although the advantage is not as big as the synchronous mode.

D. Effectiveness of Each Strategy

Our DP-SGD learning protocols are composed of two strategies: topology-aware noise reduction (TOP) and noise-aware noise decay (ND). We evaluate the integration of these two strategies in the above experiments. In this section, we measure the effectiveness of TOP separately. Figures 10 and 11 illustrate the performance comparison between TOP, the integration TOP+ND, and other DP-SGD algorithms.

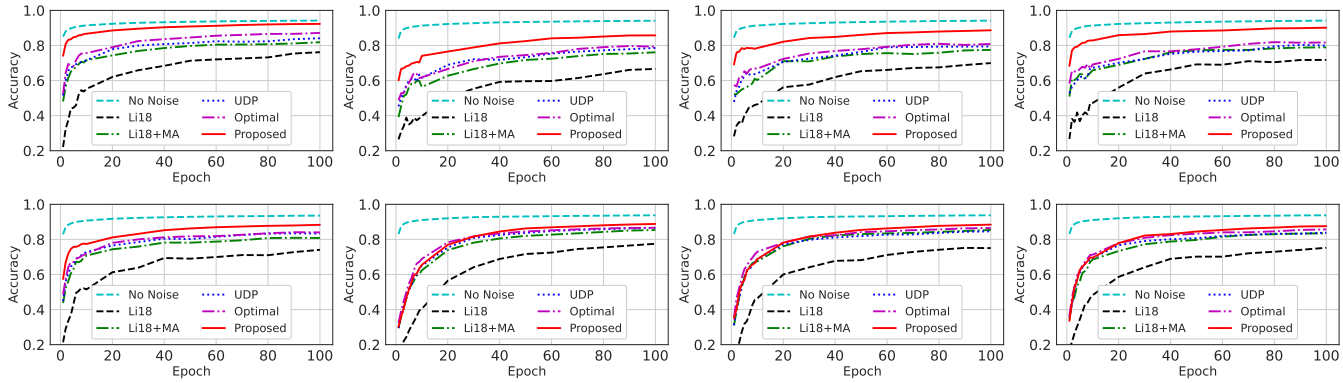
We observe that in the synchronous mode, TOP almost has the same performance as TOP+ND at the first 20 epochs, as the reduced noise from ND strategy is quite small at the first two

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/TCSVT.2021.3105723, IEEE Transactions on Circuits and Systems for Video Technology



(a) (b) (c) (d)

Fig. 8. Four types of decentralized topologies. (a) Ring topology; (b) Star topology with two star agents; (c) Tree topology; (d) Partial mesh topology.



(a) Ring (a) Star (b) Tree (c) Mesh

Fig. 9. The average accuracy of the agents with different network topologies under both synchronous (first row) and asynchronous (second row) settings.

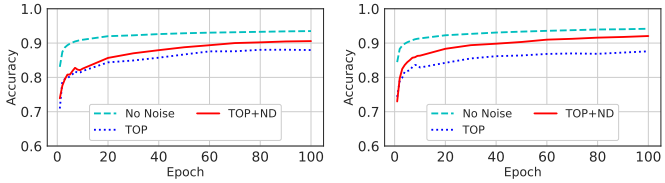
reduction steps (the noise is not reduced at the first reduction step). With more epochs, TOP+ND is slightly better than TOP only, caused by the effectiveness of ND. In the asynchronous mode, TOP almost has the same performance as TOP+ND especially when α equals 0.25.

E. Results of a More Complicated Dataset

We also evaluate TOP-DP on a more complicated training task over CIFAR10 dataset. The model to be trained is a Convolutional Neural Network, consisting of two max–pooling layers and three fully connected layers. The system settings and configurations are the same as the ones on MNIST. We set α and the connection rate as 0.25 and 0.2.

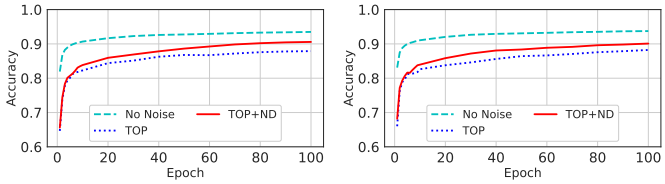
Figure 12 illustrates the experimental results in the synchronous and asynchronous modes. We observe that our solution (Proposed) outperforms prior DP-SGD algorithms and approaches the baseline (No Noise) as the training epoch increases in both of the two modes. The other four baselines even do not converge in the presence of Gaussian noise. The reason is that each parameter in the model needs to be appended with random noise to satisfy DP requirement. When the model becomes more complicated with more parameters, the overall divergence between the original model and the DP-protected model becomes larger, making it hard to converge. This scenario will never happen in our solution.

Our TOP-DP is designed to be general for various learning tasks and datasets. In terms of the computational complexity, the protocols require each agent to calculate the scale of noise that is added to its estimates at each iteration. The cost of calculating the noise scale is a constant, while the calculation of adding noise to estimates is proportional to the number of parameters, which is negligible compared to the training overhead. So we believe our solution is practical and scalable to higher-dimensional datasets and more complex neural networks. As future work, we will evaluate TOP-DP on larger-scale decentralized learning tasks.



(a) $\alpha = 0.5$ (b) $\alpha = 0.25$

Fig. 10. The effectiveness of topology-aware noise reduction with different α values under synchronous settings.

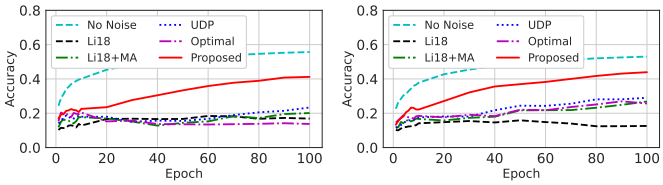


(a) $\alpha = 0.5$ (b) $\alpha = 0.25$

Fig. 11. The effectiveness of topology-aware noise reduction with different α values under asynchronous settings.

VIII. CONCLUSION

In this paper, we propose TOP-DP, a novel DP-based method to preserve the privacy of decentralized learning systems. The topology-aware technique leverages the network topology to reduce the noise scale and improve model usability while still satisfying the DP requirement. We apply the time-aware noise decay technique to the decentralized systems to



(a) Synchronous (b) Asynchronous

Fig. 12. The average accuracy of the agents in different modes on CIFAR10.

further optimize the model performance. We design learning protocols, which enables the topology-aware technique and adapts to both the synchronous and asynchronous learning modes. To the best of our knowledge, this is the first study to utilize network topology for DP optimization, and deploy DP protection to asynchronous decentralized systems. Formal analysis and empirical evaluations indicate that TOP-DP can guarantee the privacy requirement, and achieve better trade-offs between privacy and usability under different system configurations.

ACKNOWLEDGEMENTS

This research was supported by the National Research Foundation, Singapore, under its National Cybersecurity R&D Programme (NCR Award Number NRF2018NCR-NCR009-0001 and NRF2018NCR-NCR005-0001) and the AI Singapore Programme (AISG2-RP-2020-019); Singapore Ministry of Education AcRF Tier 1 RS02/19 and 2018-T1-002-069; the Singapore National Research Foundation under NCR Award Number NRF2018NCR-NSOE003-0001, NRF Investigatorship NRFI06-2020-0022; the Joint NTU-WeBank Research Centre on Fintech (NWJ-2020-008); the Nanyang Assistant Professorships (NAP); the RIE 2020 Advanced Manufacturing and Engineering Programmatic Fund (A20G8b0102), Singapore; and the SDU-NTU Centre for AI Research (C-FAIR); the National Natural Science Foundation of China under Grants 62072062 and U20A20176, and the Natural Science Foundation of Chongqing, China, under Grant cstc2019jcyjjqX0026. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of the funding agencies.

REFERENCES

[1] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[2] 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.

[3] A. Abou-Elailah, F. Dufaux, J. Farah, M. Cagnazzo, and B. Pesquet-Popescu, "Fusion of global and local motion estimation for distributed video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 158–172, 2012.

[4] C. Zhao and A. Basu, "Dynamic deep pixel distribution learning for background subtraction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4192–4206, 2019.

[5] J. Yang, L. Qing, W. Zeng, and X. He, "High-order statistical modeling based on a decision tree for distributed video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 5, pp. 1488–1502, 2018.

[6] T. Chen and S. Lu, "Robust vehicle detection and viewpoint estimation with soft discriminative mixture model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 2, pp. 394–403, 2015.

[7] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? A case study for decentralized parallel stochastic gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 5330–5340.

[8] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: information leakage from collaborative deep learning," in *ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 603–618.

[9] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 747–14 756.

[10] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *Annual Computer Security Applications Conference*, 2019, pp. 148–162.

[11] Y. Zhu, X. Yu, M. Chandraker, and Y.-X. Wang, "Private-kNN: Practical differential privacy for computer vision," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 854–11 862.

[12] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *IEEE Symposium on Security and Privacy*, 2019, pp. 691–706.

[13] K. Leino and M. Fredrikson, "Stolen memories: Leveraging model memorization for calibrated white-box membership inference," in *USENIX Security Symposium*, 2020, pp. 1605–1622.

[14] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2006, pp. 486–503.

[15] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *Journal of Machine Learning Research*, vol. 12, no. 3, 2011.

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

[17] X. Zhang, M. M. Khalili, and M. Liu, "Improving the privacy and accuracy of ADMM-based distributed algorithms," in *International Conference on Machine Learning*, 2018.

[18] C. Li, P. Zhou, L. Xiong, Q. Wang, and T. Wang, "Differentially private distributed online learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 8, pp. 1440–1453, 2018.

[19] L. Yu, L. Liu, C. Pu, M. E. Gursoy, and S. Truex, "Differentially private model publishing for deep learning," in *IEEE Symposium on Security and Privacy*, 2019, pp. 332–349.

[20] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *USENIX Security Symposium*, 2019, pp. 1895–1912.

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

[22] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," *arXiv preprint arXiv:1812.00984*, 2018.

[23] N. Hynes, R. Cheng, and D. Song, "Efficient deep learning on multi-source private data," *arXiv preprint arXiv:1807.06689*, 2018.

[24] Y. Kang, Y. Liu, and W. Wang, "Weighted distributed differential privacy ERM: Convex and non-convex," *arXiv preprint arXiv:1910.10308*, 2019.

[25] M. Gong, K. Pan, Y. Xie, A. K. Qin, and Z. Tang, "Preserving differential privacy in deep neural networks with relevance-based adaptive noise imposition," *Neural Networks*, vol. 125, pp. 131–141, 2020.

[26] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *International Conference on Machine Learning*, 2018, pp. 3043–3052.

[27] Q. Luo, J. He, Y. Zhuo, and X. Qian, "Heterogeneity-aware asynchronous decentralized training," *arXiv preprint arXiv:1909.08029*, 2019.

[28] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 172–187, 2016.

[29] J. Ding, Y. Gong, C. Zhang, M. Pan, and Z. Han, "Optimal differentially private admm for distributed machine learning," *arXiv preprint arXiv:1901.02094*, 2019.

[30] Y. Lou, L. Yu, S. Wang, and P. Yi, "Privacy preservation in distributed subgradient optimization algorithms," *IEEE Transactions on Cybernetics*, vol. 48, no. 7, pp. 2154–2165, 2017.

[31] Q. Lü, X. Liao, T. Xiang, H. Li, and T. Huang, "Privacy masking stochastic subgradient-push algorithm for distributed online optimization," *IEEE transactions on cybernetics*, vol. 51, no. 6, pp. 3224–3237, 2020.

[32] P. Zhou, K. Wang, L. Guo, S. Gong, and B. Zheng, "A privacy-preserving distributed contextual federated online learning framework with big data support in social recommender systems," *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[33] M. Hou, D. Li, X. Wu, and X. Shen, "Differential privacy of online distributed optimization under adversarial nodes," in *Chinese Control Conference*, 2019, pp. 2172–2177.

[34] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," in *USENIX Security Symposium*, 2021.

[35] N. Phan, Y. Wang, X. Wu, and D. Dou, "Differential privacy preservation for deep auto-encoders: An application of human behavior prediction," in *AAAI Conference on Artificial Intelligence*, 2016, pp. 1309–1316.

[36] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *ACM SIGSAC Conference on Computer and Communications Security*, 2015, pp. 1310–1321.

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

[38] M. Park, J. Foulds, K. Choudhary, and M. Welling, "DP-EM: Differentially private expectation maximization," in *Artificial Intelligence and Statistics*, 2017, pp. 896–904.

[39] B. Jayaraman, L. Wang, D. Evans, and Q. Gu, "Distributed learning without distress: Privacy-preserving empirical risk minimization," in *Advances in Neural Information Processing Systems*, 2018, pp. 6343–6354.

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

[41] S. Asoodeh, J. Liao, F. P. Calmon, O. Kosut, and L. Sankar, "Three variants of differential privacy: Lossless conversion and applications," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 208–222, 2021.

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

[43] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 1–11.

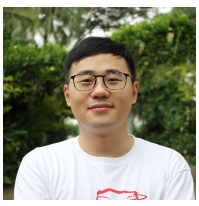
[44] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, I. Sylla, Y. Park, G. Hsu, and A. Das, "Differential privacy-enabled federated learning for sensitive health data," *arXiv preprint arXiv:1910.02578*, 2019.

[45] K. Wei, J. Li, M. Ding, C. Ma, H. Su, B. Zhang, and H. V. Poor, "User-level privacy-preserving federated learning: Analysis and performance optimization," *IEEE Transactions on Mobile Computing*, 2021.

[46] J. N. Tsitsiklis, "Problems in decentralized decision making and computation." Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, Tech. Rep., 1984.

[47] P. Blanchard, R. Guerraoui, J. Stainer *et al.*, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, 2017, pp. 119–129.

[48] I. Mironov, "Rényi differential privacy," in *IEEE Computer Security Foundations Symposium*, 2017, pp. 263–275.



Shangwei Guo is an associate professor in College of Computer Science, Chongqing University. He received the Ph.D. degree in computer science from Chongqing University, Chongqing, China at 2017. He worked as a postdoctoral research fellow at Hong Kong Baptist University and Nanyang Technological University from 2018 to 2020. His research interests include secure deep learning, secure cloud/edge computing, and database security.

Tianwei Zhang is an assistant professor in School of Computer Science and Engineering, at Nanyang Technological University. His research focuses on computer system security. He is particularly interested in security threats and defenses in machine learning systems, autonomous systems, computer architecture and distributed systems. He received his Bachelor's degree at Peking University in 2011, and the Ph.D degree in at Princeton University in 2017.

Guowen Xu is currently a Research Fellow with Nanyang Technological University, Singapore. He received the PhD degree in cyberspace security from the University of Electronic Science and Technology of China (UESTC) in 2020. As the first author, he has published more than 20 papers in top international conferences and journals, including ACM CCS, ACM ACSAC, ACM ASIACCS, IEEE TDSC and IEEE TIFS. He is the recipient of the Best Paper Award of the 26th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2020) , and the IEEE INFOCOM Student Conference Award. His research interests include Secure Outsourcing Computing and privacy-preserving issues in Deep Learning.

Han Yu received the BEng (hons) degree and PhD degree from the School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), Singapore, in 2007 and 2014, respectively. He is currently a Nanyang assistant professor (NAP) at SCSE, Nanyang Technological University, Singapore. From 2015 to 2018, he held the prestigious Lee Kuan Yew post-doctoral Fellowship (LKY PDF) at the Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY). His research focuses on the ethics of artificial intelligence and federated learning. He co-authored the book "Federated Learning" - the first monograph on the topic of federated learning.

Tao Xiang received the BEng, MS and PhD degrees in computer science from Chongqing University, China, in 2003, 2005, and 2008, respectively. He is currently a Professor of the College of Computer Science at Chongqing University. Prof. Xiang's research interests include multimedia security, cloud security, data privacy and cryptography. He has published over 100 papers on international journals and conferences. He also served as a referee for numerous international journals and conferences.

Yang Liu received the B.Comp. degree (Hons.) from the National University of Singapore (NUS) in 2005 and the Ph.D. degree from NUS and MIT, in 2010. He started his postdoctoral work in NUS and MIT. In 2012, he joined Nanyang Technological University (NTU). He is currently a Full Professor and the Director of the Cybersecurity Laboratory, NTU. He specializes in software verification, security, and software engineering. His research has bridged the gap between the theory and practical usage of formal methods and program analysis to evaluate the design and implementation of software for high assurance and security. By now, he has more than 270 publications in top tier conferences and journals. He received a number of prestigious awards, including the MSRA Fellowship, the TRF Fellowship, the Nanyang Assistant Professor, the Tan Chin Tuan Fellowship, the Nanyang Research Award, and eight best paper awards in top conferences, such as ASE, FSE, and ICSE.