SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

April 2024, Vol. 67, Iss. 4, 142303:1–142303:17
https://doi.org/10.1007/s11432-023-3918-9

Gradient sparsification for efficient wireless federated learning with differential privacy

Kang WEI[1], Jun LI[1*], Chuan MA[2,7], Ming DING[4], Feng SHU[3,1], Haitao ZHAO[5], Wen CHEN[6] & Hongbo ZHU[5]

[1]*School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing* 210096*, China;*
[2]*Zhejiang Lab, Hangzhou* 311121*, China;*
[3]*School of Information and Communication Engineering, Hainan University, Haikou* 570228*, China;*
[4]*Data61, Commonwealth Scientific and Industrial Research Organisation, Sydney* 2015*, Australia;*
[5]*School of Communications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing* 210003*, China;*
[6]*School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[7]*Key Laboratory of Computer Network and Information Integration* (*Southeast University*)*, Ministry of Education, Nanjing* 211189*, China*

Received 10 August 2023/Revised 12 October 2023/Accepted 18 December 2023/Published online 26 March 2024

Abstract Federated learning (FL) enables distributed clients to collaboratively train a machine learning model without sharing raw data with each other. However, it suffers from the leakage of private information from uploading models. In addition, as the model size grows, the training latency increases due to the limited transmission bandwidth and model performance degradation while using differential privacy (DP) protection. In this paper, we propose a gradient sparsification empowered FL framework with DP over wireless channels, to improve training efficiency without sacrificing convergence performance. Specifically, we first design a random sparsification algorithm to retain a fraction of the gradient elements in each client's local model, thereby mitigating the performance degradation induced by DP and reducing the number of transmission parameters over wireless channels. Then, we analyze the convergence bound of the proposed algorithm, by modeling a non-convex FL problem. Next, we formulate a time-sequential stochastic optimization problem for minimizing the developed convergence bound, under the constraints of transmit power, the average transmitting delay, as well as the client's DP requirement. Utilizing the Lyapunov drift-plus-penalty framework, we develop an analytical solution to the optimization problem. Extensive experiments have been implemented on three real-life datasets to demonstrate the effectiveness of our proposed algorithm. We show that our proposed algorithms can fully exploit the interworking between communication and computation to outperform the baselines, i.e., random scheduling, round robin, and delay-minimization algorithms.

Keywords federated learning, differential privacy, gradient sparsification, Lyapunov drift, convergence analysis

1 Introduction

Federated learning (FL) has emerged as a new distributed learning paradigm that enables multiple clients to collaboratively train a shared model without sharing their local data [1,2]. However, FL faces several critical challenges, that is, the storage, computational, and communication capabilities equipped at each client may differ due to variability in the computation frequency, memory, limited bandwidth, and power constraints. In light of this, the well-known federated average (FedAvg) algorithm with local stochastic gradient descent (SGD) and partial participation of clients is widely adopted to reduce the training latency and communication overhead [3]. Furthermore, several improved FL algorithms over wireless channels have been proposed to lower the inter-client variance caused by data heterogeneity and device heterogeneity [4].

Recently, to improve the training efficiency, some studies proposed effective scheduling algorithms for wireless and computation resources between clients and the edge server in FL [5–8]. The work in [5]

* Corresponding author (email: jun.li@njust.edu.cn)

characterized the performance of FL in wireless networks and developed an analytical model on FL convergence rate to evaluate the effectiveness of three different client scheduling schemes, i.e., random scheduling, round robin, and proportional fair. A control algorithm was proposed to minimize the loss function of the convergence bound of distributed SGD by formulating FL training over a wireless network as an optimization problem [6]. Further, the work in [7] improved the training efficiency based on the convergence bound by optimizing the client selection and power allocation via constructing the connection between the wireless resource allocation and the FL training performance using the convergence bound. The work in [8] formulated the FL training and communication problem as an optimization problem to minimize the total energy consumption of the system under a latency constraint. To address this problem, the work in [8] provided an iterative algorithm and closed-form solutions at every step for time allocation, bandwidth allocation, power control, computation frequency, and learning accuracy.

Different from effective scheduling algorithms for wireless and computation resources [9], model/gradient compression techniques such as sparsification [10] and quantization [11] can be an alternative method to balance training performance and transmission delay. The work in [12] introduced the finite-precision quantization in uplink and downlink communications, and provided new convergence analysis of the well-known FedAvg in the non-independent and identically distributed (non-IID) data setting and partial clients participation. Theoretical results revealed that, with a certain quantization, transmitting the weight differential can achieve a faster convergence rate, compared with transmitting the weight. The work in [13] analyzed the FL convergence in terms of quantization errors and the transmission outage, and then concluded that its performance can be improved if the clients have uniform outage probabilities. The work in [14] involved model pruning for wireless FL and formulated an optimization problem to maximize the convergence rate under the given learning latency constraint. These studies can reduce the transmission overhead efficiently by compressing the uploading models, but not consider the possible privacy leakage caused by exchanged learning models. Thus, it is desirable to utilize some privacy computing techniques to protect the shared learning models with the rigorous privacy guarantee [15], such as differential privacy (DP) [16].

Differentially private FL provides a rigorous privacy guarantee for clients' train data, in which there exists an inherent tradeoff between model performance and data privacy [17]. This is due to the implementation of the DP definition, which requires bounding the influence of each example on the gradient from SGD training or local models, and then perturbing it by random noises with a certain scale [17]. Especially for larger networks, they suffer from far greater distortions during training compared to their non-private counterparts, which results in significant penalties to utility [18]. To alleviate such degradation, one approach for the client is to leverage the gradient sparsification technique, which can keep the stochastic gradients stay in a low dimensional subspace during training [19]. However, existing studies for DP-based FL systems have not jointly considered the advantage of the gradient sparsification from both communication efficiency and DP training aspects, and designed an efficient framework to improve the training performance and communication efficiency [20].

To fill this gap, we propose a novel differentially-private FL scheme in wireless networks, termed differentially private FL with gradient sparsification (DP-SparFL), to provide a low communication overhead while maintaining a high model accuracy under the required privacy guarantee. The main contributions of this paper can be summarized as follows.

- We propose a gradient sparsification empowered communication-efficient FL system with the DP guarantee, which will randomly reduce the elements in the gradients of each local training. We also improve the DP-based FL performance using an adaptive gradient clipping technique with configurable gradient sparsification rates. Our proposed algorithm can efficiently reduce the detrimental influence of DP on the training and communication overhead in wireless networks.
- To further improve the training efficiency, we analyze the convergence bound in terms of gradient sparsification rates in the non-convex FL setting, which is more general than the convex problems. We formulate a novel stochastic optimization problem that minimizes the newly found convergence bound, while satisfying transmit power, average delay, and client's DP requirement constraints. Using the Lyapunov drift-plus-penalty framework, we provide an analytical and feasible solution to the problem.
- Extensive experimental results, for three classification tasks, including MNIST, Fashion-MNIST, and CIFAR-10 datasets, have been provided to demonstrate the effectiveness of our proposed algorithm. Moreover, we show that the proposed algorithm outperforms the baselines, i.e., random scheduling, round robin, and delay-minimization algorithms, with the DP requirement.

The rest of this paper is organized as follows. We introduce background on the FL framework and the

concept of DP in Section 2. In Section 3, we illustrate the system model of wireless FL and propose the DP-SparFL algorithm. Then, we analyze the convergence bound in terms of the gradient sparsification rate, and formulate the joint channel assignment problem as an optimization problem with feasible solutions in Section 5. Experimental results are described in Section 6. Finally, conclusion is drawn in Section 7.

2 Preliminaries

In this section, we will present preliminaries and related background knowledge on FL, DP, and the gradient sparsification technique.

2.1 Federated learning

We consider a wireless FL system where an access point (AP) with a central server and N available channels is located in the center of a wireless network and U clients are randomly distributed within the coverage of the AP. To train a machine learning (ML) model, AP needs to fully utilize all clients' data during T communication rounds. Let $\mathcal{D}_i$ denote the local dataset held by the i-th client, where $i \in \mathcal{U}$ and $\mathcal{U} = \{1, 2, \ldots, U\}$ is the index set of clients. The overall training process of such a wireless FL system can be divided into four parts as follows: (1) AP transmits the global model $\boldsymbol{w}^t \in \mathbb{R}^K$ and training information (e.g., channel assignment) to selected clients at the t-th communication round; (2) the selected clients perform local training to update the local models based on their local datasets; (3) the selected clients upload their local models/updates to AP; (4) AP aggregates all received local models/updates to generate a new global model. The aggregation process at the t-th communication round in the central server can be expressed as

$$\Delta \boldsymbol{w}^t = \sum_{i \in \mathcal{S}^t} p_i^t \Delta \boldsymbol{w}_i^t, \tag{1}$$

where $\mathcal{S}^t$ is the set of selected clients ($\mathcal{S}^t \subseteq \mathcal{U}$) at the t-th communication round, $p_i^t = |\mathcal{D}_i| / \sum_{i \in \mathcal{S}^t} |\mathcal{D}_i|$, $|\mathcal{D}_i|$ is the size of $\mathcal{D}_i$, $\Delta \boldsymbol{w}^t$ is the global update at the t-th communication round, $\Delta \boldsymbol{w}_i^t$ is the local update of the i-th client, given by $\Delta \boldsymbol{w}_i^t = \boldsymbol{w}_i^{t-1,\tau} - \boldsymbol{w}^{t-1}$, $\boldsymbol{w}_i^{t-1,\tau}$ is the local model after τ local training epochs, and $\boldsymbol{w}^{t-1}$ is the global model at the $(t-1)$-th communication round. Overall, we can formulate the FL task as

$$\boldsymbol{w}^\star = \arg\min_{\boldsymbol{w}} F(\boldsymbol{w}), \tag{2}$$

where $F(\boldsymbol{w}) = \sum_{i \in \mathcal{U}} p_i F_i(\boldsymbol{w})$ represents the global loss function, $F_i(\cdot)$ is the local loss function of the i-th client, and $p_i = |\mathcal{D}_i| / \sum_{i \in \mathcal{U}} |\mathcal{D}_i|$. We can observe that all clients have the same data structure and learn a global ML model collaboratively in the training procedure.

2.2 Differential privacy

DP mechanism has drawn a lot of attention because it can provide a strong criterion for the privacy preservation of distributed learning systems with parameters ϵ and δ. Research in differentially private ML models tracks a relaxed variant of DP, known as Rényi DP (RDP) [21] that has already been widely adopted, such as Opacus in Facebook and tensorflow privacy in Google [22]. We first define the neighborhood dataset $\mathcal{D}'$ as adding or removing one record in the dataset $\mathcal{D}$. Thus, we will adopt the RDP technique for the privacy budget computation, and then define RDP as follows.

Definition 1 ((α, ϵ)-RDP)**.** Given a real number $\alpha \in (1, +\infty)$ and privacy level (PL) ϵ, a randomized mechanism $\mathcal{M}$ satisfies (α, ϵ)-RDP for any two adjacent datasets $\mathcal{D}, \mathcal{D}' \in \mathcal{X}$, the Rényi distance between $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$ is given by

$$D_\alpha[\mathcal{M}(\mathcal{D}) \| \mathcal{M}(\mathcal{D}')] := \frac{1}{\alpha - 1} \log \mathbb{E}\left[\left(\frac{\mathcal{M}(\mathcal{D})}{\mathcal{M}(\mathcal{D}')}\right)^\alpha\right] \leqslant \epsilon, \tag{3}$$

where the expectation is taken over the output of $\mathcal{M}(\mathcal{D})$.

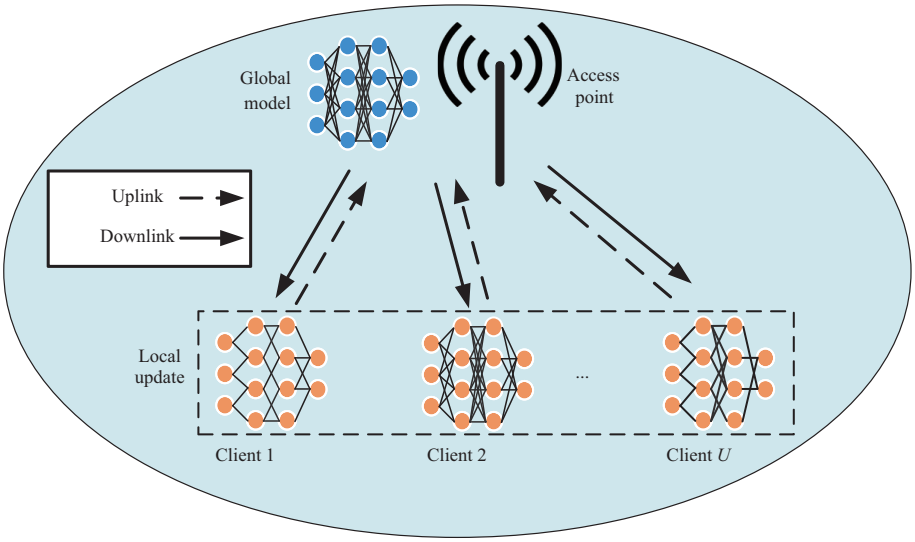

Figure 1 (Color online) Framework of wireless FL consists of multiple clients and an AP with multiple channels, where each client will train a learning model and upload the local update at each communication round with a DP constraint.

We can note that RDP is a generalization of (ϵ, δ)-DP that adopts Rényi divergence as a distance metric between two distributions. It can be shown that if a model $\mathcal{M}$ satisfies (α, ρ)-RDP, then $\mathcal{M}$ also satisfies $(\epsilon + \log \frac{1}{\delta}/(\alpha - 1), \delta)$-DP for any $\delta \in (0, 1)$. ML models achieve RDP guarantees by two alterations to the training process: clipping the per-sample gradient and adding Gaussian noise to training gradients, as known as DP-based SGD (DP-SGD). In our model, we assume that the server is curious-but-honest and may infer private information of clients by analyzing their local models/updates. The i-th client needs to achieve the (ϵ_i, δ)-DP requirement with a proper Gaussian noise standard deviation (STD) $\widehat{\sigma}$ with DP-SGD training, where ϵ_i is the required PL by the i-th client.

2.3 Gradient sparsification

Gradient sparsification [23] has been widely applied to reduce the model size as well as relieve the high communication burden over wireless channels. In the gradient sparsification method, for a given gradient sparsification rate (a certain percentage of elements that have been retained) s and a model gradient vector $\boldsymbol{g}$, this method will generate a binary mask $\boldsymbol{m} = [m_1, \ldots, m_K]$ with the same size with $\boldsymbol{g}$. When using the random sparsifier, the element in $\boldsymbol{m}$ is set to 1 with the probability s, and 0 with the probability $(1 - s)$, respectively. In each training process, the gradient update rule with gradient sparsification can be expressed as

$$g_k = \begin{cases} g_k, & \text{if } m_k = 1; \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where g_k is the k-th element in the gradient vector $\boldsymbol{g}$, $k \in \{1, \ldots, K\}$. It can be noted that when the k-th element of $\boldsymbol{m}$ is zero, the element in the same position of gradient does not need to be updated. Because $\boldsymbol{m}$ is a binary vector, the total number of bits to transmit the sparse local update to the server can be expressed as $32s|\boldsymbol{g}| + |\boldsymbol{g}|$, where $|\boldsymbol{g}|$ is the size of $\boldsymbol{g}$ and a 32-bit representation is adopted here. Hence, when s is small, using the sparsification method for each client in FL can efficiently reduce the communication overhead.

3 System model

In this subsection, we first present the system model and problem formulation of the client resource allocation supporting FL with an orthogonal frequency division multiple access (OFDMA) technique. We assume downlink communication for AP will adopt the broadcast module, i.e., the same channel for all clients. Due to the power constraint for each client device and the limited number of allocated channels to each scheduled device, we consider that not all clients will upload their updates at the aggregation step. The system model is shown as Figure 1.

3.1 Communication model

At the beginning of each communication round, the AP broadcasts the global model to selected clients. Hence, the downlink data rate and the transmission latency at the t-th communication round for the i-th client ($i \in \mathcal{U}$ and $\mathcal{U} = \{1, \ldots, U\}$) can be calculated as $C_i^{t,\mathrm{do}} = B \log_2(1 + \frac{P^t h_i^{t,\mathrm{do}}}{I_i^{t,\mathrm{do}} + \sigma^2})$ and $d_i^{t,\mathrm{do}} = \frac{Z^t}{C_i^{t,\mathrm{do}}}$, respectively, where P^t is the transmit power of the server, $h_i^{t,\mathrm{do}}$ is the average channel gain for the downlink channel, σ^2 is the noise power spectral density, $I_i^{t,\mathrm{do}}$ is the interference caused by other wireless equipments, Z^t is defined as the number of bits that the AP requires to transmit vector $\Delta \boldsymbol{w}^t$ over wireless links, and B denotes the bandwidth of the uplink channel. The uplink data rate of the i-th client transmitting its local model via the j-th channel ($j \in \mathcal{N}$ and $\mathcal{N} = \{1, \ldots, N\}$) to the AP at the t-th communication round can be expressed as $C_{i,j}^{t,\mathrm{up}} = B \log_2 (1 + \frac{P_i^t h_{i,j}^t}{I_{i,j}^{t,\mathrm{up}} + \sigma^2})$, where P_i^t is the transmit power of the i-th client, $h_{i,j}^t$ is the average channel gain, and $I_{i,j}^{t,\mathrm{up}}$ is the co-channel interference caused by the wireless equipment that is located in other service areas. The transmission delay and energy consumption between the i-th client and the j-th channel over the uplink channel at the t-th communication round can be computed as $d_{i,j}^{t,\mathrm{up}} = \frac{Z_i^t}{C_{i,j}^{t,\mathrm{up}}}$ and $E_{i,j}^{t,\mathrm{co}} = P_i^t d_{i,j}^{t,\mathrm{up}}$, respectively, where Z_i^t is defined as the number of bits to transmit the vector $\Delta \boldsymbol{w}_i^t$ over wireless links.

3.2 Computation model

In the computation model, each client is equipped with a CPU for the training task, in which the CPU frequency f_i^t (in CPU cycle/s) of the i-th client is changed at different communication rounds. Moreover, the number of CPU cycles performing the forward-backward propagation process for one data of the i-th client can be defined as Φ_i. Due to the fact that the CPU operates in the serial mode, the latency of local training can be expressed as $d_i^{t,\mathrm{lo}} = \tau |\mathcal{D}_i| \Phi_i / f_i^t$, where τ is the number of local training epochs [24–26]. Then, the CPU energy consumption of the i-th client for one local round of computation is expressed as $E_i^{t,\mathrm{cp}} = \chi_i \tau |\mathcal{D}_i| \Phi_i \left(f_i^t\right)^2 / 2$, where χ_i represents the effective capacitance coefficient of the i-th client's computing chipset [24–26].

3.3 Channel allocation

Due to the limited channels, we need to allocate the channel resources for clients, i.e., client-channel matching, at each communication round. We define $\boldsymbol{a}^t$ as a client scheduling matrix and its element $a_{i,j}^t \in \{0, 1\}, i \in \mathcal{U}, j \in \mathcal{N}$ represents the matching indictor. Thus, $a_{i,j}^t = 1$ represents the i-th client will upload its update parameters via the j-th channel. Moreover, the indictor must satisfy the constraint $\sum_{j \in \mathcal{N}} a_{i,j}^t \leqslant 1$ to avoid the channel conflict. In this way, we can rewrite the aggregation process in terms of channel allocation results as

$$\boldsymbol{w}^t = \boldsymbol{w}^{t-1} + \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{N}} p_i^t a_{i,j}^t \Delta \boldsymbol{w}_i^t. \tag{5}$$

We can observe that $\sum_{j \in \mathcal{N}} a_{i,j}^t = 1$ if the i-th client has been selected at the t-th communication round, and $\sum_{j \in \mathcal{N}} a_{i,j}^t = 0$ otherwise. In addition, the set of selected clients at the t-th communication round can be represented by $\mathcal{S}^t = \{i, \forall i \in \mathcal{U} | \sum_{j \in \mathcal{N}} a_{i,j}^t = 1\}$.

3.4 Differentially private FL over wireless networks

In differentially private FL, we utilize the DP-SGD mechanism, i.e., clipping the gradient with the threshold C and adding Gaussian noise with the variance $\widehat{\sigma}^2 C^2 \boldsymbol{I}$, where $\boldsymbol{I}$ is an identify matrix. In this way, the uploaded models/gradients from clients will satisfy the DP guarantee. We know that if the client participates in the aggregation more, there will be more privacy leakage. Thus, we also need to calculate the accumulative privacy budget across multiple training rounds. When the accumulative privacy budget $\widehat{\epsilon}_i$ of the i-th client is up to its preset PL ϵ_i at the next round, this client will be removed from the clients set. Based on the calculation method in [22], we can calculate the accumulative privacy budget for the i-th client after $\overline{t}$ times of model upload (exposures) as

$$\widehat{\epsilon}_i = \overline{\epsilon}_i + \frac{\log(\frac{1}{\delta}) + (\alpha - 1)\log(1 - \frac{1}{\alpha}) - \log(\alpha)}{\alpha - 1}, \tag{6}$$

Algorithm 1 Differentially private FL over wireless networks

Require: Each client owns a local training dataset $\mathcal{D}_i$, an individual privacy parameter (ϵ_i, δ), a preset clipping value C, and the noise STD $\widehat{\sigma}$;
Ensure: Trained global model $\boldsymbol{w}^f$;
Server initializes a global model $\boldsymbol{w}^0$, set $\widehat{\mathcal{U}} = \mathcal{U}$;
for $t = 0, 1, \ldots, T-1$ **do**
  The server selects a client set $\mathcal{S}^t$ from $\widehat{\mathcal{U}}$ and completes the resource allocation;
  The server broadcasts the global model $\boldsymbol{w}^0$ to clients in $\mathcal{S}^t$;
  for client i in $\mathcal{S}^t$ (in parallel) **do**
    Update the local model: $\boldsymbol{w}_i^{t,0} = \boldsymbol{w}^t$;
    for each local epoch ℓ from 0 to $\tau - 1$ **do**
      Clip and average gradients of samples in the batch as (7) and (8);
      Perturb the average gradient as (9);
      Update the local model using $\boldsymbol{w}_i^{t,\ell+1} = \boldsymbol{w}_i^{t,\ell} - \eta \boldsymbol{g}_i^{t,\ell}$;
    end for
    Calculate the model update by (10);
    Transmit the local model update $\Delta \boldsymbol{w}_i^t$ to the server via the allocated channel;
    Calculate the accumulative privacy budget using (6);
    Send the quit notification to the server if it exceeds the PL ϵ_i in the next communication round;
  end for
  The server aggregates all received local model differences by (5) and set $\boldsymbol{w}^f = \boldsymbol{w}^{t+1}$;
  The server removes the client with the quit notification from the client set $\widehat{\mathcal{U}}$;
end for

where $\overline{\epsilon}_i = \frac{\overline{t}\tau}{\alpha - 1} \ln \mathbb{E}_{z \sim \mu_0(z)} \left[(1 - q_i + q_i \mu_1(z)/\mu_0(z))^{\alpha} \right]$, $q_i = |\boldsymbol{b}|/|\mathcal{D}_i|$ is the sample rate in the DP-SGD training, $|\boldsymbol{b}|$ is the batch size, $\boldsymbol{b}$ is the sample set selected randomly from $\mathcal{D}_i$, $\mu_0(z)$ and $\mu_1(z)$ denote the Gaussian probability density function (PDF) of $\mathcal{N}(0, \widehat{\sigma})$ and the mixture of two Gaussian distributions $q_i \mathcal{N}(1, \widehat{\sigma}) + (1 - q_i)\mathcal{N}(0, \widehat{\sigma})$, respectively, τ is the number of local epochs, and α is a selectable variable. Based on the above preparation, the detailed steps of differentially private FL over wireless networks are introduced in Algorithm 1.

4 Gradient sparsification empowered differentially private FL

In this section, we will describe our proposed gradient sparsification empowered differentially private FL algorithm over wireless networks in detail. The motivation for the interplay between the gradient sparsification and wireless FL modules is that proper gradient sparsification is beneficial to improve both the training performance of DP-SGD and communication efficiency. Before introducing our proposed DP-SparFL algorithm, we need to solve two main challenges as follows: (1) the selection of the sparsification positions during training; (2) the selection of clipping threshold C. For the first challenge, if we use the top-K technique, there will be different binary masks corresponding to various samples in one batch, which will make the aggregated gradient from this batch not sparse enough. Thus, we use a random sparsifier to address this issue. For the second challenge, as mentioned above, the L_2 norm of each gradient in the local training needs to be clipped by a preset clipping threshold C before adding the random noise. A widely used method [18, 27] to choose C is taking the median of the norms of the unclipped parameters over the course of training. In order to upload local updates successfully over wireless networks, the gradient sparsification rates for clients at each communication need to be adjusted due to changeable channel conditions. We find that the gradient sparsification process will reduce the gradient norm. However, it is not practical to try various clipping thresholds to find the optimal one in this changeable scenario. Thus, in the following lemma, we provide an adaptive gradient clipping technique with configurable gradient sparsification rates to address the second challenge.

Lemma 1. With a given gradient sparsification rate s_i^t for the i-th client, we can adopt the clipping threshold $\sqrt{s_i^t} C$ to replace the original clipping threshold C.

Proof. Please see Appendix A.

Lemma 1 provides a feasible method to adjust the clipping threshold instead of one-by-one searching for each client with the changeable gradient sparsification rate s_i^t determined by the channel condition. Owing to the reduction of the clipping threshold, the noise variance will also be reduced, because the DP noise variance is proportional to the square of the clipping threshold.

Then, we propose the DP-SparFL algorithm based on the gradient sparsification technique to improve the local training process in Algorithm 1. The key steps of local training in the proposed DP-SparFL algorithm are described as follows.

(1) Determining the sparsification positions. Before local training, each client needs to generate a binary mask $\boldsymbol{m}_i^t = [m_{i,1}^t, \ldots, m_{i,K}^t]$, $\forall m_{i,k}^t \in \{0,1\}, k \in \{1,\ldots,K\}, \forall i \in \mathcal{S}^t$, with a random sparsifier for a given gradient sparsification rate s_i^t. The binary mask is utilized to prune the gradients in the local training process.

(2) Obtaining the sparse gradient. In local training, SGD is adopted while avoiding calculating a fraction of gradients based on the binary mask $\boldsymbol{m}_i^t$, which can be given by

$$\nabla F(\boldsymbol{w}_i^{t,\ell}, \mathcal{D}_{i,m}) = \nabla F(\boldsymbol{w}_i^{t,\ell}, \mathcal{D}_{i,m}) \odot \boldsymbol{m}_i^t, \ \forall i \in \mathcal{S}^t, \tag{7}$$

where $\odot$ denotes the element-wise product process.

(3) Perturbing the gradient. After the sparsification process, we need to clip the gradient of the m-th sample as

$$\boldsymbol{g}_i^{t,\ell} = \frac{1}{|\boldsymbol{b}|} \sum_{m \in \boldsymbol{b}} \nabla F(\boldsymbol{w}_i^{t,\ell}, \mathcal{D}_{i,m}) \cdot \min\left\{1, \frac{\|\nabla F(\boldsymbol{w}_i^{t,\ell}, \mathcal{D}_{i,m})\|}{\sqrt{s_i^t} C}\right\}, \ \forall i \in \mathcal{S}^t, \tag{8}$$

where C is the preset clipping threshold for the gradient norm of each sample. Then, we need to perturb the clipped gradient to satisfy the DP guarantee, which can be given by

$$\boldsymbol{g}_i^{t,\ell} = \boldsymbol{g}_i^{t,\ell} + \boldsymbol{n}_i^{t,\ell}, \ \forall i \in \mathcal{S}^t, \tag{9}$$

where $\boldsymbol{n}_i^{t,\ell}$ is the DP noise vector drawn from Gaussian distribution $\mathcal{N}\left(0, \widehat{\sigma}^2 s_i^t C^2 \boldsymbol{I}\right)/|\boldsymbol{b}|$. Owing to the sparsification update, we can use a smaller clipping threshold, i.e., $\sqrt{s_i^t} C$, to replace the original one as shown in Lemma 1.

(4) Generating the local update. With the sparsification, the local update can be expressed as

$$\Delta \boldsymbol{w}_i^{t,\tau} = \boldsymbol{w}_i^{t,\tau} - \boldsymbol{w}_i^{t,0} = -\eta \sum_{\ell=0}^{\tau-1} \boldsymbol{g}_i^{t,\ell} \odot \boldsymbol{m}_i^t, \ \forall i \in \mathcal{S}^t, \tag{10}$$

where η is the learning rate. Because the binary mask is unchanged via the entire local training process, the local update will also be sparse.

We can observe that a large s_i^t preserves more non-zero parameter updates of $\Delta \boldsymbol{w}_i^t$ and hence improves the learning performance, but it also increases the communication cost. Although we have introduced the key steps of the proposed DP-SparFL algorithm, the resource allocation process on the server side also needs to be optimized over wireless channels. Thus, in the following section, we will propose the communication-efficient scheduling policy design for the proposed algorithm.

5 Communication-efficient scheduling policy

In this section, we will first analyze the convergence bound of DP-SparFL. Then, we will develop a resource allocation scheme based on the convergence bound over wireless channels to improve the FL performance.

5.1 Convergence analysis

Following the literature on the convergence of gradient-based training algorithms, we make the following assumptions, which have been proven to be satisfied for most of ML models [6].

Assumption 1. We make assumptions on the global loss function $F(\cdot)$ defined by $F(\cdot) \triangleq \sum_{i=1}^{U} p_i F_i(\cdot)$, and the i-th local loss function $F_i(\cdot)$ as follows.

(1) $F_i(\boldsymbol{w})$ is L-Lipschitz smooth, i.e., $\|\nabla F_i(\boldsymbol{w}) - \nabla F_i(\boldsymbol{w}')\| \leqslant L\|\boldsymbol{w} - \boldsymbol{w}'\|$, for any $\boldsymbol{w}$ and $\boldsymbol{w}'$, where L is a constant determined by the practical loss function.

(2) The stochastic gradients are bounded, i.e., for any i and $\boldsymbol{w}$, $\mathbb{E}\left[\|\nabla F_i(\boldsymbol{w})\|^2\right] \leqslant G^2$.

(3) For any i and $\boldsymbol{w}$, $\|\nabla F_i(\boldsymbol{w}) - \nabla F(\boldsymbol{w})\|^2 \leqslant \varepsilon_i$, where ε_i is the divergence metric. we also define $\varepsilon \triangleq \mathbb{E}_i[\varepsilon_i]$.

Then, based on the above assumptions, we present the convergence results for general loss functions. As $F(\cdot)$ may be non-convex, we study the gradient of the global model $\boldsymbol{w}^t$ as t increases. Specifically, we aim to find an upper bound on $\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla F(\boldsymbol{w}^t)\|^2]$ to analyze the convergence rate. We offer the following theorem.

Theorem 1. Let (1)–(3) in Assumption 1 hold, and L, η, and τ be as defined therein and selected properly to ensure $\eta L\tau < 1$, and then the convergence bound of DP-SparFL can be given as

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla F\left(\boldsymbol{w}^t\right)\right\|^2\right] \leqslant \frac{2(F(\boldsymbol{w}^0)-F(\boldsymbol{w}^T))}{\eta\tau T} + \underbrace{3\varepsilon}_{\text{Caused by the data divergence}} + \underbrace{\frac{3G^2}{T}\sum_{t=0}^{T-1}\sum_{i\in\mathcal{U}}p_i^t\sum_{j=1}^{N}a_{i,j}^t(1-s_i^t)}_{\text{Caused by the gradient sparsification}} + \underbrace{\eta\tau^2\Theta(1+3\eta L\tau)}_{\text{Caused by DP}}, \tag{11}$$

where Θ is the expectation of L_2 norm of the DP noise vector.

Proof. Please see Appendix B.

It can be found from the right-hand side (RHS) of (11) that the convergence of DP-SparFL is affected by various parameters, including the gradient sparsification rate s_i^t, the divergence metric ε, and the noise variance. This convergence bound can achieve an $O(\frac{1}{\eta\tau T})$ convergence rate. Moreover, we list several important insights as follows.

- Firstly, when the i-th client adopts a small gradient sparsification rate, the convergence performance will decrease. However, in the resource-limited scenario, the gradient sparsification can reduce the transmit delay.
- Secondly, the divergence metric ε is also an important factor for the convergence performance, which reflects the data divergence for different clients. A balanced data distribution can directly accelerate the algorithm convergence.
- Last but not the least, the noise variance will also influence the convergence largely. It can be found that the proposed adaptive gradient clipping technique will improve the convergence performance by reducing the clipping threshold.

Hence, in the following subsection, we aim to maximize the FL training performance by trying to guarantee a high gradient sparsification rate (a large percentage of elements that have been retained) for each client under the training delay and client fairness constraints.

5.2 Optimal gradient sparsification rate and wireless resource allocation

Let us investigate the online scenario, where the channel allocation $a_{i,j}^t$, transmit power P_i^t, and the gradient sparsification rate s_i^t of each client are optimized for each communication round. To improve the learning performance, we choose to minimize the convergence bound in the RHS of (11). Because the privacy cost is determined by the participation times and the model publishing times, as shown in (6), clients will have different participation rates (i.e., client fairness [4, 28]). Thus, we calculate the participant rate based on (6) for the i-th client as

$$\beta_i = \min\left\{\frac{N\widehat{T}_i}{\sum_{i'=1}^{U}\widehat{T}_{i'}}, 1\right\}, \tag{12}$$

where

$$\widehat{T}_i = \left\lfloor \frac{(\alpha-1)\epsilon_i - \log(\frac{1}{\delta}) - (\alpha-1)\log(1-\frac{1}{\alpha}) + \log(\alpha)}{\tau\ln\mathbb{E}_{z\sim\mu_0(z)}[(1-q_i+\frac{q_i\mu_1(z)}{\mu_0(z)})^\alpha]} \right\rfloor, \tag{13}$$

$\widehat{T}_i$ is the number of communication rounds that the i-th client can participate in under the DP constraint ϵ_i using noise STD $\widehat{\sigma}$, and $\lfloor\cdot\rfloor$ denotes rounding down. The delay of each communication round is determined by the slowest client, and given by $d^t = \max_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}}a_{i,j}^t d_{i,j}^t$ and $d_{i,j}^t = d_i^{t,\mathrm{do}} + d_i^{t,\mathrm{lo}} + d_{i,j}^{t,\mathrm{up}}$. By maximizing the FL training performance with the training delay and client fairness constraints, we

formulate the following resource allocation problem:

$$\begin{aligned}
\mathrm{P1}: \quad & \min_{\substack{P_i^t, s_i^t, a_{i,j}^t, \\ i=1,\ldots,U}} -\frac{1}{T}\sum_{t=0}^{T-1}\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}} a_{i,j}^t p_i^t s_i^t \\
\text{s.t. } & \mathrm{C1}: a_{i,j}^t \in \{0,1\}, \qquad \mathrm{C2}: \sum_{j\in\mathcal{N}} a_{i,j}^t = 1, \qquad \mathrm{C3}: \sum_{i\in\mathcal{U}} a_{i,j}^t \leqslant 1, \\
& \mathrm{C4}: s^{\mathrm{th}} \leqslant s_i^t \leqslant 1, \qquad \mathrm{C5}: 0 < P_i^t \leqslant P^{\max}, \quad \mathrm{C6}: \sum_{j\in\mathcal{N}} a_{i,j}^t E_{i,j}^{t,\mathrm{co}} + E_i^{t,\mathrm{cp}} \leqslant E^{\max}, \\
& \mathrm{C7}: \frac{1}{T}\sum_{t=0}^{T}\sum_{j\in\mathcal{N}} a_{i,j}^t \leqslant \beta_i, \ \mathrm{C8}: \frac{1}{T}\sum_{t=0}^{T} d^t \leqslant d^{\mathrm{Avg}},
\end{aligned}$$

where $P^{\max}$ is the maximum available transmit power, $E^{\max}$ is the maximum available energy, d^{Avg} is the required average training latency, and s^{th} is to restrict the gradient sparsification rate since the learning accuracy decreases sharply when the gradient sparsification rate is very low [29]. The constraints C1–C3 represent the wireless channel allocation and C4 is to restrict the gradient sparsification rate. C5 and C6 represent the transmit power constraint and the total power consuming, respectively. C7 and C8 are adopted to restrict the client fairness and learning latency, respectively. Obviously, the problem is a mixed integer optimization problem, which is non-convex in general and cannot be directly solved. Therefore, in the following, we first decompose the original problem, and then develop a low computational complexity algorithm to achieve the suboptimal solutions.

We can observe that P1 is a stochastic optimization problem with long-term constraints C7 and C8, which can be transformed into part of the objective function [30]. First, we leverage the Lyapunov technique to transform constraint C7 and C8 into queue stability constraints [31]. In detail, we define virtual queues $Q_i^{t,\mathrm{fa}}$ and $Q^{t,\mathrm{de}}$ with the following update equations: $Q_i^{t,\mathrm{fa}} \triangleq [Q_i^{t-1,\mathrm{fa}} + \sum_{j=1}^{N} a_{i,j}^t - \beta_i]^+$ and $Q^{t,\mathrm{de}} \triangleq [Q^{t-1,\mathrm{de}} + d^t - d^{\mathrm{Avg}}]^+$, respectively, where $[x]^+ \triangleq \max\{x, 0\}$. Under the framework of Lyapunov optimization, we further resort to the drift-plus-penalty algorithm and obtain

$$\begin{aligned}
\mathrm{P2}: \quad & \min_{\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t} \frac{1}{T}\sum_{t=0}^{T-1} V^t(\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t) \\
\text{s.t.} \quad & \mathrm{C1}: a_{i,j}^t \in \{0,1\}, \quad \mathrm{C2}: \sum_{j\in\mathcal{N}} a_{i,j}^t = 1, \qquad \mathrm{C3}: \sum_{i\in\mathcal{U}} a_{i,j}^t \leqslant 1, \\
& \mathrm{C4}: s^{\mathrm{th}} \leqslant s_i^t \leqslant 1, \quad \mathrm{C5}: 0 < P_i^t \leqslant P^{\max}, \quad \mathrm{C6}: \sum_{j\in\mathcal{N}} a_{i,j}^t E_{i,j}^{t,\mathrm{co}} + E_i^{t,\mathrm{cp}} \leqslant E^{\max},
\end{aligned}$$

where

$$V^t(\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t) = \sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}} (Q_i^{t,\mathrm{fa}} - \lambda p_i^t s_i^t) a_{i,j}^t + Q^{t,\mathrm{de}}\left(d^t - d^{\mathrm{Avg}}\right) - Q_i^{t,\mathrm{fa}} \sum_{i\in\mathcal{U}} \beta_i, \tag{14}$$

$\boldsymbol{P}^t = \{P_i^t | i \in \mathcal{U}\}$, $\boldsymbol{s}^t = \{s_i^t | i \in \mathcal{U}\}$, and $\lambda > 0$ is a tuneable parameter that controls the trade-off between minimizing the convergence bound and the training delay. We can observe that P2 can be divided into T independent sub-optimization problems, for each communication round.

5.2.1 *Optimal gradient sparsification rate*

The optimal gradient sparsification rate of the i-th client can be determined by the following theorem.

Theorem 2. Given the client scheduling vector $\boldsymbol{a}^t$ and transmit power $\boldsymbol{P}^t$, the solution of the optimal gradient sparsification rate vector at the t-th communication round can be divided into N subproblems with closed-form solutions, and then solved as

$$\boldsymbol{s}^{t,\star} = \arg\min_{\boldsymbol{s}^t \in \mathcal{K}^t} V^t\left(\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t\right), \tag{15}$$

where $\mathcal{K}^t$ is the set of all available solutions for N subproblems.

Proof. Please see Appendix C.

From Theorem 2, we can observe that the optimization problem for the gradient sparsification rate vector can be divided into N subproblems, and each of them can be solved with the closed-form solution. We can address all N subproblems, and then select the optimal solution as the final output. Therefore, the complexity of optimizing the optimal gradient sparsification rate is linear with N.

5.2.2 *Optimal transmit power*

To obtain the optimal transmit power, we first derive the relation between P_i^t and $V^t(\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t)$ as

$$\frac{\partial V^t(\boldsymbol{P}^t, \boldsymbol{s}^t, \boldsymbol{a}^t)}{\partial P_i^t} = \begin{cases} \sum\limits_{j\in\mathcal{N}} \frac{-(\ln 2)\lambda Z p_i^t s_i^t h_{i,j}^t a_{i,j}^t}{B\sigma^2(\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2}))^2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})}, & \text{if } \sum\limits_{j\in\mathcal{N}} a_{i,j}^t d_{i,j}^t > \sum\limits_{j\in\mathcal{N}} a_{i',j}^t d_{i',j}^t, \forall i' \in \mathcal{S}^t/i, \\ 0, & \text{else.} \end{cases} \tag{16}$$

Then, we can derive the first derivative between $E_i^{t,\mathrm{co}}$ and P_i^t as

$$\begin{aligned} \frac{\partial E_i^{t,\mathrm{co}}}{\partial P_i^t} &= \sum_{j\in\mathcal{N}} \frac{-(\ln 2) P_i^t Z p_i^t s_i^t h_{i,j}^t a_{i,j}^t}{B\sigma^2(\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2}))^2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})} + \frac{Z p_i^t s_i^t a_{i,j}^t}{B\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})} \\ &= \sum_{j\in\mathcal{N}} \frac{(Z p_i^t s_i^t(\sigma^2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2}) - (\ln 2) P_i^t h_{i,j}^t))a_{i,j}^t}{B\sigma^2(\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2}))^2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})} > 0. \end{aligned} \tag{17}$$

Therefore, the largest transmit power under the constraint can be applied, i.e.,

$$P_i^{t,\star} = \max\{P^{\max}, P^{t,\mathrm{th}}\}, \tag{18}$$

where $P^{t,\mathrm{th}}$ satisfies the equation:

$$\sum_{j\in\mathcal{N}} \frac{P_i^{t,\mathrm{th}} p_i^t s_i^t Z a_{i,j}^t}{B\log_2(1+\frac{P_i^{t,\mathrm{th}} h_{i,j}^t}{\sigma^2})} = E^{\max} - E_i^{t,\mathrm{cp}}. \tag{19}$$

5.2.3 *Optimal channel allocation*

When $Q^{t,\mathrm{de}} > 0$, based on the optimizing process above, we can simplify the optimization problem P2 as

$$\begin{aligned} \mathrm{P2}: \min_{\boldsymbol{a}^t} & \sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}} (Q_i^{t,\mathrm{fa}} - \lambda p_i^t s_i^t) a_{i,j}^t + Q^{t,\mathrm{de}} \max_{i\in\mathcal{N}} \left\{ \sum_{j=1}^{N} a_{i,j}^t \left(\frac{Z p_i^t s_i^t}{B\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})} + d_i^{t,\mathrm{do}} + \frac{\tau|\mathcal{D}_i|\Phi_i}{f_i^t} \right) \right\} \\ \text{s.t. } \mathrm{C1}: a_{i,j}^t \in \{0,1\}, \mathrm{C2}: & \sum_{j\in\mathcal{N}} a_{i,j}^t = 1, \mathrm{C3}: \sum_{i\in\mathcal{U}} a_{i,j}^t \leqslant 1, \mathrm{C6}: \sum_{j\in\mathcal{N}} a_{i,j}^t E_{i,j}^{t,\mathrm{co}} + E_i^{t,\mathrm{cp}} \leqslant E^{\max}. \end{aligned} \tag{20}$$

Obviously, the objective function (20) is a mix-optimization problem, and the optimization variables are integers. We can use μ to replace the second term in P2, problem (20) can be transformed as

$$\begin{aligned} \mathrm{P3}: \quad & \min_{\mu, \boldsymbol{a}^t} \sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}} (Q_i^{t,\mathrm{fa}} - \lambda p_i^t s_i^t) a_{i,j}^t + \mu \\ \text{s.t.} \quad & \mathrm{C1}\text{–}\mathrm{C3}, \mathrm{C6}, \mathrm{C9}: \mu \geqslant \max_{i\in\mathcal{U}} \left\{ Q^{t,\mathrm{de}} \sum_{j=1}^{N} a_{i,j}^t \left(\frac{Z p_i^t s_i^t}{B\log_2(1+\frac{P_i^t h_{i,j}^t}{\sigma^2})} + d_i^{\mathrm{do}} + \frac{\tau|\mathcal{D}_i|\Phi_i}{f_i^t} \right) \right\}. \end{aligned} \tag{21}$$

Using this form, we can iteratively solve two variables μ and $a_{i,j}^t$, which correspond to two subproblems, i.e.,

$$\mathrm{P31}: \quad \min_{\mu} \mu \quad \text{s.t.} \quad \mathrm{C9}. \tag{22}$$

and

$$\mathrm{P32}: \quad \min_{\boldsymbol{a}^t} \sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{N}} (Q_i^{t,\mathrm{fa}} - \lambda p_i^t s_i^t) a_{i,j}^t \quad \text{s.t.} \quad \mathrm{C1}\text{–}\mathrm{C3}, \mathrm{C6}, \text{ and } \mathrm{C9}. \tag{23}$$

Further, the optimal solution of problem P31 can be expressed as

$$\mu^{\star} = \max_{i \in \mathcal{U}} \left\{ Q^{t,\mathrm{de}} \sum_{j \in \mathcal{N}} a_{i,j}^{t} \left(\frac{Z p_i^t s_i^t}{B \log_2(1 + \frac{P_i^t h_{i,j}^t}{\sigma^2})} + d_i^{\mathrm{do}} + \frac{\tau |\mathcal{D}_i| \Phi_i}{f_i^t} \right) \right\}. \tag{24}$$

The problem P32 can be addressed by using a bipartite matching algorithm, which aims to maximize the objective function. To use a bipartite matching algorithm for solving problem P32, we first transform the optimization problem into a bipartite matching problem. We construct a bipartite graph $\mathcal{A} = (\mathcal{U} \times \mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of channels that can be allocated to each client, each vertex in $\mathcal{U}$ represents a client, each vertex in $\mathcal{N}$ represents a channel, and $\mathcal{E}$ is the set of edges that connect to the vertices from each set $\mathcal{U}$ and $\mathcal{N}$. Let $a_{i,j}^t \in \mathcal{E}$ be the edge connecting vertex i in $\mathcal{U}$ and vertex j in $\mathcal{N}$ with $a_{i,j}^t \in \{0, 1\}$, where $a_{i,j}^t = 1$ indicates that the j-th channel is allocated to the i-th client, otherwise, we have $a_{i,j}^t = 0$. We aim to find a subset of edges in $\mathcal{E}$, in which no two edges share a common vertex in $\mathcal{N}$, such that each channel can only be allocated to one client. Here, we first prune the edges that cannot satisfy the constraint C9 and obtain a non-full-connected bipartite graph. Then, we can use the conventional Hungarian search method [32] to solve the optimal matching result $\boldsymbol{a}^{t,*}$. When $Q^{t,\mathrm{de}} \leqslant 0$, we can note that the optimization problem can be solved by the conventional Hungarian search method.

Overall, we can optimize the gradient sparsification rate $\boldsymbol{s}^t$, transmit power vector $\boldsymbol{P}^t$, and channel allocation matrix $\boldsymbol{a}^t$ for training the FL algorithm successively, until the decrement between two adjacent objective functions is smaller than a preset value.

5.3 Feasibility analysis

In this subsection, we show the proposed solution for P1 can satisfy the client fairness and the required average delay, and the virtual queue system defined is mean rate stable. We state the feasibility analysis as follows.

Theorem 3. The proposed solution for P1 is feasible. Specifically, for any minimum selection fraction and required average delay, the virtual queue system defined is strongly stable.

Proof. Please see Appendix D.

From Theorem 3, we can note that the long-term fairness and required average delay constraints can be satisfied with the proposed solution as long as the requirement is feasible. Specifically, this theorem also reveals the fact that the long-term client fairness constraint holds under any setting of the tuneable parameter λ. With a larger value of λ, the fairness and delay queues will have a slower convergence rate, indicating that the fairness and average delay could not be well guaranteed before convergence. When the training rounds are finite, the number of rounds that need to undergo before convergence could compromise the fairness and average delay to some degree. We will verify our analysis via experiment results in Section 6.

5.4 Complexity analysis

The proposed optimization method includes three parts at each communication round, i.e., optimizing gradient sparsification rate, optimizing transmit power, and channel allocation. First, for optimizing the gradient sparsification rate, it is divided into N subproblems, where each subproblem can be solved directly. Thus, the complexity of optimizing gradient sparsification rate is expressed as $\mathcal{O}(N)$. Secondly, optimizing transmit power is addressed by the convex optimization method and its complexity is given as $\mathcal{O}(N)$. Thirdly, channel allocation has been divided into two subproblems; these two subproblems are addressed by the convex optimization method and the Hungarian search method, respectively. The complexity of the Hungarian search method is expressed as $\mathcal{O}(U^2(N-1)N)$. We can notice that the relationship between the complexity and the key variables (i.e., N and U) is not exponential, which facilitates real-life applications.

6 Experiments

6.1 Setup

We evaluate the effectiveness of the proposed algorithm on two different convolutional neural networks (CNNs) and three datasets as follows:

• **CNN on MNIST and FashionMNIST.** The MNIST dataset [33] consists of handwritten digits with 10 categories formatted as 28×28 size gray scale images. There are 60000 training examples and 10000 testing examples. FashionMNIST [34] is a dataset of Zalando's article images formatted as 28×28 size gray scale images, which consists of 60000 training samples and 10000 testing samples. The CNN model for MNIST and FashionMNIST datasets contains two 5×5 convolution layers (the first layer consists of 32 filters, the second layer consists of 64 filters, each layer is followed with 2×2 max pooling and ReLu activation), a fully connected layer with 512 units followed with ReLu activation, and a final softmax output layer.

• **CNN on CIFAR-10.** The CIFAR-10 dataset [35] contains 60000 color images with 10 object classes, e.g., dog, deer, and airplane, where each class has 6000 images. This dataset has been pre-divided into two parts, i.e., the training dataset (50000 images) and the test dataset (10000 images). The CNN model used for CIFAR-10 contains three 3×3 convolution layers, two fully connected layers, and a final softmax output layer. The first, second, and third convolution layers consist of 64, 128, and 256 filters, respectively, where each layer is followed by 2×2 max pooling and ReLu activation. The first and second fully connected layers contain 128 and 256 units, respectively, in which each layer is followed by ReLu activation.

To evaluate the performance, we compare the proposed DP-SparFL algorithm with the following baselines. (1) Random scheduling [5]. The AP selects N associated clients and assigns dedicated channels for them randomly at each communication round, where these selected clients perform local training and upload their local updates with the assigned channels. (2) Round robin [5]. The AP arranges all the clients into $\lceil \frac{U}{N} \rceil$ groups, where $\lceil \cdot \rceil$ denotes rounding up to an integer, and then assigns each group to access the channels consecutively. (3) Delay-minimization [36]. The AP selects a client set, consisting of N clients, from available clients with the minimizing training delay without update sparsification.

For these classification tasks, we use the cross-entropy loss function. Unless otherwise stated, the system parameters are set as follows. We set the number of clients to 20, each containing 1000 training and 500 testing examples. For the non-IID setting, we simulate a heterogeneous partition by simulating a Dirichlet distribution, i.e., Dir(0.2), over all classes for each client [37]. As down in [37], this setting draws from a Dirichlet distribution Dir(γ), where $\gamma > 0$ is a concentration parameter controlling the identicalness among clients. With $\gamma \to \infty$, all clients have identical distributions to the prior; with $\gamma \to 0$, on the other extreme, each client holds samples from only one class chosen at random. For the condition of imbalance data numbers, we divide the 20 clients into four parts, and they have 300, 600, 1800, and 2100 samples for each client, respectively. When training with FL, we set the learning rate η as 0.002 for the MNIST, FashionMNIST, and CIFAR-10 datasets. The number of local iterations is set to 60. Besides, simulations are performed in a square area of 100 m $\times$ 100 m. Both available channels and clients are uniformly distributed in this plane. The number of available channels is set to 5. We set the bandwidth B to 15 kHz, the transmission power of the downlink channel to 23 dBm, the maximum transmission power of the client to 30 dBm, Gaussian white noise power to -107 dBm, and path loss exponent model to PLE, in which PLE (dB) $= 128.1 + 37.6 \log(\chi)$ with χ representing the distance in km. The maximum computing capability f_i^t and the maximum power constraint of each client are set to 2.4 GHz and 200 mW, respectively. In addition, the required PL ϵ_i of the i-th client is randomly distributed between 2.0 and 10.0 similar to [38], and unchanged throughout the FL training, as well as $\delta = 0.001, \forall i \in \mathcal{U}$. We use different noise STDs for different datasets, i.e., $\widehat{\sigma}_i = 0.6$, $\widehat{\sigma}_i = 0.5$, and $\widehat{\sigma}_i = 0.4$ for MNIST, FashionMNIST, and CIFAR-10, respectively. We utilize the method in [18] and choose C by taking the median of the norms of the unclipped parameters over the course of training. However, the gradient sparsification rates for clients' updates at each communication need to be adjusted to address the changeable channel conditions. We adopt the adaptive DP clipping threshold technique in Lemma 1 to address this challenge. To balance the learning performance and training delay, we adopt $\lambda = 50$.

6.2 Impact of the clipping threshold

To evaluate the adjusting method in Lemma 1, we show the test accuracy of the proposed DP-SparFL algorithm under the unadjusted and adjusted clipping thresholds on the FashionMNIST and CIFAR-10 datasets in Figure 2. The original clipping threshold is the median value obtained by the pretraining process [18]. We can note that in our system, the gradient sparsification rate for each client is uncertain and changes with the channel condition; thus it is unavailable to use the pretraining method at each communication round. We can see that under various gradient sparsification rates, the adjusted method

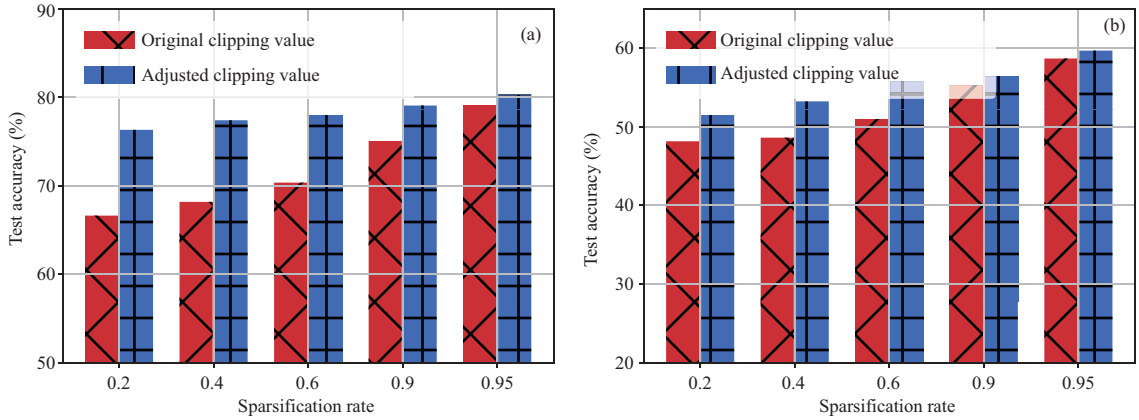

Figure 2 (Color online) Test accuracy of original clipping and adjusted clipping methods (Lemma 1) on (a) FashionMNIST and (b) CIFAR-10 datasets with various sparsification rates (percentages of elements that have been retained).

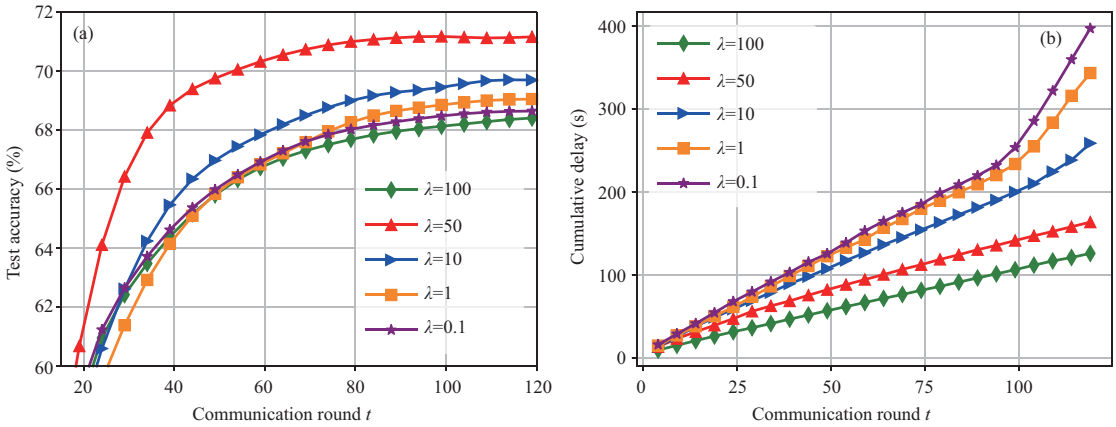

Figure 3 (Color online) Test accuracy (a) and cumulative delay (b) for the proposed DP-SparFL algorithm with various values of λ in the IID setting on the FashionMNIST dataset.

outperforms the unadjusted one on both FashionMNIST and CIFAR-10 datasets. The intuition is that a smaller gradient sparsification rate could lead to a smaller L_2 norm for the training gradient for each client; thus a smaller clipping value can reduce the noise variance and improve the learning performance.

6.3 Impact of the tuneable parameter λ and privacy level

Figure 3 illustrates the test accuracy and cumulative delay for the proposed DP-SparFL algorithm with various values of λ on the FashionMNIST dataset. From Figure 3(a), we can see that there is an optimal λ in view of the test accuracy. In Figure 3(b), we can observe that a larger λ will have a larger training delay. The rationale behind this is that a larger λ can lead to a higher consideration for the update sparsification but a smaller consideration for the training delay. The phenomenon inspires us to find a better trade-off between the training latency and update sparsification to achieve optimal training performance. When the channel condition is poor, the system tends to use a lightweight transmission, i.e., a small sparsification rate. However, we know that if the sparsification rate is too small, this sparse gradient will degrade the training performance. In this way, during the training process, our work will maintain a balance among training performance, client scheduling, and transmission power dynamically, which can find a better trade-off between performance and transmission overhead than baselines.

In Figure 4, we show the test accuracy and cumulative delay for the proposed DP-SparFL algorithm with various values of average privacy levels on the FashionMNIST dataset. We can see that the test accuracy increases with the PL. In addition, a larger PL allows clients to participate in FL training more, i.e., a larger number of communication rounds for training. In Figure 4(b), it can be noted the cumulative delay decreases with the average PL. The intuition is that clients with larger PLs can be scheduled flexibly with few privacy concerns.

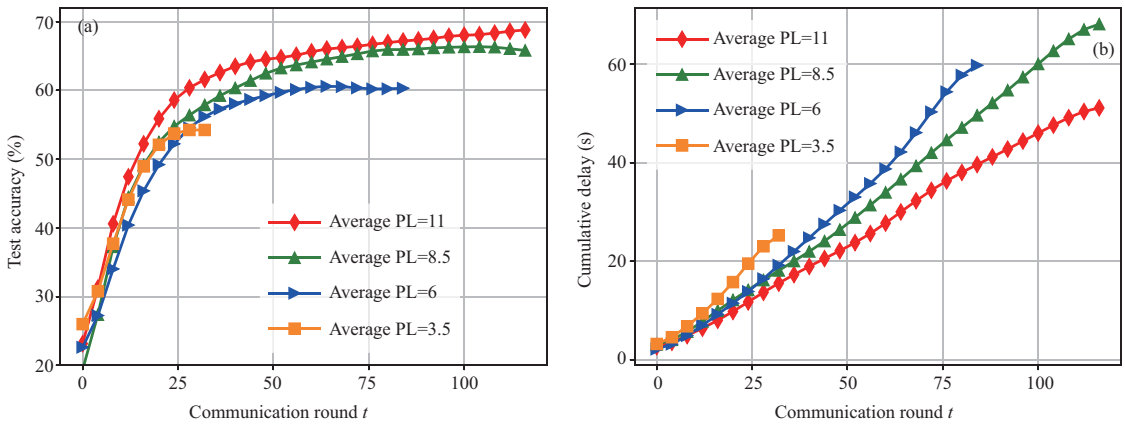

Figure 4 (Color online) Test accuracy (a) and cumulative delay (b) for the proposed DP-SparFL algorithm with various average privacy levels (PLs) on the FashionMNIST dataset. Specifically, the required DP levels of clients are randomly sampled from four intervals, i.e., $[2, 20]$, $[2, 15]$, $[2, 10]$, and $[2, 5]$.

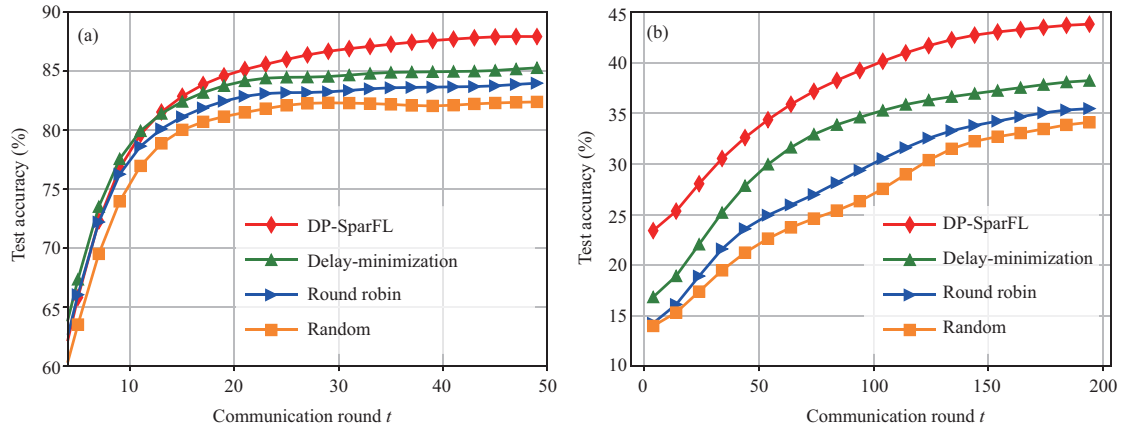

Figure 5 (Color online) Test accuracy for various algorithms in the IID data setting on the (a) MNIST and (b) CIFAR-10 datasets, i.e., DP-SparFL, delay-minimization, round robin, and random algorithms.

6.4 Comparison of the various policies

Figure 5 illustrates the test accuracy of the proposed DP-SparFL ($\lambda = 50$), delay-minimization, round robin, and random algorithms in the IID data setting on the MNIST and CIFAR-10 datasets. As seen from Figures 5(a) and (b), the proposed DP-SparFL algorithm achieves better test accuracy than baselines. The reason is that the proposed DP-SparFL algorithm can guarantee more accessible clients owing to the sparsification technique, thereby improving the learning performance. Moreover, the value of λ is a key factor to balance the trade-off between the latency and the learning performance.

In Figure 6, we show cumulative latencies for various algorithms in the IID data setting on the MNIST and CIFAR-10 datasets, i.e., DP-SparFL, delay-minimization, round robin, and random algorithms. As shown in Figure 6, the proposed DP-SparFL algorithm can obtain the lowest latency among whole algorithms. It can be noted that the delay of the delay-minimization algorithm is low in the early stages of the training, but it increases tremendously in the later stages. The intuition is that the delay-minimization algorithm will select the clients with low delays, in which clients with low distances have high probabilities. When the privacy budgets of these clients have run out, the cumulative latency will increase tremendously. The proposed DP-SparFL algorithm can schedule clients with large distances with the help of sparsification, while taking account of client fairness (clients's privacy).

In Figures 7 and 8, we show the test accuracy and cumulative delay under two different data conditions, i.e., non-IID and imbalance sample numbers, respectively. We can see that the proposed algorithm can obtain a higher test accuracy due to the fairness guarantee that is beneficial to non-IID data. The reason is that in the non-IID data setting, the proposed algorithm ensures that all clients can participate in FL training with the client fairness constraint. In addition, the proposed algorithm also takes account of the

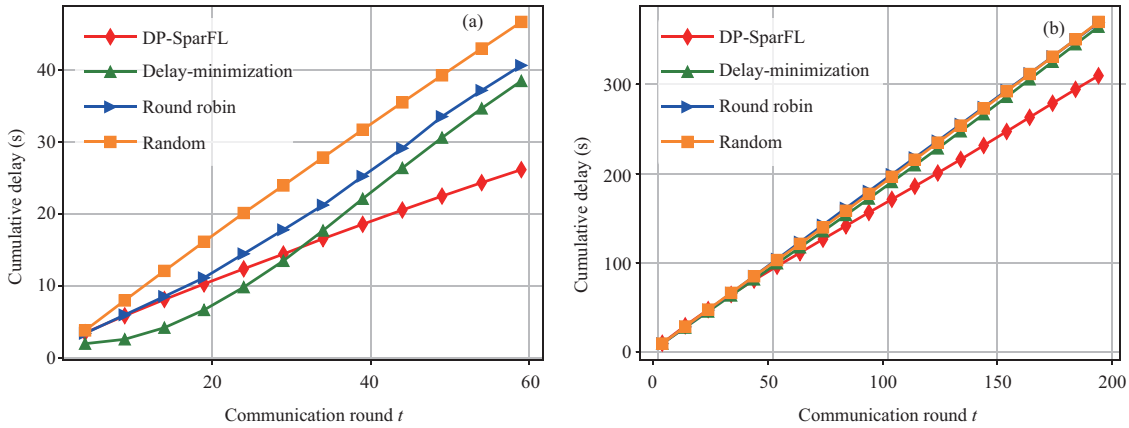

Figure 6 (Color online) Cumulative delay for various algorithms in the IID data setting on the (a) MNIST and (b) CIFAR-10 datasets, i.e., DP-SparFL, delay-minimization, round robin, and random algorithms.

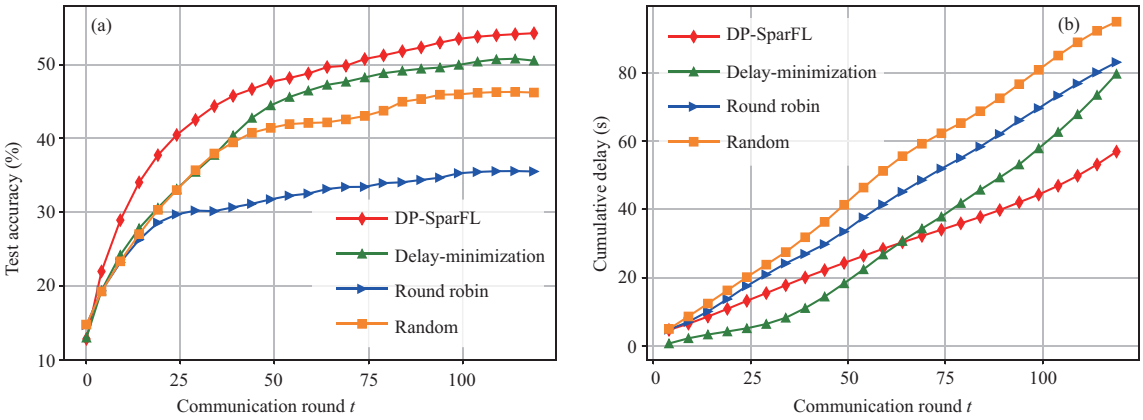

Figure 7 (Color online) Test accuracy (a) and cumulative delay (b) for various algorithms in the non-IID data setting on the FashionMNIST dataset, i.e., DP-SparFL, delay-minimization, round robin, and random algorithms.

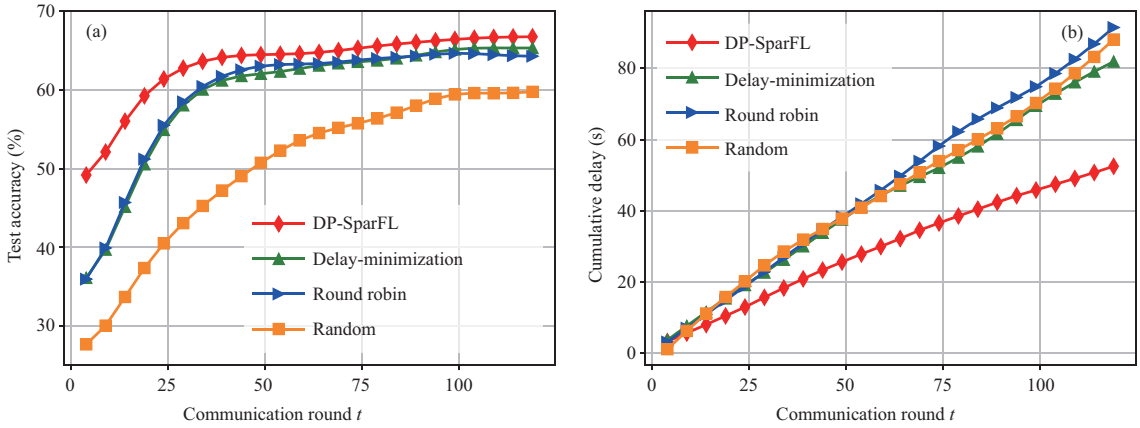

Figure 8 (Color online) Test accuracy (a) and cumulative delay (b) for various algorithms under the imbalance data condition on the FashionMNIST dataset, i.e., DP-SparFL, delay-minimization, round robin, and random algorithms.

size of the dataset (affecting the sampling rate q_i in the local training) when designing the client fairness; thus it can allocate clients that contain large datasets to high-quality channels.

7 Conclusion

In this paper, we have developed a novel framework for FL over a wireless network with imbalanced resources and DP requirements among clients. We have proposed a gradient sparsification empowered differentially private FL by an adaptive gradient clipping technique that reduces the connections in gradients of local training for each client before DP noise perturbation. Then, we have analyzed the convergence bound in terms of gradient sparsification rates by considering a non-convex FL problem. To further improve the training performance and efficiency, a novel stochastic optimization problem has been formulated to minimize the convergence bound, while satisfying transmit power, average delay, and client DP requirement constraints. We have applied the Lyapunov drift-plus-penalty framework to address the optimization problem. Our experimental results have exhibited the superiority of our proposed algorithms compared with baselines, i.e., random scheduling, round robin, and delay-minimization algorithms. In the proposed framework, we can see that a random sparsifier cannot preserve the most important elements of the per-sample gradient, but a top-K sparsification technique will make binary masks from different gradients in one batch diverse. Thus, an interesting direction for future work is to achieve a unified binary mask for the whole local training process with a low privacy budget, to enhance privacy protection and communication efficiency.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (Grant Nos. 62071296, 62002170, 62071234, U22A2002), National Key Research and Development Program of China (Grant No. 2020YFB1807700), Fundamental Research Funds for the Central Universities (Grant No. 30921013104), Key Technologies R&D Program of Jiangsu (Prospective and Key Technologies for Industry) (Grant Nos. BE2023022, BE2023022-2), Future Network Grant of Provincial Education Board in Jiangsu, Major Science and Technology Plan of Hainan Province (Grant No. ZDKJ2021022), Scientific Research Fund Project of Hainan University (Grant No. KYQD(ZR)-21008), Youth Foundation Project of Zhejiang Lab (Grant No. K2023PD0AA01), Collaborative Innovation Center of Information Technology, Hainan University (Grant No. XTCX2022XXC07), and Sciences and Technology Commission of Shanghai Municipality (Grant Nos. 22JC1404000, 20JC1416502, PKX2021-D02).

Supporting information Appendixes A–D. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Nguyen D C, Cheng P, Ding M, et al. Enabling AI in future wireless networks: a data life cycle perspective. IEEE Commun Surv Tutorials, 2021, 23: 553–595
2 Kairouz P, McMahan H B, Avent B, et al. Advances and open problems in federated learning. FNT Machine Learn, 2021, 14: 1–210
3 Li J, Shao Y, Wei K, et al. Blockchain assisted decentralized federated learning (BLADE-FL): performance analysis and resource allocation. IEEE Trans Parallel Distrib Syst, 2021, 33: 2401–2415
4 Xia W, Quek T Q S, Guo K, et al. Multi-armed bandit-based client scheduling for federated learning. IEEE Trans Wireless Commun, 2020, 19: 7108–7123
5 Yang H H, Liu Z, Quek T Q S, et al. Scheduling policies for federated learning in wireless networks. IEEE Trans Commun, 2020, 68: 317–333
6 Wang S, Tuor T, Salonidis T, et al. Adaptive federated learning in resource constrained edge computing systems. IEEE J Sel Areas Commun, 2019, 37: 1205–1221
7 Chen M, Poor H V, Saad W, et al. Convergence time optimization for federated learning over wireless networks. IEEE Trans Wireless Commun, 2021, 20: 2457–2471
8 Yang Z, Chen M, Saad W, et al. Energy efficient federated learning over wireless communication networks. IEEE Trans Wireless Commun, 2021, 20: 1935–1949
9 Deng X, Li J, Ma C, et al. Blockchain assisted federated learning over wireless channels: dynamic resource allocation and client scheduling. IEEE Trans Wireless Commun, 2022, 22: 3537–3553
10 Mocanu D C, Mocanu E, Stone P, et al. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. Nat Commun, 2018, 9: 2383
11 Jiang J, Fu F, Yang T, et al. SketchML: accelerating distributed machine learning with data sketches. In: Proceedings of International Conference on Management of Data, Houston, 2018. 1269–1284
12 Zheng S, Shen C, Chen X. Design and analysis of uplink and downlink communications for federated learning. IEEE J Sel Areas Commun, 2021, 39: 2150–2167
13 Wang Y, Xu Y, Shi Q, et al. Quantized federated learning under transmission delay and outage constraints. IEEE J Sel Areas Commun, 2022, 40: 323–341
14 Liu S, Yu G, Yin R, et al. Joint model pruning and device selection for communication-efficient federated edge learning. IEEE Trans Commun, 2022, 70: 231–244
15 Wang Z, Song M, Zhang Z, et al. Beyond inferring class representatives: user-level privacy leakage from federated learning. In: Proceedings of IEEE International Conference on Computer Communications, Paris, 2019. 2512–2520
16 Dwork C, Roth A. The algorithmic foundations of differential privacy. FNT Theor Comput Sci, 2014, 9: 211–407
17 Wei K, Li J, Ding M, et al. Federated learning with differential privacy: algorithms and performance analysis. IEEE Trans Inform Forensic Secur, 2020, 15: 3454–3469
18 Martin A, Andy C, Ian G, et al. Deep learning with differential privacy. In: Proceedings of ACM Conference on Computer and Communications Security, Vienna, 2016. 308–318
19 Zhou Y, Wu S, Banerjee A. Bypassing the ambient dimension: private SGD with gradient subspace identification. In: Proceedings of International Conference on Learning Representations, Virtual, 2021

20 Wei K, Li J, Ma C, et al. Low-latency federated learning over wireless channels with differential privacy. IEEE J Sel Areas Commun, 2020, 40: 290–307
21 Mironov I, Talwar K, Zhang L. Rényi differential privacy of the sampled Gaussian mechanism. 2019. ArXiv:1908.10530
22 Yousefpour A, Shilov I, Sablayrolles A, et al. Opacus: user-friendly differential privacy library in PyTorch. In: Proceedings of Privacy in Machine Learning Workshop, NeurIPS, Virtual, 2021
23 Stich S U, Cordonnier J, Jaggi M. Sparsified SGD with memory. In: Proceedings of Advances in Neural Information Processing Systems, Montreal, 2018, 4452–4463
24 Dinh C T, Tran N H, Nguyen M N H, et al. Federated learning over wireless networks: convergence analysis and resource allocation. IEEE ACM Trans Networking, 2021, 29: 398–409
25 Zhou X, Deng Y, Xia H, et al. Time-triggered federated learning over wireless networks. IEEE Trans Wireless Commun, 2022, 21: 11066–11079
26 You C, Feng D, Guo K, et al. Semi-synchronous personalized federated learning over mobile edge networks. IEEE Trans Wireless Commun, 2023, 22: 2262–2277
27 Xiao H, Xiang Z, Wang D, et al. A theory to instruct differentially-private learning via clipping bias reduction. In: Proceedings of IEEE Symposium on Security and Privacy, San Francisco, 2023. 2170–2189
28 Huang T, Lin W, Wu W, et al. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. IEEE Trans Parallel Distrib Syst, 2021, 32: 1552–1564
29 Molchanov P, Mallya A, Tyree S, et al. Importance estimation for neural network pruning. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, 2019. 11264–11272
30 Neely M J. Stochastic Network Optimization with Application to Communication and Queueing Systems. Cham: Springer, 2010
31 Deng X, Li J, Ma C, et al. Low-latency federated learning with DNN partition in distributed industrial IoT networks. IEEE J Sel Areas Commun, 2022, 41: 755–775
32 Mahdian M, Yan Q. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In: Proceedings of ACM Symposium on Theory of Computing, San Jose, 2011. 597–606
33 LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. Proc IEEE, 1998, 86: 2278–2324
34 Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. 2017. ArXiv:1708.07747
35 Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images. Dissertation for Master's Degree. Toronto: University of Toronto, 2009
36 Chen M, Shlezinger N, Poor H V, et al. Communication-efficient federated learning. Proc Natl Acad Sci USA, 2021, 118: e2024789118
37 Yurochkin M, Agarwal M, Ghosh S, et al. Bayesian nonparametric federated learning of neural networks. In: Proceedings of International Conference on Machine Learning, 2019. 7252–7261
38 Liu J, Lou J, Xiong L, et al. Projected federated averaging with heterogeneous differential privacy. Proc VLDB Endow, 2021, 15: 828–840