Communication-Efficient Federated Optimization Over Semi-Decentralized Networks

He Wang, *Graduate Student Member, IEEE*, and Yuejie Chi, *Fellow, IEEE*

***Abstract*—In large-scale federated and decentralized learning, communication efficiency is one of the most challenging bottlenecks. While gossip communication—where agents can exchange information with their connected neighbors—is more cost-effective than communicating with the remote server, it often requires a greater number of communication rounds, especially for large and sparse networks. To tackle the trade-off, we examine the communication efficiency under a *semi-decentralized* communication protocol, in which agents can perform both agent-to-agent and agent-to-server communication *in a probabilistic manner*. We design a tailored communication-efficient algorithm over semi-decentralized networks, referred to as `PISCO`, which inherits the robustness to data heterogeneity thanks to gradient tracking and allows multiple local updates for saving communication. We establish the convergence rate of `PISCO` for nonconvex problems and show that `PISCO` enjoys a linear speedup in terms of the number of agents and local updates. Our numerical results highlight the superior communication efficiency of `PISCO` and its resilience to data heterogeneity and various network topologies.**

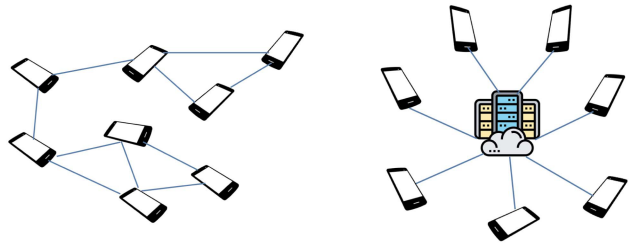
***Index Terms*—Communication efficiency, semi-decentralized networks, probabilistic communication models, local updates.**

I. Introduction

CONSIDER a networked system that n agents collectively solve the following federated or distributed optimization problem:

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\boldsymbol{x}), \text{ where } f_i(\boldsymbol{x}) := \frac{1}{m}\sum_{\boldsymbol{z}\in\mathcal{D}_i} \ell(\boldsymbol{x};\boldsymbol{z}). \tag{1}$$

Here, $\boldsymbol{x}\in\mathbb{R}^d$ denotes the optimization variable, $f_i(\boldsymbol{x})$ denotes the local and private objective function at agent i, and $f(\boldsymbol{x})$ denotes the global objective function. In addition, let $\boldsymbol{z}$ represent one data sample, $\mathcal{D}_i$ stand for the dataset with $|\mathcal{D}_i| = m$ samples at agent i, and $\ell(\boldsymbol{x};\boldsymbol{z})$ denote the empirical loss of $\boldsymbol{x}$ w.r.t. the data sample $\boldsymbol{z}$. Such problems have a wide range of applications, including but not limited to estimation in sensor networks [1], resource allocation in smart grids [2], and coordination in multi-agent systems [3].



(a) Agent-to-agent communication model with a general connected graph, where each agent only communicates with its adjacent agents.

(b) Agent-to-server communication model with a star graph, where each agent can send messages to and receive messages from the server.

Fig. 1. Two communication models for distributed ML.

In order to tackle this problem, agents have to communicate with one another for cooperation, since every agent $i\in[n]$ only has access to its own local dataset $\mathcal{D}_i$. There are two main communication protocols, consisting of agent-to-agent communication model (in decentralized ML) and agent-to-server communication model (in federated ML). Commonly, they are formulated via different network topologies [4], as shown in Fig. 1. More specifically, prior works in decentralized ML often use a general graph to capture the local communication, where every agent is only allowed to exchange information with its connected neighbors (cf. Fig. 1(a)). In federated ML, the star graph is commonly used to depict the communication between agents and the centralized coordinator (i.e., server) who can both collect information from and broadcast to each agent (cf. Fig. 1(b)).

As the network size increasingly grows, communication efficiency becomes so critical that significantly hinders both decentralized and federated ML from being applied to real-world applications. Compared with agent-to-server communication, agent-to-agent communication is much more affordable and more applicable to large-scale networks. However, without the coordination of the server, decentralized approaches may need more communication rounds to reach consensus, especially for large and sparse networks.

Given that the communication complexity depends on the trade-off between the communication rounds and the per-round cost, emerging works focus on heterogeneous communication

Received 2 May 2024; revised 6 November 2024; accepted 23 January 2025. Date of publication 6 February 2025; date of current version 25 February 2025. The work of He Wang was supported in part by the Bob Lee Gregory Fellowship at Carnegie Mellon University. This work was supported in part by ONR under Grant N00014-19-1-2404, in part by NSF under Grant CIF-2007911 Grant CNS-2148212, Grant ECCS-2318441, Grant AFRL FA8750-20-2-0504, and in part by federal agency and industry partners as specified in the Resilient & Intelligent NextG Systems (RINGS) program. An earlier version of this work was presented at the 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2024 [DOI: 10.1109/ICASSP48485.2024.10447382]. The associate editor coordinating the review of this article and approving it for publication was Dr. Chenglin Li. *(Corresponding author: He Wang.)*

The authors are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: hew2@andrew.cmu.edu; yuejiec@andrew.cmu.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TSIPN.2025.3539004, provided by the authors.

Digital Object Identifier 10.1109/TSIPN.2025.3539004

TABLE I
COMPARISON OF OURS AND OTHER SEMI-DECENTRALIZED ALGORITHMS IN NONCONVEX OPTIMIZATION USING THE SAME BATCH SIZE, REGARDING THE CONVERGENCE RATE, ALGORITHM DESIGN AND DATA HETEROGENEITY ASSUMPTIONS

Algorithm	Convergence rate [1]		Bounded data heterogeneity assumption	Accessibility of server	Multiple local updates
	Mini-batch	Large-batch			
Gossip-PGA [5]	$O\left(\frac{1}{\sqrt{nK}}\right)$	$O\left(\left(\frac{G}{K}\right)^{\frac{2}{3}}\right)$	✓	every H rounds	✗
HL-SGD [6]	$O\left(\frac{1}{\sqrt{nK}}\right)$	$O\left(\left(\frac{G_1+G_2}{K}\right)^{\frac{2}{3}}\right)$	✓	every H rounds	✗
This paper	$O\left(\frac{1}{\sqrt{nT_oK}}\right)$	$O\left(\frac{1}{K}\right)$	✗	w.p. p	✓

Here, K is the number of communication rounds, n is the number of agents, T_o is the number of multiple local updates within a single gossip/global communication round, G is the quantity in bounded dissimilarity between local objective and the global objective, G_1 is the quantity in bounded intra-cluster dissimilarity as [6, Assumption 4], G_2 is the quantity in bounded inter-cluster dissimilarity as [6, Assumption 5].

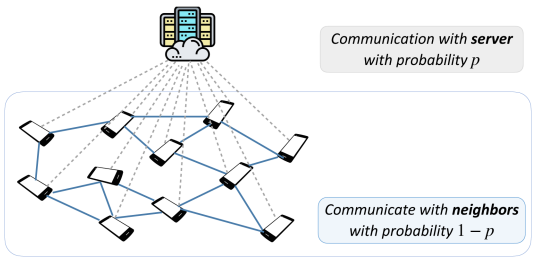


Fig. 2. The semi-decentralized communication protocol, where the server can be accessed with probability p and agents can communicate with their neighbors whenever the server is not available. Here, dotted lines represents the agent-to-server communication, while the solid ones are for agent-to-agent communication.

over *semi-decentralized networks*, to gain the best from both agent-to-agent and agent-to-server communication [5], [6], [7]. Such semi-decentralized networks—consisting of a centralized server and a network of agents—widely exist in many applications, such as autonomous vehicles [8], energy systems [9] and ML systems [10]. It has been observed that heterogeneous communication largely alleviates the heavy network dependence of distributed learning and tackles the communication bottleneck of the server [5]. However, to the best of our knowledge, all of them rely on the assumption of bounded data dissimilarity across agents and a complete characterization of the convergence behavior with respect to the network heterogeneity is still lacking. More detailed discussions on communication-saving strategies and semi-decentralized approaches are provided in Section I-B.

A. Our Contributions

To fill the void, we propose a communication-efficient algorithm called PISCO, which incorporates *gradient-tracking techniques* [11] and *multiple local updates* [12] for solving federated nonconvex optimization over semi-decentralized networks modeled by a probabilistic connection model (shown in Fig. 2). Such a semi-decentralized communication model (with local updates) allows PISCO to be viewed as a special form of gradient-tracking-based algorithms with time-varying networks. However, existing convergence guarantees for nonconvex optimization [13], [14], [15], cannot fully characterize the benefits of agent-to-server communication and multiple local updates. Specifically, applying previous analyses would yield convergence results that depend on the spectral gap of the least connected network—i.e., the underlying gossip communication network—while failing to capture the value of agent-to-server communication. To quantify these benefits, our analysis is of independent interest and can be readily extended to time-varying networks. The highlights of our contributions are as follows.

1) We prove that PISCO converges at a rate of $O(1/\sqrt{nT_oK})$ for sufficiently large K, where K is the number of communication rounds and T_o is the number of local updates. Our result does not impose the strong assumptions on data heterogeneity. Moreover, increasing the number of local updates accelerates the convergence over semi-decentralized networks. See Table I for a detailed comparison with prior art.
2) We show that the communication heterogeneity offered by a semi-decentralized network largely alleviates the network dependency of communication overheads in decentralized networks via a few agent-to-server communication rounds. For large and sparse networks (i.e., the mixing rate $\lambda_w \to 0$), with a small probability $p = \Theta(\sqrt{\lambda_w})$ of agent-to-server communication, the network dependency improves from $O(\lambda_w^{-2})$ to $O(\lambda_w^{-1})$.
3) We corroborate the superior communication efficiency of PISCO through simulations on real-world datasets. The results substantiate the convergence speedup brought by multiple local updates and the robustness of PISCO to data heterogeneity and various topologies, even for locally disconnected networks.

B. Related Works

Over the past few years, distributed optimization has attracted growing attention and has been extensively explored. For the convenience of our readers, we provide a review of the most related works below.

Distributed nonconvex optimization: As the size of the networked system increases, there are considerable algorithms developed for solving distributed nonconvex optimization. Roughly speaking, they can be categorized into two classes — decentralized algorithms [13], [16], [17], [18] where agents are only allowed to exchange information with neighbors, and federated algorithms [19], [20], [21] where agents are able to

[1] Here we only present the leading term of the rate for simplicity.

communicate with the server directly. Early attempts apply (stochastic) gradient descent to distributed optimization, which performs well in practice [16], [19]. However, the dissimilarity among local objectives could degenerate the performance under heterogeneous data and thus requires additional assumptions like bounded gradient or diminishing step-sizes. To eliminate such strong assumptions, many following works [13], [17], [20] have been developed, including gradient tracking (GT) techniques. The key idea of GT is to utilize dynamic average consensus [11] for global gradient estimation, which has been incorporated with many distributed optimization algorithms to achieve faster convergence rates in nonconvex settings [22], [23], [24], [25]. Our proposed `PISCO` also takes advantage of GT to inherit its robustness against data heterogeneity.

Communication-efficient distributed ML: Communication efficiency is one of the most important bottlenecks in distributed ML. In the decentralized setting, the communication complexity largely depends on the network topology, i.e., poor connectivity slows down the information mixing and thus requires more communication rounds to consensus [4]. As for the federated ML, the communication burden of the centralized server may be unaffordable. To overcome such bottlenecks, a number of strategies are proposed for improving communication efficiency [26], including: 1) *compression methods:* compressing the information for communication [25], [27]; 2) *multiple local communication and updates:* executing multiple gossip communication [5], [28] or successive local updates within one communication round [19], [20], [29], [30]; 3) *heterogeneous communication model over semi-decentralized networks:* allowing both agent-to-agent and agent-to-server communication [5], [31], [32]; and 4) *adaptive communication strategies:* utilizing event-triggered communication mechanisms [33], [34], [35] and tailored adaptive communication topologies [36], [37], [38], [39] for saving unnecessary communications. In this paper, we aim to gain the best communication efficiency from probabilistic heterogeneous communication over semi-decentralized networks and enable multiple local updates for more communication savings.

Semi-decentralized ML: As mentioned above, semi-decentralized ML, resorting to heterogeneous communication, tackles both the communication bottleneck of the centralized server in federated ML and the heavy network dependency of decentralized ML. We mainly summarize the works in nonconvex setting that are mostly related to this paper, while referring readers to [5], [40] for the (strongly) convex setting. For nonconvex problems, Gossip-PGA [5] first integrates Gossip SGD [16] with periodical global averaging and obtains a better scalability. It shows that intermittently communicating with the server can largely alleviate the heavy dependence on the network connectivity, especially for large or sparse networks. However, the theoretical results depend on the assumption of bounded similarity between local objectives. Moreover, HL-SGD [6] extends Gossip-PGA to the hierarchical networked structure with multiple clusters, while HA-Fed [7] can be viewed as HL-SGD with momentum. Both of them enable intra-cluster gossip averaging and inter-cluster averaging, but also rely on the data heterogeneity assumptions which may be impractical in many real-world applications. Noted that all of them consider deterministic heterogeneous communication, i.e., agents/cluster can only communicate with the server every H communication rounds, but the synchronization largely depends on the availability of the server. To this end, we consider the probabilistic communication model, where agents only exchange information with the server at the probability p. Furthermore, to the best of our knowledge, none of these approaches enable multiple local updates within a single communication round, whereas ours benefits from the linear speedup provided by the local updates. More detailed comparison can be found in Table I. Note that this comparison is based on the same batch size, while [5] and [6] may use fewer batch data per round.

C. Notation

Throughout this paper, we use the lowercase and uppercase boldface letters to represent vectors and matrices, respectively. We use $\|\boldsymbol{A}\|_{\mathsf{F}}$ for the Frobenius norm of a matrix $\boldsymbol{A}$, $\|\boldsymbol{A}\|_2$ for the largest singular value of a matrix $\boldsymbol{A}$, $\|\boldsymbol{a}\|_2$ for the l_2 norm of a vector $\boldsymbol{a}$, and $\otimes$ for the Kronecker product. In addition, we use $\boldsymbol{I}_n$ for the identity matrix of dimension n, $\mathbf{1}_n$ for the all-one vector of dimension n and $\boldsymbol{O}_{d\times n}$ for the all-zero matrix of dimension $(d \times n)$. For any two real functions $f(\cdot)$ and $g(\cdot)$ defined on $\mathbb{R}^+$, $f(x) = O(g(x))$ if there exist a positive real constant M and x_0 such that $f(x) \leq Mg(x)$ for any $x \geq x_0$. Similarly, $f(x) = \Theta(g(x))$ if there exist positive real constansts M_1, M_2 and x_0 such that $M_1 g(x) \leq f(x) \leq M_2 g(x)$ for any $x \geq x_0$. Note that "$\leq$" can be interpreted in an element-wise fashion, if it is applied to vectors or matrices with the same dimension.

II. Preliminaries

A. Communication Graph and Mixing Matrix

Consider a *semi-decentralized network* that has a centralized server to coordinate all n agents and an undirected *communication graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, n\}$ denotes the set of n agents and $\mathcal{E} \subseteq \{\{i, j\} | i, j \in \mathcal{V}\}$ represents the local communication between agents. For every agent $i \in \mathcal{V}$, let $\mathcal{N}_i = \{j | \{i, j\} \in \mathcal{E}\}$ denote agent i's neighbors whom the agent i can communicate with.

Moreover, for any communication graph $\mathcal{G}$, the mixing of local communication can be formally characterized by the *mixing matrix* $\boldsymbol{W} = [w_{ij}]_{1\leq i,j\leq n}$ defined in Definition 1.

Definition 1 (Mixing matrix and mixing rate): Given an undirected communication graph $\mathcal{G}$, a nonnegative matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ is the mixing matrix, whose element $w_{ij} = 0$ if and only if $\{i, j\} \notin \mathcal{E}$ and $i \neq j$ and $\boldsymbol{W}$ is doubly stochastic, i.e., $\boldsymbol{W}\mathbf{1}_n = \mathbf{1}_n \; and \; \mathbf{1}_n^\top \boldsymbol{W} = \mathbf{1}_n^\top$. The mixing rate of $\boldsymbol{W}$ is a nonnegative constant, i.e.,

$$\lambda_w := 1 - \left\| \boldsymbol{W} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top \right\|_2^2.$$

Note that the doubly stochasticity implies $\|\boldsymbol{W}\|_2 \leq 1$ and the mixing rate $\lambda_w = 1 - \lambda^2 \in [0, 1]$, where λ denotes the second largest eigenvalue. The mixing rate can depict the connectivity of the communication graph $\mathcal{G}$, or to say, the speed of information

mixing. Mathematically,

$$\|\boldsymbol{W}\boldsymbol{x}-\overline{\boldsymbol{x}}\|_2^2 \le (1-\lambda_w)\|\boldsymbol{x}-\overline{\boldsymbol{x}}\|_2^2, \quad \forall \boldsymbol{x}\in\mathbb{R}^d,$$

where $\overline{\boldsymbol{x}} = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top \boldsymbol{x} \in \mathbb{R}^n$. In other words, a larger mixing rate λ_w indicates a better connectivity as well as a faster process of information mixing, while disconnected graphs have $\lambda_w = 0$. For example, considering a fully connected graph where every agent can communicate with each other, the mixing matrix can be defined as

$$\boldsymbol{J} := \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^\top,$$

and its mixing rate is equal to 1. Specifically, in this paper, we use $\boldsymbol{J}$ to describe the mixing of the agent-to-server communication.

B. Stochastic Gradient Methods

To improve the computational efficiency, one popular approach is to replace the full-batch gradients with stochastic gradients on the mini-batch data samples. In distributed setting, we define the local stochastic gradient for each agent $i \in [n]$ as:

$$\boldsymbol{g}_i = \frac{1}{b}\sum_{\boldsymbol{z}_i\in\mathcal{Z}_i} \nabla\ell(\boldsymbol{x_i};\boldsymbol{z}_i), \quad \forall \boldsymbol{x}_i\in\mathbb{R}^d, \tag{2}$$

where $\mathcal{Z}_i \subset \mathcal{D}_i$ denotes the sampled data batch. Here, we assume that $\mathcal{Z}_i$ is drawn i.i.d. from $\mathcal{D}_i$ with the same mini-batch size $b \le m$ for every agent $i \in [n]$ for simplicity, while using an adaptive batch size could be of interest for better controlling the variance of stochastic gradients [41]. Note that the local stochastic gradient $\boldsymbol{g}_i$ is an unbiased estimate of $\nabla f_i(\boldsymbol{x}_i)$, i.e.,

$$\mathbb{E}[\boldsymbol{g}_i] = \nabla f_i(\boldsymbol{x}_i), \quad \forall \boldsymbol{x}_i\in\mathbb{R}^d.$$

C. Gradient-Tracking Techniques

In many real-world applications, the local dataset $\mathcal{D}_i$ on every agent $i \in [n]$ may be quite different from each other, referred to as *data heterogeneity*. Accordingly, there exists some local stationary solution $\boldsymbol{x}$ satisfying $\nabla f_i(\boldsymbol{x}) = 0$ for some $i \in [n]$, but not necessarily with $\nabla f(\boldsymbol{x}) = \sum_{i=1}^n \nabla f_i(\boldsymbol{x}) = 0$. Under such circumstances, directly incorporating stochastic gradient methods with gossip or global averaging may not converge to the global stationary solution [16] without the strong assumption like bounded data dissimilarity.

To address this issue, gradient-tracking (GT) techniques [13], [42], [43] have been proposed, which utilizes gossip communication for global gradient estimation leveraging dynamic average consensus [11]. Recently, DSGT [44] incorporates GT with stochastic gradient methods for computational efficiency. The updates at the k-th iteration are defined as: every agent $i \in [n]$ updates its optimization variable $\boldsymbol{x}_i^k$ and gradient-tracking variable $\boldsymbol{y}_i^k$ by

$$\boldsymbol{x}^{k+1} = \sum_{j=1}^n w_{ij}(\boldsymbol{x}_j^k - \eta\boldsymbol{y}_j^k),$$

$$\boldsymbol{y}_i^{k+1} = \sum_{j=1}^n w_{ij}\boldsymbol{y}_j^k + \boldsymbol{g}_i^{k+1} - \boldsymbol{g}_i^k,$$

where $\eta > 0$ is the step-size and the initialization $\boldsymbol{y}_i^0 = \boldsymbol{g}_i^0$. In addition, [45] incorporates GT with variance-reduced techniques and [28] develops an approximate Newton-type methods with variance-reduced GT to further accelerate the convergence. More recent works [30], [46] prove that DSGT with multiple local updates is able to converge under high data heterogeneity in nonconvex setting.

III. Proposed PISCO Algorithm

In this section, we introduce PISCO, which exploits communication heterogeneity from the probabilistic communication model and inherits the robustness to data heterogeneity from GT. Before the depiction of PISCO, we first introduce some compact-form notations for convenience. Let the matrices $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$ and $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n] \in \mathbb{R}^{d\times n}$ represent the collection of all the optimization variables and gradient-tracking variables, respectively. We also denote the gradient of empirical loss given the sampled batch dataset $\mathcal{Z} = \{\mathcal{Z}_i\}_{i=1}^n$ as

$$\nabla\ell(\boldsymbol{X};\mathcal{Z}) = \left[\sum_{\boldsymbol{z}_1\in\mathcal{Z}_1}\nabla\ell(\boldsymbol{x}_1;\boldsymbol{z}_1), \ldots, \sum_{\boldsymbol{z}_n\in\mathcal{Z}_n}\nabla\ell(\boldsymbol{x}_n;\boldsymbol{z}_n)\right].$$

With the local stochastic gradient as (2) in hand, the distributed stochastic gradient can be represented by

$$\boldsymbol{G} = [\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_n] = \frac{1}{b}\nabla\ell(\boldsymbol{X};\mathcal{Z}).$$

Then, we are ready to describe PISCO detailed in Algorithm 1, using the above compact notations. At the beginning of the k-th communication round, PISCO maintains the model estimate $\boldsymbol{X}^k$, the global gradient estimate $\boldsymbol{Y}^k$ and the distributed stochastic gradient $\boldsymbol{G}^k$. It then boils down to the following two stages for achieving both communication efficiency and exact convergence under data heterogeneity.

- The first stage is to execute T_o local steps without any communication (cf. line 4–7). The key idea is to utilize the local computational resources to facilitate the convergence. At the beginning of the local updates, initialize the local-update variables $\boldsymbol{X}^{k+1,0} = \boldsymbol{X}^k$, $\boldsymbol{Y}^{k+1,0} = \boldsymbol{Y}^k$ and $\boldsymbol{G}^{k+1,0} = \boldsymbol{G}^k$. At the t-th local update, update $\{\boldsymbol{X}^{k+1,t}, \boldsymbol{Y}^{k+1,t}, \boldsymbol{G}^{k+1,t}\}$ via (3), maintaining the fashion of gradient-tracking techniques.
- The second stage is to perform the information exchange over the semi-decentralized network via a probabilistic communication model (cf. line 8–10), i.e., there are two possible communication schemes — agent-to-server communication with probability p and agent-to-agent communication otherwise. Different schemes correspond to different mixing matrices (cf. line 8), i.e., if agents implement the global communication, set $\boldsymbol{W}^k = \boldsymbol{J}$; otherwise, set $\boldsymbol{W}^k = \boldsymbol{W}$. Then, agents update $\{\boldsymbol{X}^{k+1}, \boldsymbol{Y}^{k+1}, \boldsymbol{G}^{k+1}\}$ via (4), using the output of local updates $\{\boldsymbol{X}^{k+1,T_o}, \boldsymbol{Y}^{k+1,T_o}, \boldsymbol{G}^{k+1,T_o}\}$ via the selected communication scheme.

Algorithm 1: PISCO for Semi-Decentralized Nonconvex Optimization.

input: $\boldsymbol{X}^0 = \boldsymbol{x}^0\mathbf{1}_n^\top$, local-update and communication step sizes η_l, η_c, number of iterations K, number of local updates T_o, mini-batch size b.
initialize: Draw the mini-batch $\mathcal{Z}^0 = \{\mathcal{Z}_i^0\}_{i=1}^n$ randomly and set $\boldsymbol{Y}^0 = \boldsymbol{G}^0 = \frac{1}{b}\nabla\ell(\boldsymbol{X}^0; \mathcal{Z}^0)$.
for $k = 0, 1, \ldots, K-1$ **do**
  Set $\boldsymbol{X}^{k+1,0} = \boldsymbol{X}^k$, $\boldsymbol{Y}^{k+1,0} = \boldsymbol{Y}^k$ and $\boldsymbol{G}^{k+1,0} = \boldsymbol{G}^k$.
  for $t = 1, 2\cdots, T_o$ **do**
    Draw the mini-batch $\mathcal{Z}^{k+1,t}$ and compute

$$\boldsymbol{X}^{k+1,t} = \boldsymbol{X}^{k+1,t-1} - \eta_l \boldsymbol{Y}^{k+1,t-1} \tag{3a}$$

$$\boldsymbol{G}^{k+1,t} = \frac{1}{b}\nabla\ell(\boldsymbol{X}^{k+1,t}; \mathcal{Z}^{k+1,t}) \tag{3b}$$

$$\boldsymbol{Y}^{k+1,t} = \boldsymbol{Y}^{k+1,t-1} + \boldsymbol{G}^{k+1,t} - \boldsymbol{G}^{k+1,t-1}. \tag{3c}$$

  end for
  Define $\boldsymbol{W}^k = \begin{cases} \boldsymbol{J} & \text{with probability } p, \\ \boldsymbol{W} & \text{otherwise.} \end{cases}$
  Draw the mini-batch $\mathcal{Z}^{k+1}$ and update

$$\boldsymbol{X}^{k+1} = \left((1-\eta_c)\boldsymbol{X}^k + \eta_c(\boldsymbol{X}^{k+1,T_o} - \eta_l \boldsymbol{Y}^{k+1,T_o})\right)\boldsymbol{W}^k \tag{4a}$$

$$\boldsymbol{G}^{k+1} = \frac{1}{b}\nabla\ell(\boldsymbol{X}^{k+1}; \mathcal{Z}^{k+1}) \tag{4b}$$

$$\boldsymbol{Y}^{k+1} = \left(\boldsymbol{Y}^{k+1,T_o} + \boldsymbol{G}^{k+1} - \boldsymbol{G}^{k+1,T_o}\right)\boldsymbol{W}^k. \tag{4c}$$

end for

IV. Theoretical Guarantees

In this section, we provide the convergence results of our PISCO under different settings: PISCO converges at a rate of $O(1/\sqrt{nT_oK})$ using mini-batch gradients and $O(1/(nK))$ with full-batch gradients.

A. Assumptions

Before proceeding to the results, we first impose the following assumptions on the network model, objective functions and data sampling.

Assumption 1 (Semi-decentralized network model): Given the undirected graph $\mathcal{G}$ and its mixing matrix $\boldsymbol{W}$ following the Definition 1, then $\boldsymbol{W}^k$ defined in Algorithm 1 satisfies

$$\mathbb{E}[\|\boldsymbol{W}^k\boldsymbol{x} - \overline{\boldsymbol{x}}\|_2^2] \le (1-\lambda_p)\|\boldsymbol{x} - \overline{\boldsymbol{x}}\|_2^2, \quad \forall \boldsymbol{x} \in \mathbb{R}^n,$$

where $\overline{\boldsymbol{x}} = \boldsymbol{J}\boldsymbol{x} \in \mathbb{R}^n$ and the expected mixing rate $\lambda_p = \lambda_w + p(1-\lambda_w) \in (0, 1]$.

Note that Assumption 1 is weaker than the connected assumption in prior semi-decentralized literatures [5], [6], [7], i.e., $\lambda_w > 0$. More specifically, Assumption 1 implies that the underlying graph can be disconnected if and only if $p > 0$. Only in the case that the centralized server is unavailable (i.e., $p = 0$), Assumption 1 presumes the connectivity of $\mathcal{G}$.

Regarding the objective functions, we assume that the optimal value $f^\star := \min_{\boldsymbol{x}} f(\boldsymbol{x})$ exists and $f^\star > -\infty$. The local objective functions $\{f_i\}_{i=1}^n$ could be nonconvex but satisfy the standard smoothness assumption provided below.

Assumption 2 (L-smooth): Each local function $f_i(\boldsymbol{x})$ is differentiable and there exists a constant L such that

$$\|\nabla f_i(\boldsymbol{x}) - \nabla f_i(\boldsymbol{y})\|_2 \le L\|\boldsymbol{x} - \boldsymbol{y}\|_2, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d.$$

In addition, we assume that the local stochastic gradient $\boldsymbol{g}_i$ is an unbiased estimate with a bounded variance, which is widely used in the literature [5], [6], [7], [25], [30], [47].

Assumption 3 (Bounded variance): For every agent $i \in [n]$, there exists a constant $\sigma \ge 0$ s.t.

$$\mathbb{E}_{\mathcal{Z}_i \sim \mathcal{D}_i}[\|\boldsymbol{g}_i - \nabla f_i(\boldsymbol{x})\|_2^2] \le \sigma^2/b, \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

Note that in the case of the full-batch gradients, i.e., the mini-batch size $b = m$, we can simply set $\sigma = 0$ and thus Assumption 3 always holds.

B. Convergence Analysis of PISCO

Now, we are ready to present our main results. First, the following theorem demonstrates that our proposed PISCO is able to converge to the neighborhood of the stationary solution to the problem (1) at the rate of $O(1/K)$ with constant step-sizes. The proof is postponed to the Appendix B.

Theorem 1 (Convergence rate): Suppose Assumption 1, 2 and 3 hold. Let $\tilde{f} = f(\overline{\boldsymbol{x}}^0) - f^\star$ and $\boldsymbol{\Phi}_y^0 = \boldsymbol{Y}^0 - \boldsymbol{Y}^0\boldsymbol{J}$. For any $\alpha \ge 0.1$ s.t. $\eta_c = \alpha\sqrt{(1+p)\lambda_p}$ and $\eta_l \le \frac{\sqrt{(1+p)\lambda_p}}{360\alpha L(T_o+1)}$, it holds that $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]$ converges at the rate of

$$\underbrace{O\left(\frac{\tilde{f}}{\eta T_o K} + \left(L^2 T_o^2 \eta_l^2 + \frac{L\eta}{n}\right)\frac{\sigma^2}{b}\right)}_{\text{terms due to SGD and local updates}} + \underbrace{O\left(\frac{(1-p)L^2T_o^2\eta^2}{(1+p)^2\lambda_p^4}\frac{\sigma^2}{b} + \frac{1}{nK}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]\right)}_{\text{terms due to decentralized overhead}}, \tag{5}$$

where the average model estimate $\overline{\boldsymbol{x}}^k = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^k \in \mathbb{R}^d$ and $\eta = \eta_c\eta_l$.

Note that the above convergence result can hold even under significant data heterogeneity across agents, since we do not assume any bounded similarity between local objectives.

Due to the existence of the variance σ^2, we fine-tune the local-update step-size to obtain the exact convergence rate with the leading term $O(1/\sqrt{nT_oK})$, based on Theorem 1. Specifically, the following corollary considers the scenarios with mini-batch gradients (i.e., $b \le O(\sigma^2 K)$ and $\sigma > 0$), while the case of large or full batch gradients (i.e., the batch size $b \ge \Theta(\sigma^2 K)$ or $\sigma = 0$) will be discussed later in Corollary 2. The proof of Corollary 1 is postponed to Appendix C.

Corollary 1 (Convergence rate with mini batch): Suppose all the conditions in Theorem 1 hold. Consider that the number of communication rounds K is sufficiently large, i.e., $K \ge \Theta(\frac{nbT_oL\tilde{f}}{\lambda_p^4\sigma^2})$, and the mini-batch

TABLE II
THE NUMBER OF THE EXPECTED AGENT-TO-SERVER/AGENT COMMUNICATION ROUNDS OF OURS AND EXISTING DECENTRALIZED AND FEDERATED ML ALGORITHMS WITH STOCHASTIC GRADIENTS AND LOCAL UPDATES, TO ACHIEVE THE ϵ-ACCURACY, WHERE ϵ IS SUFFICIENTLY SMALL

Algorithm	# Agent-to-server communication	# Agent-to-agent communication
SCAFFOLD [20]	$O\left(\frac{\sigma^2}{nT_o\epsilon^4}+\frac{1}{\epsilon^2}\right)$	0
LSGT [30]	0	$O\left(\frac{\sigma^4}{nT_o\lambda_w^8\epsilon^4}+\frac{1}{nT_o^{1/3}\lambda_w^{8/3}\epsilon^{4/3}}+\frac{1}{nT_o\epsilon^2}\right)$
Periodical-GT [46]	0	$O\left(\frac{\sigma^2}{nT_o\epsilon^4}+\frac{\sigma}{\lambda_w^2\epsilon^3}+\frac{1}{\lambda_w^2\epsilon^2}\right)$
K-GT [46]	0	$O\left(\frac{\sigma^2}{nT_o\epsilon^4}+\frac{\sigma}{\lambda_w^2\sqrt{T_o}\epsilon^3}+\frac{1}{\lambda_w^2\epsilon^2}\right)$
This paper	$O\left(\frac{p\sigma^2}{nT_o\epsilon^4}+\frac{p\sigma}{(\lambda_w+p)^2\epsilon^3}+\frac{p}{n\epsilon^2}\right)$	$O\left(\frac{(1-p)\sigma^2}{nT_o\epsilon^4}+\frac{(1-p)\sigma}{(\lambda_w+p)^2\epsilon^3}+\frac{1-p}{n\epsilon^2}\right)$

Here, n is the number of agents, T_o is the number of local updates, λ_w is the mixing rate of the underlying graph.

size $b \le O(\sigma^2 K)$, where $\sigma > 0$. If the step-sizes $\eta_c = \alpha\sqrt{1+p}\lambda_p$, $\eta_l = \frac{1}{\alpha T_o}\min\{\sqrt{\frac{n\alpha^2 bT_o\tilde{f}}{\eta_c^2 L\sigma^2 K}}, \sqrt[3]{\frac{\eta_c b\tilde{f}}{\alpha L^2\sigma^2 K}}\}$. Then, $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]$ can converge at the rate of

$$O\left(\left(\frac{L\sigma^2}{nT_obK}\right)^{\frac{1}{2}}+\left(\frac{L\sigma}{\lambda_p^2\sqrt{b}K}\right)^{\frac{2}{3}}+\frac{1}{nK}\right).$$

From Corollary 1, PISCO can achieve the ϵ-*accuracy*, i.e., $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2] \le \epsilon^2$ after

$$O\left(\frac{L\sigma^2}{nT_ob\epsilon^4}+\frac{L\sigma}{\lambda_p^2\sqrt{b}\epsilon^3}+\frac{1}{n\epsilon^2}\right)$$

communication rounds. Notice that if K is sufficiently large and the first term $(\frac{L\sigma^2}{nT_obK})^{\frac{1}{2}}$ correspondingly becomes dominant, increasing the number of agents n or the number of local updates T_o can accelerate the convergence. Such a linear speedup matches the findings in the special cases of semi-decentralized ML, i.e., decentralized setting [30], [46] when $p = 0$ and federated setting [20] when $p = 1$.

In fact, PISCO can be generalized to the decentralized case and federated case by setting $p = 0$ and $p = 1$ respectively, while maintaining comparable convergence guarantees.

Remark 1 (Decentralized case): When $p = 0$, Algorithm 1 becomes fully decentralized, i.e., agents only perform local communication. Then, the communication complexity to achieve ϵ-accuracy becomes

$$O\left(\frac{L\sigma^2}{nT_ob\epsilon^4}+\frac{L\sigma}{\lambda_w^2\sqrt{b}\epsilon^3}+\frac{1}{n\epsilon^2}\right),$$

which is better than the rate of Periodical-GT in [46] and LSGT [30], since the network dependency is $O(1/\lambda_w^2)$ and only appears in the second term (see Table II). The second term is slightly worse than K-GT, the variance-reduced Periodical-GT [46], since it corrects the descent direction with the average of T_o local updates instead of the last local update at communication. However, they require that the initial local correction variables are settled in a centralized way. Combining our analysis with such a variance-reduction method while avoiding the centralized initialization would be a promising future direction of this paper.

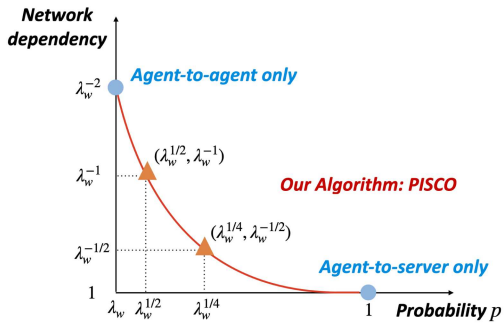


Fig. 3. The network dependency of PISCO regarding agent-to-server communication probability p.

Remark 2 (Federated case): When $p = 1$, every agent can communicate with the server directly and thus PISCO performs in the federated fashion at any iteration $k \ge 1$. Then, the communication complexity becomes

$$O\left(\frac{L\sigma^2}{nT_ob\epsilon^4}+\frac{L\sigma}{\sqrt{b}\epsilon^3}+\frac{1}{n\epsilon^2}\right),$$

where the leading term is the same as that of SCAFFOLD in [20] with the linear speedup in terms of the network size n and the number of local updates T_o.

Moreover, the highlight of our work is to fill the void of semi-decentralized ML with the probabilistic communication model and gain the best communication efficiency from both agent-to-agent communication and agent-to-server communication, as shown in Table II. In addition, PISCO is able to improve the network dependency of the communication overheads from $O(\lambda_w^{-2})$ [46] to $O(\lambda_p^{-2})$, where the trade-off between the communication probability and the network dependency is illustrated in Fig. 3. The flexible heterogeneous communication brings the superior communication efficiency of PISCO in both well-connected and poorly-connected networks.

Remark 3 (For well-connected networks): As gossip communication is efficient to mix information for well-connected networks, PISCO is able to achieve a comparable convergence rate with much fewer agent-to-server communication rounds compared with using only agent-to-server communication. Therefore, our PISCO can significantly reduce the communication

costs for well-connected networks whenever local agent-to-agent communications are inexpensive.

Remark 4 (For poorly-connected networks): When $\lambda_w \to 0$, performing agent-to-agent only communication often results in a large number of communication rounds and prohibitive communication costs. As shown in Fig. 3, with any probability $p \geq \lambda_w$, the network dependency can be reduced to $O(p^{-2})$. More specifically, even a small agent-to-server probability $p = \Theta(\sqrt{\lambda_w})$ can significantly improve the network dependency from $O(\lambda_w^{-2})$ to $O(\lambda_w^{-1})$. Take the large-scale path graph as an example, where the mixing rate λ_w scales on the order of $O(1/n^2)$ [4]. Our PISCO with $p = \Theta(1/n)$ can improve the network dependency from $O(n^4)$ to $O(n^2)$. Moreover, if $p = \Theta(1)$, the communication complexity can be network-independent like Gossip-PGA [5], but our theoretical analysis does not require the additional assumption of the bounded dissimilarity between local objectives.

In many real-world scenarios, it is also popular to choose large mini-batch size b to guarantee the exact convergence to the stationary point. As the terms related to the variance σ^2 on the right hand side of (5) scale on the order of $O(\sigma^2/b)$, if we choose a large enough mini-batch size $b \geq \Theta(\sigma^2/\epsilon^2)$, the following desirable result holds.

Corollary 2 (Communication complexity with large batch): Suppose all the conditions in Theorem 1 holds. If the batch size b is sufficiently large, i.e., $b \geq \Theta(\frac{\sigma^2}{\epsilon^2})$, it holds $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\bar{\boldsymbol{x}}^k)\|_2^2] \leq \epsilon^2$ after

$$O\left(\frac{L}{(1+p)\lambda_p^2\epsilon^2} + \frac{1}{\epsilon^2}\right)$$

communication rounds. In addition, if the mini-batch size $b = m$, i.e., we take the full-batch gradient, the comunication complexity will be improved to

$$O\left(\frac{L}{(1+p)\lambda_p^2\epsilon^2} + \frac{1}{n\epsilon^2}\right).$$

Note that Corollary 2 also matches the result in decentralized setting [29] and federated setting [20], by setting $p = 0$ and $p = 1$ respectively.

V. Numerical Experiments

In this section, we present the numerical performance of PISCO on real-world datasets, to substantiate its superior performance in terms of communication efficiency and robustness to various topologies and data heterogeneity.

A. Logistic Regression With Nonconvex Regularization

To investigate communication efficiency of PISCO, we conduct experiments on logistic regression with a nonconvex regularization term [48] using the a9a dataset [49]. Given the model parameter $\boldsymbol{x}$ and data sample $\boldsymbol{z} = (\boldsymbol{a}, y)$, the empirical loss $\ell(\boldsymbol{x}; \boldsymbol{z})$ is defined as:

$$\ell(\boldsymbol{x}; \boldsymbol{z}) = \log\left(1 + \exp(-y\boldsymbol{a}^\top \boldsymbol{x})\right) + \rho\sum_{l=1}^{d}\frac{\boldsymbol{x}(l)^2}{1+\boldsymbol{x}(l)^2},$$

where $\boldsymbol{a} \in \mathbb{R}^d$ is the feature vector, $y \in \{-1, 1\}$ is the corresponding label, the regularizer coefficient ρ is set as 0.01, and $\boldsymbol{x}(l)$ denotes the l-th coordinate of $\boldsymbol{x}$.

In this subsection, we consider a ring topology with $n = 10$ agents and evenly partition the sorted a9a dataset to 10 agents to augment the data heterogeneity. Roughly speaking, every agent will receive $m = 3256$ training samples of dimension $d = 124$, where 5 agents will receive data with label 1 and the others will receive data with label 0. Regarding the mixing matrix, we follow the symmetric FDLA matrix [50] to aggregate information among neighbors. In addition, we set the batch size $b = 256$ for the following experiments. To reduce the impact of randomness, we run every experiment with 5 different seeds and show the average results.

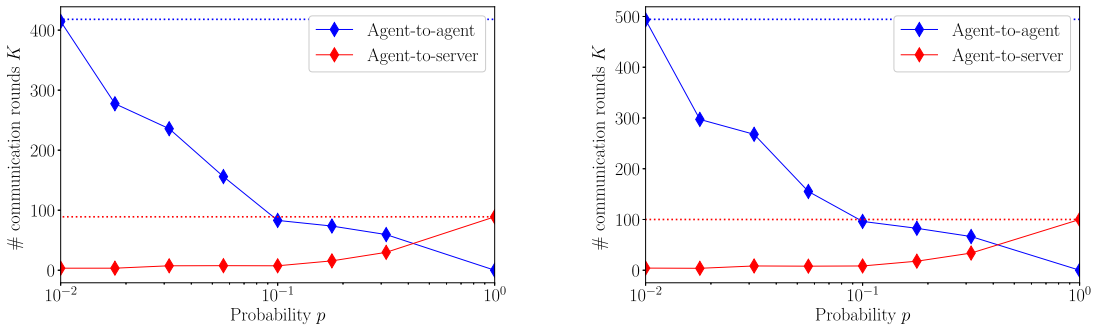
The impact of different agent-to-server probabilities: First, we study the influence of the agent-to-server communication probability p on the training and test performance. To this end, we vary the probability p from $\{1, 1/10^{0.5}, 1/10^{0.75}, 1/10, 1/10^{1.25}, 1/10^{1.5}, 1/10^{1.75}, 1/10^2, 0\}$ and present the number of communication rounds of PISCO with different p to achieve 0.05 training accuracy (i.e., $\frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f(\bar{\boldsymbol{x}}^k)\|_2^2 \leq 0.05$) and 80% test accuracy ($\geq 95\%$ of the peak accuracy within 1000 communication rounds), in Fig. 4.

From Fig. 4, we observe that just a small agent-to-server probability (e.g., $p \leq 0.1$) can considerably reduce the number of communication rounds required to attain a specific accuracy during both training and testing phases. For instance, PISCO with $p = 10^{-1.25} \approx 0.06$ can reduce agent-to-agent communication rounds by 60%, with several agent-to-server communication rounds. Moreover, even if the server is more accessible (e.g., $p \geq 0.1$), increasing the agent-to-server communication probability p might not further save the total communication rounds. This indicates that not all costly communications between agents and the server are crucial for accelerating the convergence compared with decentralized methods. Therefore, by leveraging heterogeneous communication, we can reduce the average per-round communication expense while preserving a comparable rate of convergence.

The speedup of multiple local updates: To verify the speedup of multiple local updates, we plot the training accuracy and test accuracy of PISCO with different numbers of local updates $T_o = 1$ (cf. Fig. 5(a)) and $T_o = 10$ (cf. Fig. 5(b)). In both cases, we vary the probability $p \in \{1, 10^{-0.5}, 10^{-1}, 0\}$. It is worth noting that with only $p = 0.1$ or $p = 10^{-0.5}$, PISCO already achieves almost the same performance as PISCO with $p = 1$. Comparing Fig. 5(b) with Fig. 5(a), we can clearly observe the speedup brought by multiple local updates for different probabilities. For example, for PISCO with $p = 0.1$, the number of communication rounds required to attain 0.05 training accuracy or 80% testing accuracy decreases roughly by 50% if we increase T_o from 1 to 10.

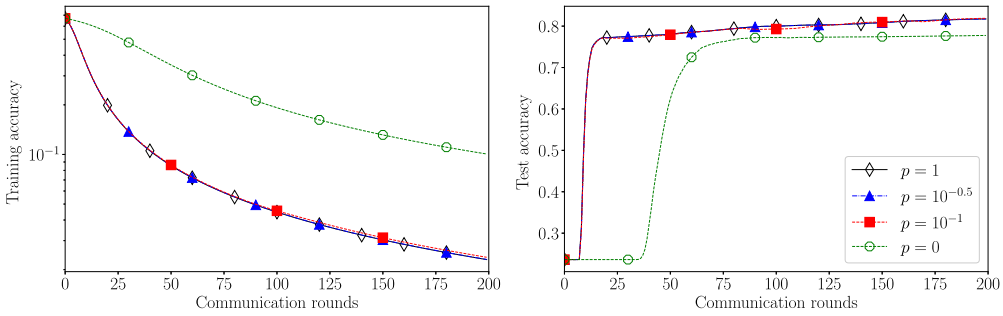
B. Neural Network Training

Further, we run the single hidden-layer neural network training with 32 hidden neurons on the MNIST dataset [51]. More specifically, we use the sigmoid and softmax function as the

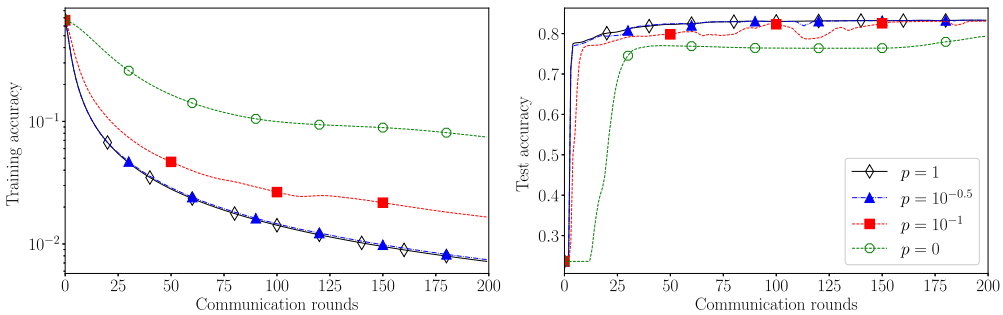


(a) Training accuracy $\frac{1}{K+1}\sum_{k=0}^{K}\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2 \leq 0.05$. (b) Test accuracy $\leq 80\%$.

Fig. 4. The number of agent-to-agent and agent-to-server communication rounds required to achieve 0.05% training accuracy (the left panel) and 80% test accuracy (the right panel) for PISCO with $T_o = 1$ and different $p \in \{0, 10^{-2}, 10^{-1.75}, 10^{-1.5}, 10^{-1.25}, 10^{-1}, 10^{-0.75}, 10^{-0.5}, 1\}$. Here, the blue (red) dotted line represents the number of agent-to-agent (agent-to-server) communication rounds that PISCO with $p = 0$ (with $p = 1$) requires.



(a) The number of local updates $T_o = 1$.



(b) The number of local updates $T_o = 10$.

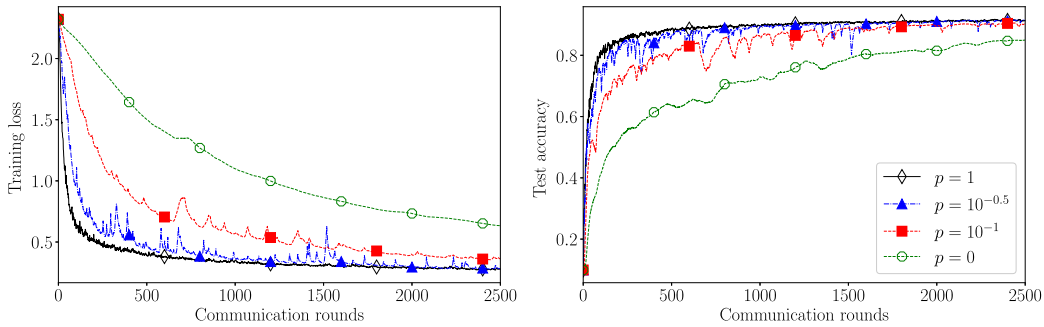
Fig. 5. The training accuracy (left two panels) and testing accuracy (right two panels) against communication rounds with different probabilities $p = 1, 10^{-0.5}, 10^{-1}, 0$ and different number of local updates $T_o = 1, 10$, over a ring topology for logistic regression with a nonconvex regularizer on the sorted a9a dataset.

activation function, where the empirical loss w.r.t. the training parameter $\boldsymbol{x} = \text{vec}(\boldsymbol{W}_1, \boldsymbol{c}_1, \boldsymbol{W}_2, \boldsymbol{c}_2)$ and the sample $\boldsymbol{z} = (\boldsymbol{a}, y)$ is defined using the cross entropy loss as:

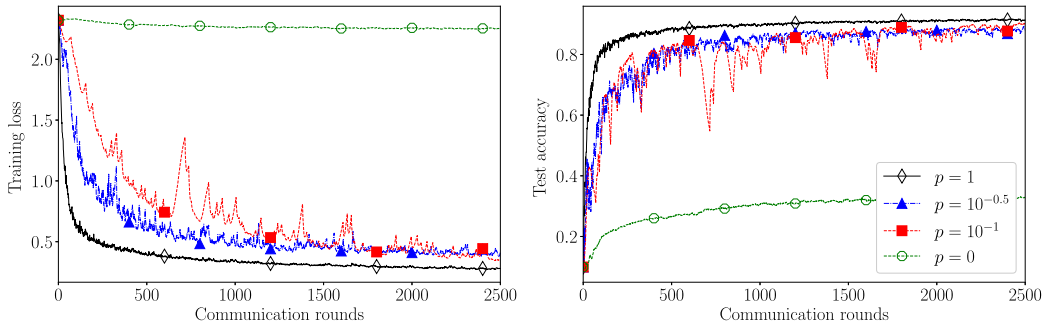
$$\texttt{CrossEntropy}(\texttt{softmax}(\boldsymbol{W}_2\,\texttt{sigmoid}(\boldsymbol{W}_1\boldsymbol{a}+\boldsymbol{c}_1)+\boldsymbol{c}_2), y),$$

where the training weights $\boldsymbol{W}_1 \in \mathbb{R}^{32\times 784}$, $\boldsymbol{W}_2 \in \mathbb{R}^{10\times 32}$, $\boldsymbol{c}_1 \in \mathbb{R}^{32}$, and $\boldsymbol{c}_2 \in \mathbb{R}^{10}$.

To verify the robustness of PISCO to diverse topologies, we consider a well-connected Erdős-Rényi topology with a connectivity probability of 0.3 (corresponding to $\lambda_w = 0.38$) and a disconnected Erdős-Rényi topology with a connectivity probability of 0.1 (corresponding to $\lambda_w = 0$). To simulate the highly data-heterogeneous scenario, we evenly split the sorted MNIST dataset to $n = 10$ agents, where agent $i \in [n]$ will receive the training data associated with label i. Moreover, we set the batch size $b = 100$, the number of local updates $T_o = 10$ and the agent-to-server communication probability $p \in \{1, 1/\sqrt{n}, 1/n, 0\} = \{1, 10^{-0.5}, 10^{-1}, 0\}$. To reduce the impact of randomness, we run every experiment with 3 different seeds and show the average results in Fig. 6.



(a) A connected Erdős-Rényi graph with 0.3 connectivity probability.



(b) A disconnected Erdős-Rényi graph with 0.1 connectivity probability.

Fig. 6. The training loss (the left two panels) and testing accuracy (the right two panels) against communication rounds with different probabilities $p = 1, 10^{-0.5}, 10^{-1}, 0$ and the number of local updates $T_o = 10$ over both well-connected and disconnected Erdős-Rényi graphs for 1-hidden-layer network training on the sorted `MNIST` dataset.

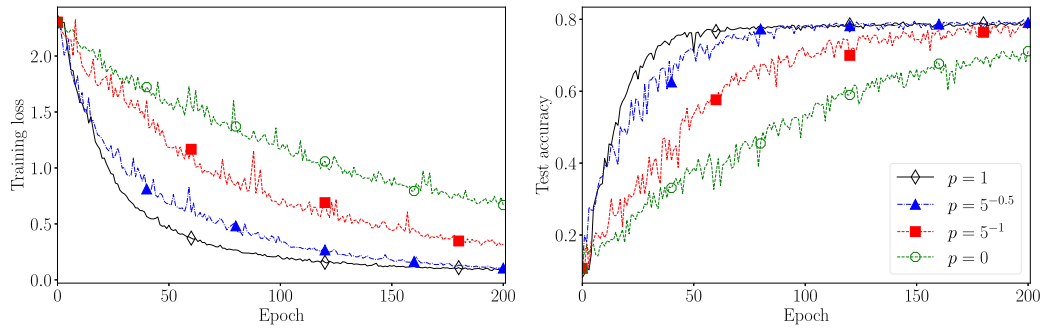


Fig. 7. The training loss and testing accuracy across epochs with different probabilities $p = 1, 1/\sqrt{5}, 0.2, 0$ and the number of local updates $T_o = 4$, over a ring topology for CNN training on the sorted `CIFAR10` dataset.

In Fig. 6, our `PISCO` shows impressive robustness to high data heterogeneity and different topologies, including the well-connected network (cf. Fig 6(a)) and the disconnected network (cf. Fig. 6(b)). By comparing Fig. 6(a) with Fig. 6(b), we observe that better connectivity makes gossip communication sufficiently efficient to mix information. As a result, heterogeneous communication with a smaller p can attain a comparable performance to that of `PISCO` with $p = 1$ in Fig. 6(a). Notice that the performance of `PISCO` with no agent-to-server communication degenerates remarkably when the network is disconnected. In contrast, semi-decentralized `PISCO` (i.e., $0 < p < 1$) maintains performance levels similar to `PISCO` with $p = 1$. It illustrates that a few number of agent-to-server communication rounds can largely mitigate the impact of the network connectivity, even for disconnected graphs.

We also evaluate the performance of `PISCO` by training a convolutional neural network (CNN) on the unshuffled `CIFAR10` dataset [52]. The network architecture includes three sequential CNN modules, each containing two 2D convolutional layers with ReLU activation, followed by max pooling (kernel size 2, stride 2) and dropout (rate 0.2) for regularization. Specifically, in the first module, the initial convolutional layer transforms the input from 3 to 32 channels, and the second convolutional layer maintains 32 channels; the second module follows this pattern,

mapping 32 to 64 channels; and the third similarly increases from 64 to 128 channels. After feature extraction, the fully connected layers process the 2048 flattened output, first mapping it to 128 features with ReLU activation and dropout, and then to 10 outputs for classification. We use a ring topology with $n = 5$ agent and set the batch size $b = 20$ and the number of local updates $T_o = 4$. To introduce data heterogeneity, we split the sorted `CIFAR10` dataset across the 5 agents, so that each agent $i \in [n]$ obtains training data with label i and $i + 5$.

In Fig. 7, we illustrate the effectiveness of `PISCO`, in terms of training loss and test accuracy across epochs. We observe that, due to sparse agent-to-agent communication in the ring topology and extremely high data heterogeneity, `PISCO` with $p = 0$ converges more slowly than `PISCO` with $p > 0$. Notably, `PISCO` with $p = 1/\sqrt{5}$ achieves performance comparable to `PISCO` with $p = 1$, demonstrating the efficiency of the heterogeneous communication protocol in reducing costly agent-to-server communications.

VI. Conclusion

In this paper, we develop a communication-efficient algorithm called `PISCO` for solving federated nonconvex optimization over semi-decentralized networks, which enjoys the linear speedup of local updates and addresses data dissimilarity without any additional assumptions. By leveraging the heterogeneous communication model, `PISCO` largely reduces communication overheads in terms of the network dependency with a few agent-to-server communication rounds, particularly evident in poorly-connected networks. Both theoretical guarantees and empirical experiments underscore `PISCO`'s outstanding communication efficiency and robustness to data heterogeneity and various network topologies.

In the future, it will be of interest to incorporate variance reduction techniques [28], [41] into the algorithm design, apply communication compression [25] to further reduce the per-round communication costs, and enable varying agent-to-server communication probabilities [53], allowing for personalized and heterogeneous communication strategies for each agent.

Appendix A
Technical Lemmas

This section establishes several critical lemmas which will be used in the proof of Theorem 1, whose proofs are delegated to the supplemental materials. Let $\eta = \eta_c \eta_l$.

To begin with, the following lemma shows that $\overline{\boldsymbol{Y}}^k = \boldsymbol{Y}^k \boldsymbol{J}$ is able to track the average of local stochastic gradients, i.e., $\overline{\boldsymbol{G}}^k = \boldsymbol{G}^k \boldsymbol{J} = (\frac{1}{n}\sum_{i=1}^n \boldsymbol{g}_i^k)\mathbf{1}_n^\top$. We define the average model estimate as $\overline{\boldsymbol{x}}^k = \frac{1}{n}\sum_{i=1}^n \boldsymbol{x}_i^k \in \mathbb{R}^d$ and $\overline{\boldsymbol{X}}^k = \boldsymbol{X}^k \boldsymbol{J}$.

Lemma 1 (Gradient tracking property): Suppose Assumption 1 holds. Then for any $k \in \mathbb{N}$,

$$\overline{\boldsymbol{Y}}^k = \overline{\boldsymbol{G}}^k.$$

In addition, if Assumption 2 and 3 hold, we have

$$\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2] \le \frac{3\sigma^2}{b} + 3L^2\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2] + 3n\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2], \quad (6)$$

where $\boldsymbol{\Phi}_x^k = \boldsymbol{X}^k - \overline{\boldsymbol{X}}^k$ is the consensus error at iteration k.

The following two auxiliary lemmas bound the progress improvement between the successive iterates and their averages. Similar to $\boldsymbol{\Phi}_x^k = \boldsymbol{X}^k - \overline{\boldsymbol{X}}^k$, we also use $\boldsymbol{\Phi}_y^k = \boldsymbol{Y}^k - \overline{\boldsymbol{Y}}^k$ to represent the tracking error at the k-th iteration. As for the local updates, we define t-th local-update consensus error as $\boldsymbol{\Phi}_x^{k,t} = \boldsymbol{X}^{k+1,t} - \overline{\boldsymbol{X}}^k$ and tracking error as $\boldsymbol{\Phi}_y^{k,t} = \boldsymbol{Y}^{k+1,t} - \overline{\boldsymbol{Y}}^k$.

Lemma 2 (Progress improvement between successive iterates): Suppose Assumption 1, 2 and 3 hold. Then, we have

$$\begin{aligned}
&\mathbb{E}[\|\boldsymbol{X}^k - \boldsymbol{X}^{k-1}\|_{\mathsf{F}}^2] \\
&\le 12\left(1 + 2T_o^2L^2\eta^2\right)\mathbb{E}[\|\boldsymbol{\Phi}_x^{k-1}\|_{\mathsf{F}}^2] \\
&+ 6(1-p)\lambda^2(T_o+1)^2\eta^2\mathbb{E}[\|\boldsymbol{\Phi}_y^{k-1}\|_{\mathsf{F}}^2] + \frac{48nT_o^2\eta^2\sigma^2}{b} \\
&+ 24T_oL^2\eta^2\sum_{t=1}^{T_o}\mathbb{E}[\|\boldsymbol{\Phi}_x^{k-1,t}\|_{\mathsf{F}}^2] + 3(T_o+1)^2\eta^2\mathbb{E}[\|\overline{\boldsymbol{Y}}^{k-1}\|_{\mathsf{F}}^2].
\end{aligned}$$

Lemma 3 (Progress improvement between the averages): Suppose Assumption 1, 2 and 3 hold. Then, we have

$$\begin{aligned}
\mathbb{E}\left[\left\|\overline{\boldsymbol{X}}^{k+1} - \overline{\boldsymbol{X}}^k\right\|_{\mathsf{F}}^2\right] &\le \frac{3T_o\eta^2\sigma^2}{b} + 3T_oL^2\eta^2\sum_{t=0}^{T_o}\mathbb{E}\left[\left\|\boldsymbol{\Phi}_x^{k,t}\right\|_{\mathsf{F}}^2\right] \\
&+ 3nT_o^2\eta^2\mathbb{E}\left[\left\|\nabla f(\overline{\boldsymbol{x}}^k)\right\|_2^2\right].
\end{aligned}$$

Next, we present the following lemma for bounding the accumulated consensus errors for local updates, in order to control the consensus and tracking errors at every iteration.

Lemma 4 (Accumulated consensus drift for local updates): Suppose Assumption 2 and 3 hold. If $\eta_l \le \frac{1}{8L(T_o+1)}$, we have

$$\begin{aligned}
\sum_{t=1}^{T_o}\mathbb{E}[\|\boldsymbol{\Phi}_x^{k,t}\|_{\mathsf{F}}^2] &\le 9T_o\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2] + 8\eta_l^2(T_o+1)^3\mathbb{E}[\|\boldsymbol{\Phi}_y^k\|_{\mathsf{F}}^2] \\
&+ \frac{64n\eta_l^2T_o^3\sigma^2}{b} + 3\eta_l^2T_o(T_o+1)^2\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2].
\end{aligned}$$

With Lemma 4 in hand, we are ready to bound the consensus error $\mathbb{E}[\|\boldsymbol{\Phi}_x\|_{\mathsf{F}}^2]$ and the tracking error $\mathbb{E}[\|\boldsymbol{\Phi}_y\|_{\mathsf{F}}^2]$, respectively.

Lemma 5 (Consensus error for communication updates): Suppose Assumption 1, 2 and 3 hold. If $\eta_l \le \frac{1}{8L(T_o+1)}$ and $\eta \le \frac{\lambda_p}{80L(T_o+1)}$, we have

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2] < (1-p)\Bigg[&\frac{1+(1+p)\lambda^2}{2}\mathbb{E}[\|\boldsymbol{\Phi}_x^{k-1}\|_{\mathsf{F}}^2] \\
&+ \frac{40\lambda^2}{\lambda_p}(T_o+1)^2\eta^2\mathbb{E}[\|\boldsymbol{\Phi}_y^{k-1}\|_{\mathsf{F}}^2] \\
&+ \frac{240\lambda^2L^2(T_o+1)^4\eta^2\eta_l^2}{\lambda_p}\mathbb{E}[\|\overline{\boldsymbol{Y}}^{k-1}\|_{\mathsf{F}}^2] \\
&+ \frac{320\lambda^2n(T_o+1)^2\eta^2}{\lambda_p}\frac{\sigma^2}{b}\Bigg].
\end{aligned}$$

Lemma 6 (Tracking error for communication updates): Suppose Assumption 1,2 and 3 hold. If $\eta_l \le \frac{1}{8L(T_o+1)}$ and

$\eta \le \frac{\lambda_p}{80L(T_o+1)}$, we have

$$\begin{aligned}
&\mathbb{E}[\|\mathbf{\Phi}_y^k\|_{\mathsf{F}}^2] \\
&\le (1-p)\lambda^2\left[\frac{1+(1+p)}{2}\mathbb{E}[\|\mathbf{\Phi}_y^{k-1}\|_{\mathsf{F}}^2]+\frac{400}{\lambda_p}L^2\mathbb{E}[\|\mathbf{\Phi}_x^{k-1}\|_{\mathsf{F}}^2]\right] \\
&\quad+\frac{(1-p)\lambda^2}{\lambda_p}\left[125L^2\eta^2(T_o+1)^2\mathbb{E}[\|\overline{\boldsymbol{Y}}^{k-1}\|_{\mathsf{F}}^2]+180n\frac{\sigma^2}{b}\right].
\end{aligned}$$

Finally, we establish the descent lemma for PISCO.

Lemma 7 (Descent lemma): Suppose Assumption 1, 2 and 3 hold. If $\eta_l \le \frac{1}{8L(T_o+1)}$ and $\eta \le \frac{1}{6L(T_o+1)}$, we have

$$\begin{aligned}
&\mathbb{E}[f(\overline{\boldsymbol{x}}^{k+1})]-\mathbb{E}[f(\overline{\boldsymbol{x}}^k)] \\
&\le -\frac{\eta(T_o+1)}{4}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]+\frac{3L^2(T_o+1)^3\eta\eta_l^2}{n}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2] \\
&\quad+\frac{10(T_o+1)L^2\eta}{n}\left(\mathbb{E}[\|\mathbf{\Phi}_x^k\|_{\mathsf{F}}^2]+(T_o+1)^2\eta_l^2\mathbb{E}[\|\mathbf{\Phi}_y^k\|_{\mathsf{F}}^2]\right) \\
&\quad+\left(64L^2T_o^3\eta\eta_l^2+\frac{3LT_o\eta^2}{2n}\right)\frac{\sigma^2}{b}, \qquad (7)
\end{aligned}$$

for any $k \ge 0$.

Appendix B
Proof of Theorem 1

From the descent lemma, i.e., Lemma 7, summing (7) from $k=0$ to $k=K-1$ gives

$$\begin{aligned}
&\frac{\eta(T_o+1)}{4}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2] \\
&\le \tilde{f}+\frac{3L^2(T_o+1)^3\eta\eta_l^2}{n}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2] \\
&\quad+\frac{10(T_o+1)L^2\eta}{n}\sum_{k=0}^{K-1}\left(\mathbb{E}[\|\mathbf{\Phi}_x^k\|_{\mathsf{F}}^2]+(T_o+1)^2\eta_l^2\mathbb{E}[\|\mathbf{\Phi}_y^k\|_{\mathsf{F}}^2]\right) \\
&\quad+\left(64L^2T_o^3\eta\eta_l^2+\frac{3LT_o\eta^2}{2n}\right)\frac{K\sigma^2}{b}, \qquad (8)
\end{aligned}$$

where $\tilde{f}=f(\overline{\boldsymbol{x}}^0)-f^*$.

To show the convergence of $\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]$, we need to bound right hand side of (8). To this end, we first formulate the dynamics of consensus errors and tracking errors. For any $k>0$, let

$$\mathbf{\Phi}^k \triangleq \begin{bmatrix}\mathbb{E}[\|\mathbf{\Phi}_x^k\|_{\mathsf{F}}^2]\\ \mathbb{E}[\|\mathbf{\Phi}_y^k\|_{\mathsf{F}}^2]\end{bmatrix} \quad \text{and} \quad \boldsymbol{e}^k \triangleq \begin{bmatrix}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2]\\ \frac{\sigma^2}{b}\end{bmatrix}.$$

Assuming that $\eta \le \frac{\lambda_p}{80L(T_o+1)}$ and $\eta_l \le \frac{1}{8L(T_o+1)}$, we can formulate the dynamics from Lemma 5 and Lemma 6,

$$\mathbf{\Phi}^{k+1} \le (1-p)\boldsymbol{A}\mathbf{\Phi}^k+\frac{(1-p)\lambda^2}{\lambda_p}\boldsymbol{B}\boldsymbol{e}^k,$$

where

$$\boldsymbol{A}=\begin{bmatrix}\frac{1+(1+p)\lambda^2}{2} & \frac{40\lambda^2(T_o+1)^2\eta^2}{\lambda_p}\\ \frac{400\lambda^2L^2}{\lambda_p} & \frac{1+(1+p)\lambda^2}{2}\end{bmatrix}, \qquad (9)$$

$$\boldsymbol{B}=\begin{bmatrix}240L^2(T_o+1)^4\eta^2\eta_l^2 & 320n(T_o+1)^2\eta^2\\ 125L^2(T_o+1)^2\eta^2 & 180n\end{bmatrix}. \qquad (10)$$

By telescoping, we have

$$\mathbf{\Phi}^k \le (1-p)^k\boldsymbol{A}^k\mathbf{\Phi}^0+\sum_{t=0}^{k-1}((1-p)\boldsymbol{A})^t\frac{(1-p)\lambda^2}{\lambda_p}\boldsymbol{B}\boldsymbol{e}^{k-1-t}. \qquad (11)$$

Summing (11) from $k=0$ to K gives

$$\begin{aligned}
\sum_{k=0}^{K}\mathbf{\Phi}^k &\le \sum_{k=0}^{K}(1-p)^k\boldsymbol{A}^k\mathbf{\Phi}^0 \\
&\quad+\sum_{k=0}^{K}\sum_{t=0}^{k-1}((1-p)\boldsymbol{A})^t\frac{(1-p)\lambda^2}{\lambda_p}\boldsymbol{B}\boldsymbol{e}^{k-1-t} \\
&\le \left(\sum_{k=0}^{\infty}(1-p)^k\boldsymbol{A}^k\right)\mathbf{\Phi}^0 \\
&\quad+\left(\sum_{k=0}^{\infty}((1-p)\boldsymbol{A})^k\right)\sum_{k=0}^{K-1}\frac{(1-p)\lambda^2}{\lambda_p}\boldsymbol{B}\boldsymbol{e}^k, \qquad (12)
\end{aligned}$$

where we define $0^0=1$.

To control $\sum_{k=0}^{\infty}(1-p)^k\boldsymbol{A}^k$ in (12), we then establish the following lemma implying that $\boldsymbol{I}-(1-p)\boldsymbol{A}$ is invertible, where the proof is delegated to the supplemental materials.

Lemma 8 (The spectral radius of $\boldsymbol{A}$): If $\eta \le \frac{(1+p)\lambda_p^2}{80\sqrt{10}(T_o+1)L}$,

$$\rho\left((1-p)\boldsymbol{A}\right)<1,$$

where $\rho(\boldsymbol{A})$ denotes the spectral radius of $\boldsymbol{A}$ defined in (9).

Due to the invertibility of $\boldsymbol{I}-(1-p)\boldsymbol{A}$, it follows [54, Corollary 5.6.16] that

$$\sum_{k=0}^{\infty}(1-p)^k\boldsymbol{A}^k=\left(\boldsymbol{I}-(1-p)\boldsymbol{A}\right)^{-1},$$

such that (12) becomes

$$\sum_{k=0}^{K}\mathbf{\Phi}^k \le \left(\boldsymbol{I}-(1-p)\boldsymbol{A}\right)^{-1}\mathbf{\Phi}^0+\boldsymbol{C}\sum_{k=0}^{K-1}\boldsymbol{e}^k, \qquad (13)$$

where

$$\boldsymbol{C}=\left(\boldsymbol{I}-(1-p)\boldsymbol{A}\right)^{-1}\frac{(1-p)\lambda^2}{\lambda_p}\boldsymbol{B}.$$

Now, we are going to control the upper bound of the $(\boldsymbol{I}-(1-p)\boldsymbol{A})^{-1}$, i.e., the upper bound of

$$\frac{1}{\det(\boldsymbol{I}-(1-p)\boldsymbol{A})}\operatorname{adj}(\boldsymbol{I}-(1-p)\boldsymbol{A}),$$

where $\det(\boldsymbol{A})$ means the determinant of $\boldsymbol{A}$ and $\operatorname{adj}(\boldsymbol{A})$ represents the adjugate of $\boldsymbol{A}$. If the step-size further satisfies

$\eta \le \frac{(1+p)\lambda_p^2}{360(T_o+1)L}$, we have

$$\begin{aligned}&\det(\boldsymbol{I}-(1-p)\boldsymbol{A})\\&=\left(\frac{(1+p)\lambda_p}{2}\right)^2-\frac{16000\lambda^4(T_o+1)^2L^2(1-p)^2\eta^2}{\lambda_p^2}\\&\ge\frac{(1+p)^2\lambda_p^2}{4}-\frac{(1+p)^2\lambda_p^2}{8}=\frac{(1+p)^2\lambda_p^2}{8}.\end{aligned}$$

Then,

$$\begin{aligned}&(\boldsymbol{I}-(1-p)\boldsymbol{A})^{-1}\\&\le\frac{4}{(1+p)^2\lambda_p^3}\begin{bmatrix}(1+p)\lambda_p^2 & 80(1-\lambda_p)(T_o+1)^2\eta^2\\ 800(1-\lambda_p)L^2 & (1+p)\lambda_p^2\end{bmatrix},\end{aligned}$$

and

$$\boldsymbol{C}\le\frac{240(1-\lambda_p)}{(1+p)^2\lambda_p^4}\begin{bmatrix}4c_1L^2(T_o+1)^4\eta^2\eta_l^2 & 6nc_2(T_o+1)^2\eta^2\\ 3c_3L^2(T_o+1)^2\eta^2 & 3nc_4\end{bmatrix},$$

where

$$\begin{aligned}c_1&=\left((1+p)\lambda_p^2+60(1-\lambda_p)\eta_c^2\right),\\c_2&=\left((1+p)\lambda_p^2+40(1-\lambda_p)\right),\\c_3&=\left((1+p)\lambda_p^2+1100(1-\lambda_p)L^2(T_o+1)^2\eta_l^2\right),\\c_4&=\left((1+p)\lambda_p^2+1600(1-\lambda_p)L^2(T_o+1)^2\eta^2\right).\end{aligned}$$

Thus, if $\boldsymbol{X}^0=\boldsymbol{x}^0\mathbf{1}_n^\top$,

$$\begin{aligned}&\sum_{k=0}^{K-1}\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2]\le\sum_{k=0}^{K}\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2]\\&\le\frac{320(1-\lambda_p)(T_o+1)^2\eta^2}{(1+p)^2\lambda_p^3}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]\\&\quad+\frac{960c_1(1-\lambda_p)L^2(T_o+1)^4\eta^2\eta_l^2}{(1+p)^2\lambda_p^4}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2]\\&\quad+\frac{1440c_2(1-\lambda_p)(T_o+1)^2\eta^2}{b(1+p)^2\lambda_p^4}nK\sigma^2,\end{aligned}\tag{14a}$$

and

$$\begin{aligned}&\sum_{k=0}^{K-1}\mathbb{E}[\|\boldsymbol{\Phi}_y^k\|_{\mathsf{F}}^2]\le\sum_{k=0}^{K}\mathbb{E}[\|\boldsymbol{\Phi}_y^k\|_{\mathsf{F}}^2]\\&\le\frac{4}{(1+p)\lambda_p}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]\\&\quad+\frac{720c_3(1-\lambda_p)L^2(T_o+1)^2\eta^2}{(1+p)^2\lambda_p^4}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2]\\&\quad+\frac{720c_4(1-\lambda_p)}{b(1+p)^2\lambda_p^4}nK\sigma^2.\end{aligned}\tag{14b}$$

Substituting (14) into the (8), we have

$$\begin{aligned}&\frac{\eta(T_o+1)}{4}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]\\&\le\tilde{f}+\underbrace{\frac{15L^2(T_o+1)^3\eta\eta_l^2}{n}\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2]}_{T_1}\\&\quad+\underbrace{\frac{80c_1L^2(T_o+1)^3\eta\eta_l^2}{n(1+p)^2\lambda_p^3}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]}_{T_2}\\&\quad+\left(T_3+\frac{3LT_o\eta^2}{2n}\right)K\sigma^2/b\end{aligned}\tag{15}$$

where $T_3=\frac{7200(84\eta_c^2+2(1+p)\lambda_p^2)(1-\lambda_p)L^2(T_o+1)^3\eta\eta_l^2}{(1+p)^2\lambda_p^4}+64\,L^2(T_o+1)^3\eta\eta_l^2$. Together with (6) and (14a), we have

$$\begin{aligned}&\sum_{k=0}^{K-1}\mathbb{E}[\|\overline{\boldsymbol{Y}}^k\|_{\mathsf{F}}^2]\\&\le3K\sigma^2/b+3L^2\sum_{k=0}^{K-1}\mathbb{E}[\|\boldsymbol{\Phi}_x^k\|_{\mathsf{F}}^2]+3n\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]\\&\le6n\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]\\&\quad+\frac{1920(1-\lambda_p)L^2(T_o+1)^2\eta^2}{(1+p)^2\lambda_p^3}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]\\&\quad+\left(\frac{1440c_2(1-\lambda_p)L^2(T_o+1)^2\eta^2}{(1+p)^2\lambda_p^4}+\frac{1}{n}\right)6nK\sigma^2/b\end{aligned}$$

where the last inequality is due to the step-size conditions, i.e., $\eta\le\frac{(1+p)\lambda_p^2}{360L(T_o+1)}$ and $\eta_l\le\frac{1}{8L(T_o+1)}$ s.t.

$$\frac{3\cdot960c_1L^4(T_o+1)^4\eta^2\eta_l^2}{(1+p)^2\lambda_p^4}\le\frac{1}{2}.$$

By further assuming that $\eta_l\le\frac{1}{27L(T_o+1)}$ we have

$$\begin{aligned}T_1&\le\frac{\eta(T_o+1)}{8}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]\\&\quad+\frac{\alpha_0(1-p)(T_o+1)\eta}{n}\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2]+\alpha_1L^2(T_o+1)^3\eta\eta_l^2K\frac{\sigma^2}{b},\end{aligned}$$

for some positive absolute constant α_0,α_1. Suppose that $\eta_c=\alpha\sqrt{1+p}\lambda_p$ and $\eta_l\le\frac{\sqrt{1+p}\lambda_p}{360\alpha L(T_o+1)}$ for some positive $\alpha>0.1$. Then,

$$\begin{aligned}T_2&\le\alpha_2\left(\frac{(T_o+1)\eta}{n}\right)\mathbb{E}[\|\boldsymbol{\Phi}_y^0\|_{\mathsf{F}}^2],\\T_3&\le\alpha_3\left(\frac{(1-p)L^2(T_o+1)^3\eta^3}{(1+p)^2\lambda_p^4}+L^2(T_o+1)^3\eta\eta_l^2\right)\end{aligned}$$

for some positive absolute constants α_2 and α_3. Therefore,

$$\frac{\eta(T_o+1)}{8}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2]$$

$$\leq \tilde{f} + O\left(\frac{(T_o+1)\eta}{n}\right)\mathbb{E}[\|\mathbf{\Phi}_y^0\|_{\mathsf{F}}^2] + O\left(\frac{(1-p)L^2(T_o+1)^2\eta^2}{(1+p)^2\lambda_p^4} + L^2(T_o+1)^2\eta_l^2 + \frac{3L\eta}{2n}\right)\frac{K(T_o+1)\eta\sigma^2}{b},$$

which is equivalent to (5).

Appendix C
Proof of Corollary 1

By rearranging and relaxing (5), we have

$$\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[\|\nabla f(\overline{\boldsymbol{x}}^k)\|_2^2] \leq O\left(\frac{\tilde{f}}{\eta_l\eta_c T_o K} + \frac{\eta_l\eta_c L\sigma^2}{nb} + \frac{1}{(1+p)\lambda_p^2}\frac{L^2T_o^2\eta_l^2\sigma^2}{b} + \frac{1}{nK}\mathbb{E}[\|\mathbf{\Phi}_y^0\|_{\mathsf{F}}^2]\right). \tag{16}$$

To further fine-tune the step-size and obtain the exact convergence rate, we establish the following lemma, which is slightly different from the Lemma 17 in [55].

Lemma 9: For any parameters $r_0 \geq 0, a_1 > 0, a_2 > 0$, if K is sufficiently large s.t. $\eta' = \min\{(\frac{r_0}{a_1 K})^{\frac{1}{2}}, (\frac{r_0}{a_2 K})^{\frac{1}{3}}\} \leq \bar{\eta}$, i.e.,

$$K \geq \max\left\{\frac{r_0}{a_1\bar{\eta}^2}, \frac{r_0}{a_2\bar{\eta}^3}\right\},$$

we have

$$\Psi^K = \frac{r_0}{\eta' K} + a_1\eta' + a_2(\eta')^2 \leq 2\left(\frac{a_1 r_0}{K}\right)^{\frac{1}{2}} + 2\left(\frac{\sqrt{a_2}r_0}{K}\right)^{\frac{2}{3}}.$$

Proof: We mainly follow the proof of [55, Lemma 17].

- If $\eta' = (\frac{r_0}{a_1 K})^{\frac{1}{2}} \leq (\frac{r_0}{a_2 K})^{\frac{1}{3}}$,

$$\Psi^K \leq 2\left(\frac{r_0 a_1}{K}\right)^{\frac{1}{2}} + a_2\left(\frac{r_0}{a_1 K}\right) \leq 2\left(\frac{r_0 a_1}{K}\right)^{\frac{1}{2}} + \left(\frac{\sqrt{a_2}r_0}{K}\right)^{\frac{2}{3}}.$$

- If $\eta' = (\frac{r_0}{a_2 K})^{\frac{1}{3}} \leq (\frac{r_0}{a_1 K})^{\frac{1}{2}}$,

$$\Psi^K \leq 2\left(\frac{\sqrt{a_2}r_0}{K}\right)^{\frac{2}{3}} + a_1\left(\frac{r_0}{a_2 K}\right)^{\frac{1}{3}} \leq 2\left(\frac{\sqrt{a_2}r_0}{K}\right)^{\frac{2}{3}} + \left(\frac{r_0 a_1}{K}\right)^{\frac{1}{2}}.$$

From $K \geq \max\{\frac{r_0}{a_1\bar{\eta}^2}, \frac{r_0}{a_2\bar{\eta}^3}\}$, we have $\eta' \leq \bar{\eta}$. □

Then, applying Lemma 9 with $\eta' = \alpha T_o\eta_l$ and

$$r_0 = \frac{\tilde{f}}{\sqrt{1+p}\lambda_p}, \quad a_1 = \frac{\sqrt{1+p}\lambda_p L\sigma^2}{nT_o b}, a_2 = \frac{L^2\sigma^2}{(1+p)\lambda_p^2 b}.$$

we bound the right hand side of (16) as

$$O\left(\left(\frac{L\tilde{f}\sigma^2}{nT_o bK}\right)^{\frac{1}{2}} + \left(\frac{L\tilde{f}\sigma}{(1+p)\lambda_p^2\sqrt{b}K}\right)^{\frac{2}{3}} + \frac{1}{nK}\mathbb{E}[\|\mathbf{\Phi}_y^0\|_{\mathsf{F}}^2]\right),$$

if the number of communication rounds K is sufficiently large, i.e., $K \geq \max\{\frac{360^2 nT_o bL\tilde{f}}{(1+p)^2\lambda_p^4\sigma^2}, \frac{360^3 bL\tilde{f}}{(1+p)\lambda_p^2\sigma^2}\}$ such that $\min\{(\frac{r_0}{a_1 K})^{\frac{1}{2}}, (\frac{r_0}{a_2 K})^{\frac{1}{3}}\} \leq \bar{\eta} = \frac{\sqrt{1+p}\lambda_p}{360\,L}$.

References

[1] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. 3rd Int. Symp. Inf. Process. Sensor Netw.*, 2004, pp. 20–27.

[2] A. Beck, A. Nedić, A. Ozdaglar, and M. Teboulle, "An $O(1/k)$ gradient method for network resource allocation problems," *IEEE Trans. Control Netw. Syst.*, vol. 1, no. 1, pp. 64–73, Mar. 2014.

[3] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Trans. Ind. Informat.*, vol. 9, no. 1, pp. 427–438, Feb. 2013.

[4] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proc. IEEE*, vol. 106, no. 5, pp. 953–976, May 2018.

[5] Y. Chen, K. Yuan, Y. Zhang, P. Pan, Y. Xu, and W. Yin, "Accelerating gossip SGD with periodic global averaging," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 1791–1802.

[6] Y. Guo, Y. Sun, R. Hu, and Y. Gong, "Hybrid local SGD for federated learning with heterogeneous communications," in *Proc. Int. Conf. Learn. Representations*, 2022.

[7] Y. Wang, P. Fang, and J. Chen, "Accelerating adaptive federated optimization with local gossip communications," in *Proc. Workshop Federated Learning, Recent Adv. New Challenges (Conjunction NeurIPS 2022)*, 2022.

[8] I. E. Carvajal-Roca and J. Wang, "A semi-decentralized security framework for connected and autonomous vehicles," in *Proc. IEEE 94th Veh. Technol. Conf.*, 2021, pp. 1–6.

[9] M. Navidi, S. M. Moghaddas-Tafreshi, and A. M. Alishvandi, "A semi-decentralized framework for simultaneous expansion planning of privately owned multi-regional energy systems and sub-transmission grid," *Int. J. Elect. Power Energy Syst.*, vol. 128, 2021, Art. no. 106795.

[10] X. Miao, Y. Shi, Z. Yang, B. Cui, and Z. Jia, "SDPipe: A semi-decentralized framework for heterogeneity-aware pipeline-parallel training," *Proc. VLDB Endowment*, vol. 16, no. 9, pp. 2354–2363, 2023.

[11] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[13] P. D. Lorenzo and G. Scutari, "NEXT: In-network nonconvex optimization," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 2, pp. 120–136, Jun. 2016.

[14] S. Lu and C. W. Wu, "Decentralized stochastic non-convex optimization over weakly connected time-varying digraphs," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 5770–5774.

[15] X. Huang and K. Yuan, "Optimal complexity in non-convex decentralized learning over time-varying networks," in *Proc. OPT 2022: Optim. Mach. Learn. (NeurIPS 2022 Workshop)*, 2022.

[16] A. Nedić and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, Jan. 2009.

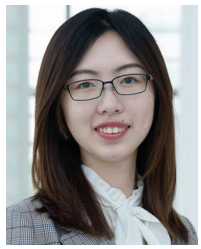
[17] M. Hong, D. Hajinezhad, and M. M. Zhao, "Prox-PDA: The proximal primal-dual algorithm for fast distributed nonconvex optimization and learning over networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1529–1538.

[18] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D^2: Decentralized training over decentralized data," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4848–4856.

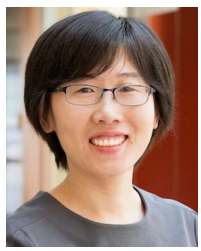
[19] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. Int. Conf. Learn. Representations*, 2020.

[20] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5132–5143.

[21] Y. Wang, L. Lin, and J. Chen, "Communication-efficient adaptive federated learning," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 22802–22838.
[22] G. Scutari and Y. Sun, "Distributed nonconvex constrained optimization over time-varying digraphs," *Math. Program.*, vol. 176, no. 1–2, pp. 497–544, 2019.
[23] H. Sun, S. Lu, and M. Hong, "Improving the sample and communication complexity for decentralized non-convex optimization: Joint gradient estimation and tracking," in *Int. Conf. Mach. Learn.*, 2020, pp. 9217–9228.
[24] Y. Sun, M. Maros, G. Scutari, and G. Cheng, "High-dimensional inference over networks: Linear convergence and statistical guarantees," 2022, *arXiv:2201.08507*.
[25] H. Zhao, B. Li, Z. Li, P. Richtárik, and Y. Chi, "BEER: Fast $O(1/T)$ rate for decentralized nonconvex optimization with communication compression," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 31653–31667.
[26] X. Cao et al., "Communication-efficient distributed learning: An overview," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 851–873, Apr. 2023.
[27] A. Koloskova, T. Lin, S. U. Stich, and M. Jaggi, "Decentralized deep learning with arbitrary communication compression," in *Proc. Int. Conf. Learn. Representations*, 2020.
[28] B. Li, S. Cen, Y. Chen, and Y. Chi, "Communication-efficient distributed optimization in networks with gradient tracking and variance reduction," *J. Mach. Learn. Res.*, vol. 21, no. 180, pp. 1-51, 2020.
[29] E. D. H. Nguyen, S. A. Alghunaim, K. Yuan, and C. A. Uribe, "On the performance of gradient tracking with local updates," in *Proc. IEEE 62nd Conf. Decis. Control*, 2023, pp. 4309–4313.
[30] S. Ge and T. H. Chang, "Gradient and variable tracking with multiple local SGD for decentralized non-convex learning," in *Proc. IEEE 62nd Conf. Decis. Control*, 2023, pp. 133–138.
[31] Y. Sun, J. Shao, Y. Mao, J. H. Wang, and J. Zhang, "Semi-decentralized federated edge learning with data and device heterogeneity," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 2, pp. 1487–1501, Jun. 2023.
[32] F. P. C. Lin, S. Hosseinalipour, S. S. Azam, C. G. Brinton, and N. Michelusi, "Semi-decentralized federated learning with cooperative D2D local model aggregations," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3851–3869, Dec. 2021.
[33] J. George and P. Gurram, "Distributed stochastic gradient descent with event-triggered communication," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 5, 2020, pp. 7169–7178.
[34] W. Du, X. Yi, J. George, K. H. Johansson, and T. Yang, "Distributed optimization with dynamic event-triggered mechanisms," in *Proc. 2018 IEEE Conf. Decis. Control*, 2018, pp. 969–974.
[35] X. He, X. Yi, Y. Zhao, K. H. Johansson, and V. Gupta, "Asymptotic analysis of federated learning under event-triggered communication," *IEEE Trans. Signal Process.*, vol. 71, pp. 2654–2667, 2023.
[36] Z. Song et al., "Communication-efficient topologies for decentralized learning with $O(1)$ consensus rate," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 1073–1085.
[37] L. Ding, K. Jin, B. Ying, K. Yuan, and W. Yin, "DSGD-CECA: Decentralized SGD with communication-optimal exact consensus algorithm," in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 8067–8089.
[38] N. Tupitsa, S. Horváth, M. Takáč, and E. Gorbunov, "Federated learning can find friends that are beneficial," 2024, *arXiv:2402.05050*.
[39] S. Liu, C. Liu, D. Wen, and G. Yu, "Efficient collaborative learning over unreliable D2D network: Adaptive cluster head selection and resource allocation," *IEEE Trans. Commun.*, vol. 73, no. 1, pp. 425–438, Jan. 2025.
[40] R. Parasnis, S. Hosseinalipour, Y. W. Chu, M. Chiang, and C. G. Brinton, "Connectivity-aware semi-decentralized federated learning over time-varying D2D networks," in *Proc. 24th Int. Symp. Theory, Algorithmic Foundations, Protocol Des. Mobile Netw. Mobile Comput.*, 2023, pp. 31–40.
[41] R. Bollapragada, R. Byrd, and J. Nocedal, "Adaptive sampling strategies for stochastic optimization," *SIAM J. Optim.*, vol. 28, no. 4, pp. 3312–3343, 2018.
[42] A. Nedić, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, 2017.
[43] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 3, pp. 1245–1260, Sep. 2018.
[44] S. Pu and A. Nedić, "Distributed stochastic gradient tracking methods," *Math. Program.*, vol. 187, pp. 409–457, 2021.
[45] R. Xin, U. A. Khan, and S. Kar, "Variance-reduced decentralized stochastic optimization with accelerated convergence," *IEEE Trans. Signal Process.*, vol. 68, pp. 6255–6271, 2020.
[46] Y. Liu, T. Lin, A. Koloskova, and S. U. Stich, "Decentralized gradient tracking with local steps," *Optim. Methods Softw.*, pp. 1–28, 2024.
[47] A. Koloskova, T. Lin, and S. U. Stich, "An improved analysis of gradient tracking for decentralized machine learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 11422–11435.
[48] Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh, "SpiderBoost and momentum: Faster variance reduction algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 2406–2416.
[49] C. C. Chang and C.J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, 2011, Art. no. 27.
[50] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Control Lett.*, vol. 53, no. 1, pp. 65–78, 2004.
[51] L. Deng, "The MNIST database of handwritten digit images for machine learning research [best of the web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.
[52] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Master's thesis, Univ. Tront, 2009.
[53] R. Saha, M. Seif, M. Yemini, A. J. Goldsmith, and H. V. Poor, "Privacy preserving semi-decentralized mean estimation over intermittently-connected networks," *IEEE Trans. Signal Process.*, vol. 72, pp. 5306–5321, 2024.
[54] R. A. Horn and C. R. Johnson, *Matrix Analysis*. New York, NY, USA: Cambridge Univ. Press, 2012.
[55] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, "A unified theory of decentralized SGD with changing topology and local updates," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5381–5393.



He Wang (Graduate Student Member, IEEE) received the B.E. degree in computer science and technology from ShanghaiTech University, Shanghai, China, in 2019, and the M.S.E. degree in communication and information systems from the University of Chinese Academy of Sciences, Beijing, China, in 2022. She is currently working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, USA. Her research interests include distributed optimization, game theory, and reinforcement learning.



Yuejie Chi (Fellow, IEEE) received the B.E. (Hons.) degree in electrical engineering from Tsinghua University, Beijing, China, in 200, the M.A. and Ph.D. degrees. in electrical engineering from Princeton University, Princeton, NJ, USA, in 2009 and 2012, respectively. She is currently the Sense of Wonder Group Endowed Professor of electrical and computer engineering in AI systems with Carnegie Mellon University, Pittsburgh, PA, USA with courtesy appointments in the Machine Learning Department and CyLab. Her research interests include the theoretical and algorithmic foundations of data science, signal processing, machine learning and inverse problems, with applications in sensing, imaging, decision making, and AI systems. She was the recipient of Presidential Early Career Award for Scientists and Engineers, SIAM Activity Group on Imaging Science Best Paper Prize, IEEE Signal Processing Society Young Author Best Paper Award, and the inaugural IEEE Signal Processing Society Early Career Technical Achievement Award for contributions to high-dimensional structured signal processing. She was named a Goldsmith Lecturer by IEEE Information Theory Society, a Distinguished Lecturer by IEEE Signal Processing Society, and a Distinguished Speaker by ACM. She is/was currently an Associate Editor for IEEE TRANSACTIONS ON INFORMATION THEORY, IEEE TRANSACTIONS ON SIGNAL PROCESSING, IEEE TRANSACTIONS ON PATTERN RECOGNITION AND MACHINE INTELLIGENCE, *Information and Inference*: *A Journal of the IMA, and SIAM Journal on Mathematics of Data Science*.