Decentralized Federated Averaging

Tao Sun, Dongsheng Li, and Bao Wang

Abstract—Federated averaging (FedAvg) is a communication efficient algorithm for the distributed training with an enormous number of clients. In FedAvg, clients keep their data locally for privacy protection; a central parameter server is used to communicate between clients. This central server distributes the parameters to each client and collects the updated parameters from clients. FedAvg is mostly studied in centralized fashions, which requires massive communication between server and clients in each communication. Moreover, attacking the central server can break the whole system's privacy. In this paper, we study the decentralized FedAvg with momentum (DFedAvgM), which is implemented on clients that are connected by an undirected graph. In DFedAvgM, all clients perform stochastic gradient descent with momentum and communicate with their neighbors only. To further reduce the communication cost, we also consider the quantized DFedAvgM. We prove convergence of the (quantized) DFedAvgM under trivial assumptions; the convergence rate can be improved when the loss function satisfies the PŁ property. Finally, we numerically verify the efficacy of DFedAvgM.

Index Terms—Decentralized Optimization, Federated Averaging, Momentum, Stochastic Gradient Descent

1 Introduction

Federated learning (FL) is a privacy-preserving distributed machine learning (ML) paradigm [1]. In FL, a central server connects with enormous clients (e.g., mobile phones, pad, etc.); the clients keep their data without sharing it with the server. In each communication round, clients receive the current global model from the server, and a small portion of clients are selected to update the global model by running stochastic gradient descent (SGD) [2] for multiple iterations using local data. The central server then aggregates these updated parameters to obtain the updated global model. The above learning algorithm is known as federated average (FedAvg) [1]. In particular, if the clients are homogeneous, FedAvg is equivalent to the local SGD [3]. FedAvg involves multiple local SGD updates and one aggregation by the server in each communication round, which significantly reduces the communication cost between sever and clients compared to the conventional distributed training with one local SGD update and one communication.

In FL applications, large companies and government organizations usually play the role of the central server. On the one hand, since the number of clients in FL is massive, the communication cost between the server and clients can be a bottleneck [4]. On the other hand, the updated models collected from clients encode the private information of the local data; hackers can attack the central server to break the privacy of the whole system, which remains the privacy issue as a serious concern. To this end, decentralized federated learning has been proposed [5], [6], where all clients are connected with an undirected graph. Decentralized FL replaces the server-clients communication in FL with clients-clients communication.

In this paper, we consider two issues about decentralized FL: 1) Although there is no expensive communication between server and clients in decentralized FL, the communication between local clients is costly when the ML model itself is large. Therefore, it is crucial to ask *can we reduce the communication cost between clients?* 2) Momentum is a well-known acceleration technique for SGD [7]. It is natural to ask *can we use SGD with momentum to improve the training of ML models in decentralized FL with theoretical convergence guarantees?*

1.1 Our Contributions

We answer the above questions affirmatively by proposing the decentralized FedAvg with momentum (DFedAvgM). To further reduce the communication cost between clients, we also integrate quantization with DFedAvgM. Our contributions in this paper are elaborated below in threefold.

- Algorithmically, we extend FedAvg to the decentralized setting, where all clients are connected by an undirected graph. We motivate DFedAvgM from the decentralized SGD (DSGD) algorithm. In particular, we use SGD with momentum to train ML models on each client. To reduce the communication cost between each client, we further introduce a quantized version of DFedAvgM, in which each client will send and receive a quantized model.

- Theoretically, we prove the convergence of (quantized) DFedAvgM. Our theoretical results show that the convergence rate of (quantized) DFedAvgM is not inferior to that of SGD or DSGD. More specifically, we show that the convergence rates of both DFedAvgM and quantized DFedAvgM depend on the local training and the graph that connects all clients. Besides the convergence results under nonconvex assumptions, we also establish their convergence guarantee under the Polyak-Łojasiewicz (PŁ) condition, which has been widely studied in nonconvex optimization. Under the

This work is sponsored in part by the National Key R&D Program of China under Grant (2018YFB0204300) and the National Natural Science Foundation of China under Grants (61932001 and 61906200).

Tao Sun and Dongsheng Li are with the College of Computer, National University of Defense Technology, Changsha, 410073, Hunan, China. (e-mails: nudtsuntao@163.com, dsli@nudt.edu.cn)

Bao Wang is with the Scientific Computing & Imaging Institute, University of Utah, USA. (e-mail: wangbaonj@gmail.com)

Dongsheng Li and Bao Wang the co-corresponding authors.

PŁ condition, we establish a faster convergence rate for (quantized) DFedAvgM. Furthermore, we present a sufficient condition to guarantee reducing communication costs.

- Empirically, we perform extensive numerical experiments on training deep neural networks (DNNs) on various datasets in both **IID** and **Non-IID** settings. Our results show the effectiveness of (quantized) DFedAvgM for training ML models, saving communication costs, and protecting training data's membership privacy.

1.2 More Related Works

We briefly review three lines of work that are most related to this paper, i.e., federated learning, decentralized training, and decentralized federated learning.

Federated Learning. Many variants of FedAvg have been developed with theoretical guarantees. [8] uses the momentum method for local clients in FedAvg. [9] proposes the adaptive FedAvg, whose central parameter server uses the adaptive learning rate ti aggregate local models. Lazy and quantizatized gradients are used to reduce communications [10], [11]. [12] proposes a Newton-type scheme for FL. The convergence analysis of FedAvg on heterogeneous data is discussed by [13], [14]. The advances and open problems in FL is available in two survey papers [15], [16].

Decentralized Training. Decentralized algorithms are originally developed to calculate the mean of data that are distributed over multiple sensors [17], [18], [19], [20]. Decentralized (sub)gradient descents (DGD), one of the simplest and efficient decentralized algorithms, have been studied in [21], [22], [23], [24], [25]. In DGD, the convexity assumption is unnecessary [26], which makes DGD useful for nonconvex optimization. A provably convergent DSGD is proposed in [27], [28], [4]. [27] provides the complexity result of a stochastic decentralized algorithm. [28] designs a stochastic decentralized algorithm with the dual information and provide the theoretical convergence guarantee. [4] proves that DSGD outperforms SGD in communication efficiency. Asynchronous DSGD is analyzed in [29]. DGD with momentum is proposed in [30], [31]. Quantized DSGD has been proposed in [32].

Decentralized Federated Learning. Decentralized FL is a learning paradigm of choice when the edge devices do not trust the central server in protecting their privacy [33]. The authors in [34] propose a novel FL framework without a central server for medical applications, and the new method offers a highly dynamic peer-to-peer environment. [6] considers training an ML model with a connected network whose nodes take a Bayesian-like approach by introducing a belief of the parameter space.

1.3 Organizations

We organize this paper as follows: in section 2, we present a mathematical formulation of our problem and some necessary assumption. In section 3, we present the DFedAvgM and its quantized algorithms. We present the convergence of the proposed algorithm in section 4. We provide extensive numerical verification of DFedAvgM in section 6. This paper ends up with concluding remarks. Technical proofs and more experimental details are provided in the appendix.

1.4 Notation

We denote scalars and vectors by lower case and lower case boldface letters, respectively, and matrices by upper case boldface letters. For a vector $\mathbf{x} = (x_1, \cdots, x_d) \in \mathbb{R}^d$, we denote its ℓ_p norm ($p \geq 1$) by $\|\mathbf{x}\|_p = (\sum_{i=1}^d |x_i|^p)^{1/p}$, and denote the ℓ_∞ norm of $\mathbf{x}$ by $\|\mathbf{x}\|_\infty = \max_{i=1}^d |x_i|$ and denote ℓ_2 norm as $\|\mathbf{x}\|$. For a matrix $\mathbf{A}$, we denote its transpose as $\mathbf{A}^\top$. Given two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n = \mathcal{O}(b_n)$ if there exists a positive constant $0 < C < +\infty$ such that $a_n \leq Cb_n$, and we write $a_n = \Theta(b_n)$ if there exist two positive constants C_1 and C_2 such that $a_n \leq C_1 b_n$ and $b_n \leq C_2 a_n$. $\tilde{\mathcal{O}}(a_n)$ hides the logarithmic factor of a_n. For a function $f(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$, we denote its gradient as $\nabla f(\mathbf{x})$ and its Hessian as $\nabla^2 f(\mathbf{x})$, and denote its minimum as $\min f$. We use $\mathbb{E}[\cdot]$ to denote the expectation with respect to the underlying probability space.

2 Problem Formulation and Assumptions

We consider the following optimization task

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) := \frac{1}{m}\sum_{i=1}^m f_i(\mathbf{x}), \quad f_i(\mathbf{x}) = \mathbb{E}_{\xi\sim\mathcal{D}_i} F_i(\mathbf{x};\xi), \qquad (1)$$

where $\mathcal{D}_i$ denotes the data distribution in the i-th client and $F_i(\mathbf{x};\xi)$ is the loss function associated with the training data ξ. Problem (1) models many applications in ML, which is known as empirical risk minimization (ERM). We list several assumptions for the subsequent analysis.

Assumption 1. The function f_i is differentiable and ∇f_i is L-Lipschitz continuous, $\forall i \in \{1, 2, \ldots, m\}$, i.e., $\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$, for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

The first-order Lipschitz assumption is commonly used in the ML community. Here, for simplicity, we suppose all functions enjoy the same Lipschitz constant L. We can also assume that these functions have non-uniform Lipschitz constants, which does not affect our convergent analysis.

Assumption 2. The gradient of the function f_i have σ_l-bounded variance, i.e., $\mathbb{E}[\|\nabla F_i(\mathbf{x};\xi) - \nabla f_i(\mathbf{x})\|^2] \leq \sigma_l^2$ for all $\mathbf{x} \in \mathbb{R}^d$ $\forall i \in \{1, 2, \ldots, m\}$. This paper also assumes the (global) variance is bounded, i.e., $\frac{1}{m}\sum_{i=1}^m \|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \sigma_g^2$ for all $\mathbf{x} \in \mathbb{R}^d$.

The uniform local variance assumption is also used for the ease of presentation, which is straightforward to generalize to non-uniform cases. The global variance assumption is used in [9], [35]. The constant σ_g reflects the heterogeneity of the data sets $(\mathcal{D}_i)_{1\leq i\leq m}$, and when $(\mathcal{D}_i)_{1\leq i\leq m}$ follow the same distribution, $\sigma_g = 0$.

Assumption 3. [36], [4] For any $i \in \{1, 2, \ldots, m\}$ and $\mathbf{x} \in \mathbb{R}^d$, we have $\|\nabla f_i(\mathbf{x})\| \leq B$ for some $B > 0$.

An important notion in decentralized optimization is the *mixing matrix*, which is usually associated with a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the vertex set $\mathcal{V} = \{1, ..., m\}$ and the edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. Any edge $(i, l) \in \mathcal{E}$ represents a communication link between nodes i and l. We recall the definition of the mixing matrix associated with the graph $\mathcal{G}$.

Definition 1 (Mixing matrix). The mixing matrix $\mathbf{W} = [w_{i,j}] \in \mathbb{R}^{m\times m}$ is assumed to have the following properties: 1. (Graph) If $i \neq j$ and $(i, j) \notin \mathcal{E}$, then $w_{i,j} = 0$,

otherwise, $w_{i,j} > 0$; 2. (Symmetry) $\mathbf{W} = \mathbf{W}^\top$; 3. (Null space property) $\text{null}\{\mathbf{I} - \mathbf{W}\} = \text{span}\{\mathbf{1}\}$; 4. (Spectral property) $\mathbf{I} \succeq \mathbf{W} \succ -\mathbf{I}$.

For a graph, the corresponding mixing matrix is not unique; given the adjacency matrix of a graph, its maximum-degree matrix and metropolis-hastings [37] are both mixing matrices. The symmetric property of $\mathbf{W}$ indicates that its eigenvalues are real and can be sorted in the non-increasing order. Let $\lambda_i(\mathbf{W})$ denote the i-th largest eigenvalue of $\mathbf{W}$, that is, $\lambda_1(\mathbf{W}) = 1 > \lambda_2(\mathbf{W}) \geq \cdots \geq \lambda_m(\mathbf{W}) > -1$.[1] The mixing matrix also serves as a probability transition matrix of a Markov chain. A quite important constant of $\mathbf{W}$ is $\lambda = \lambda(\mathbf{W}) := \max\{|\lambda_2(\mathbf{W})|, |\lambda_m(\mathbf{W})|\}$, which describes the speed of the Markov chain introduced by the mixing matrix converges to the stable state.

3 Decentralized Federated Averaging

3.1 Decentralized FedAvg with Momentum

We first briefly review the previous work on decentralized training, which carries out in the following fashion:

1) client i holds an approximate copy of the parameters $\mathbf{x}(i) \in \mathbb{R}^d$ and calculate an unbiased estimate of $\nabla f_i := \mathbf{g}(i)$ at $\mathbf{x}(i)$. $(\mathbf{x}(i))_{1\leq i\leq m}$ can be non-consensus;
2) (*communication*) client i updates its local parameters $\mathbf{x}(i)$ as the weighted average of its neighbors: $\tilde{\mathbf{x}}(i) = \sum_{l\in\mathcal{N}(i)} w_{i,l}\mathbf{x}(l)$;
3) (*training*) client i updates its parameters as $\mathbf{x}(i) \leftarrow \tilde{\mathbf{x}}(i) - \eta\mathbf{g}(i)$ with a learning rate $\eta > 0$.

Algorithm 1 DFedAvgM

Parameters: $\eta > 0, K \in \mathbb{Z}^+, 0 \leq \theta < 1$.
Initialization: $\mathbf{x}^0 = \mathbf{0}$
for $t = 1, 2, \ldots$ **do**
  for $i = \{1, 2, \ldots, m\}$ **do**
    node i performs local training (4) K times and sends $\mathbf{z}^t(i) = \mathbf{y}^{t,K}(i)$ to $\mathcal{N}(i)$
    node i updates as (5)
  end for
end for

The traditional decentralization can be described in Figure 1 (a), in which, a *communication* step is needed after each *training* iteration. This indicate that the above vanilla decentralized algorithm is different from FedAvg, and the later performs multiple local *training* step before *communication*. To this end, we have to slightly modify the scheme of the decentralized algorithm. For simplicity, we consider modifying DSGD to motivate our decentralized FedAvg algorithm. Note that when the original DGD is applied to solve (1), we end up with the following iteration

$$\begin{aligned}\mathbf{x}^{t+1}(i) &= \sum_{l\in\mathcal{N}(i)} w_{i,l}\mathbf{x}^t(l) - \gamma\mathbf{g}^t(i)\\ &= \sum_{l\in\mathcal{N}(i)} w_{i,l}[\mathbf{x}^t(l) - \gamma\mathbf{g}^t(i)],\end{aligned} \tag{2}$$

where we used the fact that $\sum_{l\in\mathcal{N}(i)} w_{i,l} = 1$. In (2), if we replace $\mathbf{x}^t(l)$ by $\mathbf{x}^t(i)$, the algorithm then iterates as

$$\mathbf{x}^{t+1}(i) = \sum_{l\in\mathcal{N}(i)} w_{i,l}[\mathbf{x}^t(i) - \gamma\mathbf{g}^t(i)]. \tag{3}$$

1. This is based on the spectral property of mixing matrix.

In (3), clients communicate with their neighbors after one training iteration, which is then possible to generalize to the federated optimization setting. We replace the single SGD iteration in (3) with multiple SGD with heavy-ball [38] iterations. Therefore, the DFedAvgM can be presented as follows: In each $t \in \mathbb{Z}^+$, for each client $i \in \{1, 2, \ldots, m\}$, let $\mathbf{y}^{t,-1}(i) = \mathbf{y}^{t,0}(i) = \mathbf{x}^t(i)$. The inner iteration in each node then performs as

$$\mathbf{y}^{t,k+1}(i) = \mathbf{y}^{t,k}(i) - \eta\tilde{\mathbf{g}}^{t,k}(i) + \theta(\mathbf{y}^{t,k}(i) - \mathbf{y}^{t,k-1}(i)), \tag{4}$$

where $\mathbb{E}\tilde{\mathbf{g}}^{t,k}(i) = \nabla f_i(\mathbf{y}^{t,k}(i))$. After K inner iterations in each local client, the resulting parameters $\mathbf{z}^t(i) \leftarrow \mathbf{y}^{t,K}(i)$ is sent to its neighbors ($\mathcal{N}(i)$). Every client then updates its parameters by taking the local averaging as follows

$$\mathbf{x}^{t+1}(i) = \sum_{l\in\mathcal{N}(i)} w_{i,l}\mathbf{z}^t(l). \tag{5}$$

The procedure of DFedAvgM can be illustrated as Figure 1 (b). It is seen that DFedAvgM plays the tradeoff between local computing and communications. It is well-known that the communication costs are usually much more expensive than the computation costs [39], which indicates DFedAvgM can be more efficient than DSGD.

Algorithm 2 Quantized DFedAvgM

parameters: $\eta > 0$, $K \in \mathbb{Z}^+$, $0 \leq \theta < 1$, s, b.
initialization: $\mathbf{x}^0 = \mathbf{0}$
for $t = 1, 2, \ldots$ **do**
  for $i = \{1, 2, \ldots, m\}$ **do**
    node i performs local training (4) K times and sends $\mathbf{q}^t(i) = Q[\mathbf{y}^{t,K}(i) - \mathbf{x}^t(i)]$ to $\mathcal{N}(i)$
    node i updates as (7)
  end for
end for

3.2 Efficient Communication via Quantization

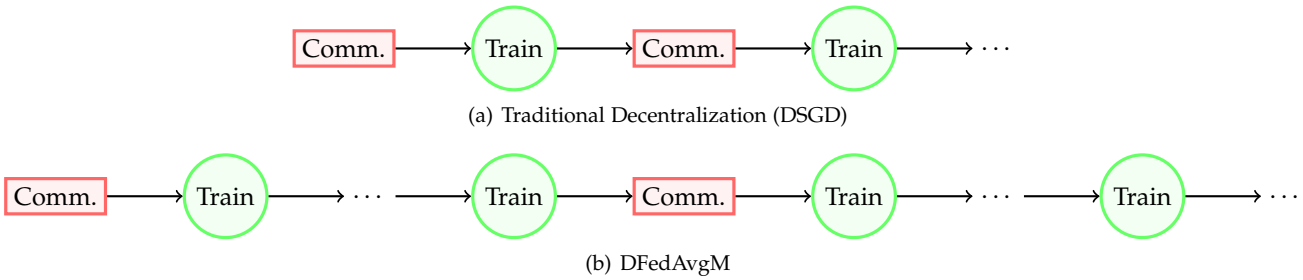
In DFedAvgM, client i needs to send $\mathbf{x}^t(i)$ to its neighbours $\mathcal{N}(i)$. Thus, when the number of neighbours $\mathcal{N}(i)$ grows, client-client communications become the major bottleneck on algorithms' efficiency. We leverage the quantization trick to reduce the communication cost [40], [41]. In particular, we consider the following quantization procedure: Given a constant $s > 0$ and the limited bit number $b \in \mathbb{Z}^+$, the representable range is then $\{-2^{b-1}s, -(2^{b-1}-1)s, \ldots, 0, s, 2s, \ldots, (2^{b-1}-1)s\}$. For any $a \in \mathbb{R}$ with $-2^{b-1}s \leq a < (2^{b-1}-1)s$, we can find an integer $k \in \mathbb{Z}$ such that $ks \leq a < (k+1)s$ and we then use ks to replace a. The above quantization scheme is deterministic, which can be written as $q(a) := \lfloor \frac{a}{s} \rfloor s$ for $a \in \mathbb{R}$; Besides the deterministic rule, the stochastic quantization uses the following scheme

$$q(a) := \begin{cases} ks, \text{ w.p. } 1 - \frac{a-ks}{s}, \\ (k+1)s, \text{ w.p. } \frac{a-ks}{s}. \end{cases}$$

It is easy to see that the stochastic quantization is unbiased, i.e., $\mathbb{E}[q(a)] = a$ for any $a \in \mathbb{R}$. When s is small, deterministic and stochastic quantization schemes perform very similarly. For a vector $\mathbf{x} \in \mathbb{R}^d$ whose coordinates are all stored with 32 bits, we consider quantizing all coordinates of $\mathbf{x} = [x_1, x_2, \ldots, x_d] \in \mathbb{R}^d$. The multi-dimension quantization operator is then defined as

$$Q(\mathbf{x}) := [q(x_1), q(x_2), \ldots, q(x_d)]. \tag{6}$$

For both deterministic and stochastic quantization schemes, we have $\mathbb{E}\|Q(\mathbf{x}) - \mathbf{x}\|^2 \leq \frac{d}{4}s^2$ if $x_i \in [-2^{b-1}s, (2^{b-1}-1)s]$



(a) Traditional Decentralization (DSGD)

(b) DFedAvgM

Fig. 1. Comparison of communication and training styles of traditional decentralized stochastic gradient descent (DSGD) and the proposed decentralized federated average with momentum (DFedAvgM). In DSGD, each client will communicate with its neighbors after one single training step. In DFedAvgM, however, each client will communicate with its neighbors after multiple training iterations.

for $i \in \{1, 2, \ldots, d\}$. In this paper, we consider a quantization operator with the following assumption, which hold for the two quantization schemes mentioned above.

Assumption 4. The quantization operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ satisfies $\mathbb{E}\|Q(\mathbf{x}) - \mathbf{x}\|^2 \leq \frac{d}{4}s^2$ with $s > 0$ for any $\mathbf{x} \in \mathbb{R}^d$.

Directly quantize the parameters is feasible for sufficiently smooth loss functions, but may be impossible for DNNs. To this end, we consider quantizing the difference the difference of parameters. Quantized DFedAvgM can be summarized as: After running (4) K times, client i quantizes $\mathbf{q}^t(i) \leftarrow Q(\mathbf{y}^{t,K}(i) - \mathbf{x}^t(i))$ and send it to $\mathcal{N}(i)$. After receiving $[\mathbf{q}^t(j)]_{j\in\mathcal{N}(i)}$, every client updates its local parameters as

$$\mathbf{x}^{t+1}(i) = \mathbf{x}^t(i) + \sum_{l\in\mathcal{N}(i)} w_{i,l}\mathbf{q}^t(l). \tag{7}$$

In each communication, client i just needs to send the pair $(s, \mathbf{q}^t(i))$ to $\mathcal{N}(i)$, whose representation requires $(32 + db)\deg(\mathcal{N}(i))$ bits rather than $32d\deg(\mathcal{N}(i))$ bits for sending the unquantized version. If d is large and $b < 32$, the communications can be significantly reduced.

4 Convergence Analysis

In this section, we analyze the convergence of the proposed (quantized) DFedAvgM. The convergence analysis of DFedAvgM is much more complicated than SGD, SGD with momentum, and DSGD; the technical difficulty is because $\mathbf{z}^t(i) - \mathbf{x}^t(i)$ fails to be an unbiased estimate of the gradient $\nabla f_i(\mathbf{x}^t(i))$ after multiple iterations of SGD or SGD with momentum in each client. In the following, we consider the convergence of the average point, which is defined as $\overline{\mathbf{x}^t} := \sum_{i=1}^m \mathbf{x}^t(i)/m$. We first present the convergence of DFedAvgM for general nonconvex objective function in the following Theorem.

Theorem 1 (General nonconvexity). Let the sequence $\{\mathbf{x}^t(i)\}_{t\geq 0}$ be generated by DFedAvgM for $i \in \{1, 2 \ldots, m\}$ and suppose Assumptions 1, 2 and 3 hold. Moreover, assume the stepsize η for SGD with momentum that used for training client models satisfies

$$0 < \eta \leq \tfrac{1}{8LK} \text{ and } 64L^2K^2\eta^2 + 64LK\eta < 1,$$

where L is the Lipschitz constant of ∇f and K is the number of client iterations before each communication. Then

$$\min_{1\leq t\leq T} \mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \leq \frac{2f(\overline{\mathbf{x}^1}) - 2\min f}{\gamma(K,\eta)T} + \alpha(K,\eta) + \beta(K,\eta),$$

where T is the total number of communication rounds and the constants are given as

$$\gamma(K,\eta) := \frac{\eta(K-\theta)}{(1-\theta)} - \frac{64(1-\theta)L^2K^4\eta^3}{K-\theta} - 64LK^2\eta^2,$$

$$\alpha(K,\eta) := \frac{(\frac{(1-\theta)L^2K^2\eta^3}{(K-\theta)} + L\eta^2)(8K\sigma_l^2 + 32K^2\sigma_g^2 + \frac{64K^2\theta^2(\sigma_l^2+B^2)}{(1-\theta)^2})}{\frac{\eta(K-\theta)}{(1-\theta)} - \frac{64(1-\theta)L^2K^4\eta^3}{K-\theta} - 64LK^2\eta^2},$$

and

$$\beta(K,\eta,\lambda) := (\frac{64(1-\theta)L^4K^4\eta^5}{(K-\theta)} + 64L^3K^2\eta^4)\times \frac{(8K\sigma_l^2 + 32K^2\sigma_g^2 + 32K^2B^2 + \frac{64K^2\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2))}{[(1-\lambda)(\frac{\eta(K-\theta)}{(1-\theta)} - \frac{64(1-\theta)L^2K^4\eta^3}{K-\theta} - 64LK^2\eta^2)]}.$$

To get an explicit rate on T from Theorem 1, we choose $\eta = \Theta(1/LK\sqrt{T})$. As T is large enough and $64L^2K^2\eta^2 + 64LK\eta < 1$. Then, $\gamma(K,\eta) = \Theta(1/((1-\theta)\sqrt{T}))$, and $\alpha(K,\eta) = \Theta(\frac{(1-\theta)\sigma_l^2 + (1-\theta)K\sigma_g^2 + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2+B^2)}{K\sqrt{T}})$, and $\beta(K,\eta,\lambda) = \Theta(\frac{(1-\theta)(\sigma_l^2 + K\sigma_g^2 + KB^2) + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2+B^2)}{(1-\lambda)KT^{3/2}})$. Based on this choice of η and the Theorem 1, we have the following convergence rate for DFedAvgM.

Proposition 1. As the communication round number T is large enough, it holds that

$$\min_{1\leq t\leq T} \mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 = \mathcal{O}\Big(\frac{(1-\theta)(f(\overline{\mathbf{x}^1}) - \min f)}{\sqrt{T}} + \frac{(1-\theta)\sigma_l^2 + (1-\theta)K\sigma_g^2 + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2+B^2)}{K\sqrt{T}} + \frac{(1-\theta)(\sigma_l^2 + K\sigma_g^2 + KB^2) + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2+B^2)}{(1-\lambda)KT^{3/2}}\Big).$$

From Proposition 1, we can see that the speed of DFedAvgM can be improved when the number of local iteration, K, increases. Also, when the momentum θ is 0 and K is large enough, the bound will be dominated by $\mathcal{O}\Big(\frac{1}{\sqrt{T}} + \frac{\sigma_g^2}{\sqrt{T}} + \frac{\sigma_g^2+B^2}{(1-\lambda)^2T^{3/2}}\Big)$, in which the local variance bound diminishes. This phenomenon coincides with our intuitive understanding: in local client, the use of large

K can result in a local minimizer; then the local variance bound will hurt nothing. To reach any given $\epsilon > 0$ error, DFedAvgM needs $\mathcal{O}(\frac{1}{\epsilon^2})$ communication rounds, which is the same as SGD and DSGD. It is worth mentioning that the above theoretical results show that whether the momentum θ can accelerate the algorithm depends on the relation between $f(\overline{\mathbf{x}^1}) - \min f + \sigma_g^2$ and $\sigma_l^2 + B^2$, i.e., if $f(\overline{\mathbf{x}^1}) - \min f + \sigma_g^2 \gg \sigma_l^2 + B^2$, as $\theta \in [0, 1)$ increases, the rate improves; if $f(\overline{\mathbf{x}^1}) - \min f + \sigma_g^2 \ll \sigma_l^2 + B^2$, large θ may degrade the performance of DFedAvgM.

The convergence results established above, which simply require smooth assumption on the objective functions, are quite general and somehow not sharp due to extra properties are missing. For example, recent (non)convex studies [42], [43], [44] have exploited the algorithmic performance under the PŁ property, which is named after Polyak and Łojasiewicz [45], [46]. For a smooth function f, we say it satisfies PŁ-ν property provided

$$\|\nabla f(\mathbf{x})\|^2 \geq 2\nu(f(\mathbf{x}) - \min f), \quad \forall \mathbf{x} \in \mathrm{dom}(f). \tag{8}$$

The well-known strong convexity implies PŁ condition, but not vice verse. In the following, we present the convergence of DFedAvgM under the PŁ condition.

Theorem 2 (PŁ condition). Assume function f satisfies the PŁ-ν condition, the following convergence rate holds

$$\mathbb{E}f(\overline{\mathbf{x}^T}) - \min f \leq [1 - \nu\gamma(K, \eta)]^T (f(\overline{\mathbf{x}^0}) - \min f) + \frac{\alpha(K, \eta)}{2\nu} + \frac{\beta(K, \eta, \lambda)}{2\nu}.$$

Due to the fact that $f(\overline{\mathbf{x}^0}) - \min f \geq 0$, the right side is larger than $\frac{\alpha(K,\eta)}{2\nu} + \frac{\beta(K,\eta,\lambda)}{2\nu} = \mathcal{O}(\eta)$. If we still let $\eta = \Theta(\frac{1}{LK\sqrt{T}})$, the convergence rate is at least $\mathcal{O}(1/\sqrt{T})$. But we cannot choose a very small η; otherwise, the dominated term $[1 - \nu\gamma(K, \eta)]^T (f(\overline{\mathbf{x}^0}) - \min f)$ will decay very slowly. If the learning rate enjoys the form as $\eta = c_1 \ln^{c_3} T/(LKT^{c_2})$ with $c_1, c_2 > 0$ [2], we can prove the following results on the optimal choices for c_1, c_2, c_3.

Proposition 2. Let $\eta = c_1 \ln^{c_3} T/(LKT^{c_2})$ with $c_1, c_2 > 0$, the optimal rate of DFedAvgM is $\widetilde{\mathcal{O}}(1/T)$, in which case $c_1 = L/\nu$, $c_2 = 1$, and $c_3 = -1$, that is, $\eta = 1/(\nu KT \ln T)$.

This finding coincides with existing results of the optimal rate for SGD with strong convexity [47], [48]. Under the PŁ condition, the convergence rate of DFedAvgM is improved.

Next, we provide the convergence guarantee for the quantized DFedAvgM, which is stated in the following theorem.

Theorem 3. Let the sequence $\{\mathbf{x}^t(i)\}_{t\geq 0}$ be generated by the quantized DFedAvgM for all $i \in \{1, 2 \ldots, m\}$, and all the assumptions in Theorem 1 and Assumption 4 hold. Let $\eta = \Theta(\frac{1}{LK\sqrt{T}})$, as T is sufficiently large, it holds that

$$\min_{1\leq t\leq T} \mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 = \mathcal{O}\Big(\frac{(1-\theta)(f(\overline{\mathbf{x}^1}) - \min f)}{\sqrt{T}} + \frac{(1-\theta)(\sigma_l^2 + K\sigma_g^2) + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2 + B^2)}{K\sqrt{T}} + \frac{(1-\theta)(\sigma_l^2 + K\sigma_g^2 + KB^2) + \frac{\theta^2}{(1-\theta)}K(\sigma_l^2 + B^2)}{(1-\lambda)KT^{3/2}} + \sqrt{T}s\Big).$$

If the function f further satisfies the PŁ condition and $\eta = \frac{1}{\nu TK \ln T}$, it follows that

$$\mathbb{E}(f(\overline{\mathbf{x}^T}) - \min f) = \widetilde{\mathcal{O}}(\frac{1}{T} + Ts).$$

According to Theorem 3, to reach any given $\epsilon > 0$ error in general case, we need to set $s = \mathcal{O}(\epsilon^2)$ and set the the number of communication round as $T = \Theta(\frac{1}{\epsilon^2})$. While with PŁ condition, we set $T = \Theta(\frac{1}{\epsilon})$ and $s = \mathcal{O}(\epsilon^2)$. It follows $\mathbb{E}(f(\overline{\mathbf{x}^T}) - \min f) = \widetilde{\mathcal{O}}(\epsilon)$. Therefore, under the PŁ condition, the number of communication round is reduced.

In the following, we provide a sufficient condition for communications-saving of the two quantization rules mentioned in Sec. 3.2 used in quantized DFedAvgM.

Proposition 3. Assume we use the stochastic or deterministic quantization rule with b bits using stepsize $\eta = \frac{1}{LK\sqrt{T}}$. Assume that the parameters trained in all clients do not overflow, that is, all coordinates are contained in $[-2^{b-1}s, (2^{b-1} - 1)s]$. Let Assumptions 1, 2 and 3 hold. If the desired error

$$\epsilon > (1-\theta)\sqrt{3LBs}d^{\frac{1}{4}} \times \sqrt{2(f(\overline{\mathbf{x}^0}) - \min f) + \frac{8\sigma_l^2}{K} + 32\sigma_g^2 + \frac{64\theta^2(\sigma_l^2 + B^2)}{(1-\theta)^2}}$$

and $b < \frac{128}{9} + \frac{32}{d}$, the quantized DFedAvgM can beat DFedAvgM with 32 bits in term of the required communications to reach ϵ.

Proposition 3 indicates that the superiority of the quantized DFedAvgM retains when the desired error ϵ is not smaller than $\mathcal{O}((1-\theta)\sqrt{s})$. We can also see that as K increases, the guaranteed lower bound of ϵ decreases, which demonstrates the necessity of multiple local iterations. Moreover, a larger θ can also reduce the lower bound.

5 PROOFS

5.1 Technical Lemmas

We define $\mathbf{1} := [1, 1, \ldots, 1]^\top \in \mathbb{R}^m$ and

$$\mathbf{P} := \frac{\mathbf{1}\mathbf{1}^\top}{m} \in \mathbb{R}^{m\times m}.$$

For a matrix $\mathbf{A}$, we denote its spectral norm as $\|\mathbf{A}\|_{\mathrm{op}}$. We also define $\mathbf{X} := \big[\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(m)\big]^\top \in \mathbb{R}^{m\times d}$.

Lemma 1. [Lemma 4 , [4]] For any $k \in \mathbb{Z}^+$, the mixing matrix $\mathbf{W} \in \mathbb{R}^m$ satisfies

$$\|\mathbf{W}^k - \mathbf{P}\|_{op} \leq \lambda^k,$$

where $\lambda := \max\{|\lambda_2|, |\lambda_m(W)|\}$.

In [Proposition 1, [21]], the author also proved that $\|W^k - \mathbf{P}\|_{\mathrm{op}} \leq C\lambda^k$ for some $C > 0$ that depends on the matrix.

2. This learning rate is commonly used in the ML community.

Lemma 2. Assume that Assumptions 2 and 3 hold, and $0 \leq \theta < 1$. Let $(\mathbf{y}^{t,k}(i))_{t\geq 0}$ be generated by the (quantized) DFedAvgM. It then follows

$$\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{y}^{t,k}(i)\|^2 \leq \frac{1}{(1-\theta)^2}(2\eta^2\sigma_l^2+2\eta^2B^2)$$

when $0 \leq k \leq K-1$.

Lemma 3. Given the stepsize $0 < \eta \leq \frac{1}{8LK}$ and $i \in \{1,2,\ldots,m\}$ and assume $(\mathbf{y}^{t,k}(i))_{t\geq 0}$, $(\mathbf{x}^t(i))_{t\geq 0}$ are generated by the (quantized) DFedAvgM for all $i \in \{1,2\ldots,m\}$. If Assumption 3 holds, it then follows

$$\begin{aligned}&\frac{1}{m}\sum_{i=1}^m \mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2\\ &\quad\leq C_1\eta^2+32K^2\eta^2\frac{\sum_{i=1}^m\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m},\end{aligned}$$

where $C_1 := 8K\sigma_l^2+32K^2\sigma_g^2+\frac{64K^2\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2)$ when $0 \leq k \leq K$.

With the fact that $\mathbf{y}^{t,K}(i)=\mathbf{z}^t(i)$, Lemma 3 also holds for $\mathbf{z}^t(i)$.

Lemma 4. Given the stepsize $\eta > 0$ and let $\{\mathbf{x}^t(i)\}_{t\geq 0}$ be generated by DFedAvgM for all $i \in \{1,2\ldots,m\}$. If Assumption 3 holds, we have the following bound

$$\frac{1}{m}\sum_{i=1}^m\mathbb{E}\|\mathbf{x}^t(i)-\overline{\mathbf{x}^t}\|^2 \leq C_2\frac{\eta^2}{1-\lambda}, \tag{9}$$

where $C_2 := 8K\sigma_l^2+32K^2\sigma_g^2+\frac{64K^2\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2)+32K^2B^2$.

Lemma 5. Given the stepsize $\eta > 0$ and assume $\{\mathbf{x}^t(i)\}_{t\geq 0}$ are generated by the quantized DFedAvgM for all $i \in \{1,2\ldots,m\}$. If Assumption 3 holds, it follows that

$$\frac{1}{m}\sum_{i=1}^m\mathbb{E}\|\mathbf{x}^t(i)-\overline{\mathbf{x}^t}\|^2 \leq 2C_2\frac{\eta^2}{1-\lambda}+\frac{2ds^2}{1-\lambda}. \tag{10}$$

5.2 Proof of Technical Lemmas

5.2.1 Proof of Lemma 2

Given any $\psi > 0$, the Cauchy inequality gives us

$$\begin{aligned}&\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{y}^{t,k}(i)\|^2\\ &=\mathbb{E}\|-\eta\tilde{\mathbf{g}}^k(i)+\theta(\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i))\|^2\\ &\overset{a)}{\leq}(1+\psi)\theta^2\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i)\|^2\\ &\quad+(1+\frac{1}{\psi})\eta^2\mathbb{E}\|\tilde{\mathbf{g}}^k(i)-\nabla f_i(\mathbf{y}^{t,k}(i))+\nabla f_i(\mathbf{y}^{t,k}(i))\|^2\\ &\leq(1+\psi)\theta^2\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i)\|^2\\ &\quad+(2+\frac{2}{\psi})\eta^2\|\nabla f_i(\mathbf{y}^{t,k}(i))\|^2\\ &\quad+2(1+\frac{1}{\psi})\eta^2\mathbb{E}\|\tilde{\mathbf{g}}^k(i)-\nabla f_i(\mathbf{y}^{t,k}(i))\|^2,\end{aligned}$$

where $a)$ uses the Cauchy's inequality $\mathbb{E}\|\mathbf{a}+\mathbf{b}\|^2 \leq (1+\frac{1}{\psi})\mathbb{E}\|\mathbf{a}\|^2+(1+\psi)\mathbb{E}\|\mathbf{b}\|^2$ with $\mathbf{a}=-\eta\tilde{\mathbf{g}}^k(i)$ and $\mathbf{b}=\theta(\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i))$. Without loss of generality, we assume $\theta \neq 0$. Let $\psi=\frac{1}{\theta}-1$, we get

$$\begin{aligned}&\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{y}^{t,k}(i)\|^2\\ &\quad\leq\theta\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i)\|+\frac{2\eta^2\sigma_l^2}{1-\theta}+\frac{2\eta^2B^2}{1-\theta}.\end{aligned}$$

Using the mathematical induction, for any integer $0 \leq k \leq K$, we have

$$\begin{aligned}&\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{y}^{t,k}(i)\|^2\\ &\quad\leq\frac{2\eta^2\sigma_l^2+2\eta^2B^2}{1-\theta}(\sum_{i=0}^{k-1}\theta^i)\leq\frac{2\eta^2\sigma_l^2+2\eta^2B^2}{(1-\theta)^2}.\end{aligned}$$

5.2.2 Proof of Lemma 3

Note that for any $k \in \{0,1,\ldots,K-1\}$, in node i it holds

$$\begin{aligned}&\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{x}^t(i)\|^2\\ &=\mathbb{E}\|\mathbf{y}^{t,k}(i)-\eta\tilde{\mathbf{g}}^k(i)-\mathbf{x}^t(i)+\theta(\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i))\|^2\\ &\leq\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)-\eta\Big(\tilde{\mathbf{g}}^k(i)-\nabla f_i(\mathbf{y}^{t,k}(i))+\nabla f_i(\mathbf{y}^{t,k}(i))\\ &\quad-\nabla f_i(\mathbf{x}^t(i))+\nabla f_i(\mathbf{x}^t(i))-\nabla f(\mathbf{x}^t(i))+\nabla f(\mathbf{x}^t(i))\Big)\\ &\quad+\theta(\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i))\|^2\leq \mathrm{I}+\mathrm{II},\end{aligned}$$

where $\mathrm{I}=(1+\frac{1}{2K-1})\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)-\eta(\tilde{\mathbf{g}}^k(i)-\nabla f_i(\mathbf{y}^{t,k}(i))\|^2$ and $\mathrm{II}=2K\eta^2\mathbb{E}\|\nabla f_i(\mathbf{y}^{t,k}(i))-\nabla f_i(\mathbf{x}^t(i))+\nabla f_i(\mathbf{x}^t(i))-\nabla f(\mathbf{x}^t(i))+\nabla f(\mathbf{x}^t(i))+\theta(\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i))\|^2$. The unbiased expectation property of $\tilde{\mathbf{g}}^k(i)$ gives us

$$\begin{aligned}\mathrm{I}=(1+\frac{1}{2K-1})\Big(&\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2\\ &+\eta^2\mathbb{E}\|\tilde{\mathbf{g}}^k(i)-\nabla f_i(\mathbf{y}^{t,k}(i))\|^2\Big).\end{aligned}$$

On the other hand, with Lemma 2, we have the following bound

$$\begin{aligned}\mathrm{II}&\leq 8K\eta^2\mathbb{E}\|\nabla f_i(\mathbf{y}^{t,k}(i))-\nabla f_i(\mathbf{x}^t(i))\|\\ &+8K\eta^2\mathbb{E}\|\nabla f_i(\mathbf{x}^t(i))-\nabla f(\mathbf{x}^t(i))\|\\ &+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2+8K\theta^2\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{y}^{t,k-1}(i)\|^2\\ &\leq 8L^2K\eta^2\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2+8K\eta^2\sigma_g^2\\ &+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2+\frac{16K\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2).\end{aligned}$$

Thus, we can obtain

$$\begin{aligned}&\mathbb{E}\|\mathbf{y}^{t,k+1}(i)-\mathbf{x}^t(i)\|^2\\ &\leq(1+\frac{1}{2K-1}+8L^2K\eta^2)\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2+2\eta^2\sigma_l^2\\ &+8K\eta^2\sigma_g^2+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2+\frac{16K\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2)\\ &\leq(1+\frac{1}{K-1})\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2+2\eta^2\sigma_l^2+8K\eta^2\sigma_g^2\\ &+\frac{16K\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2)+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2,\end{aligned}$$

where the last inequality depends on the selection of the stepsize. The recursion from $j=0$ to k yeilds

$$\begin{aligned}
&\mathbb{E}\|\mathbf{y}^{t,k}(i)-\mathbf{x}^t(i)\|^2\\
&\le \sum_{j=0}^{K-1}(1+\frac{1}{K-1})^j\Big[2\eta^2\sigma_l^2+8K\eta^2\sigma_g^2\\
&\quad+\frac{16K\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2)+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2\Big]\\
&\le (K-1)\Big[(1+\frac{1}{K-1})^K-1\Big]\times\Big[2\eta^2\sigma_l^2+8K\eta^2\sigma_g^2\\
&\quad+\frac{16K\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2)+8K\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2\Big]\\
&\le 8K\eta^2\sigma_l^2+32K^2\eta^2\sigma_g^2+\frac{64K^2\theta^2}{(1-\theta)^2}(\eta^2\sigma_l^2+\eta^2B^2)\\
&\quad+32K^2\eta^2\mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2,
\end{aligned}$$

where we used the inequality $(1+\frac{1}{K-1})^K\le 5$ holds for any $K\ge 1$.

5.2.3 Proof of Lemma 4

We denote $\mathbf{Z}^t:=\left[\mathbf{z}^t(1),\mathbf{z}^t(2),\ldots,\mathbf{z}^t(m)\right]^\top\in\mathbb{R}^{m\times d}$. With these notation, we have

$$\mathbf{X}^{t+1}=\mathbf{W}\mathbf{Z}^t=\mathbf{W}\mathbf{X}^t-\zeta^t, \tag{11}$$

where $\zeta^t:=\mathbf{W}\mathbf{X}^t-\mathbf{W}\mathbf{Z}^t$. The iteration (11) can be rewritten as the following expression

$$\mathbf{X}^t=W^t\mathbf{X}^0-\sum_{j=0}^{t-1}\mathbf{W}^{t-1-j}\zeta^j. \tag{12}$$

Obviously, it follows

$$\mathbf{W}\mathbf{P}=\mathbf{P}\mathbf{W}=\mathbf{P}. \tag{13}$$

According to Lemma 1, it holds

$$\|\mathbf{W}^t-\mathbf{P}\|\le\lambda^t.$$

Multiplying both sides of (12) with $\mathbf{P}$ and using (13), we then get

$$\mathbf{P}\mathbf{X}^t=\mathbf{P}\mathbf{X}^0-\sum_{j=0}^{t-1}\mathbf{P}\zeta^j=-\sum_{j=0}^{t-1}\mathbf{P}\zeta^j, \tag{14}$$

where we used initialization $\mathbf{X}^0=\mathbf{0}$. Then, we are led to

$$\begin{aligned}
&\|\mathbf{X}^t-\mathbf{P}\mathbf{X}^t\|=\|\sum_{j=0}^{t-1}(\mathbf{P}-\mathbf{W}^{t-1-j})\zeta^j\|\\
&\le\sum_{j=0}^{t-1}\|\mathbf{P}-\mathbf{W}^{t-1-j}\|_{\text{op}}\|\zeta^j\|\le\sum_{j=0}^{t-1}\lambda^{t-1-j}\|\zeta^j\|.
\end{aligned} \tag{15}$$

With Cauchy inequality,

$$\begin{aligned}
&\mathbb{E}\|\mathbf{X}^t-\mathbf{P}\mathbf{X}^t\|^2\le\mathbb{E}(\sum_{j=0}^{t-1}\lambda^{\frac{t-1-j}{2}}\cdot\lambda^{\frac{t-1-j}{2}}\|\zeta^j\|)^2\\
&\le(\sum_{j=0}^{t-1}\lambda^{t-1-j})(\sum_{j=0}^{t-1}\lambda^{t-1-j}\mathbb{E}\|\zeta^j\|^2)
\end{aligned}$$

Direct calculation gives us

$$\mathbb{E}\|\zeta^j\|^2\le\|\mathbf{W}\|^2\cdot\mathbb{E}\|\mathbf{X}^j-\mathbf{Z}^j\|^2\le\mathbb{E}\|\mathbf{X}^j-\mathbf{Z}^j\|^2.$$

With Lemma 3 and Assumption 3, for any j,

$$\begin{aligned}
&\mathbb{E}\|\mathbf{X}^j-\mathbf{Z}^j\|^2\\
&\le m(8K\sigma_l^2+32K^2\sigma_g^2+\frac{64K\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2)+32K^2B^2)\eta^2.
\end{aligned}$$

Thus, we get

$$\begin{aligned}
&\mathbb{E}\|\mathbf{X}^t-\mathbf{P}\mathbf{X}^t\|^2\\
&\le\frac{m(8K\sigma_l^2+32K^2\sigma_g^2+\frac{64K\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2)+32K^2B^2)\eta^2}{1-\lambda}.
\end{aligned}$$

The fact that $\mathbf{X}^t-\mathbf{P}\mathbf{X}^t=\begin{pmatrix}\mathbf{x}^t(1)-\overline{\mathbf{x}^t}\\ \mathbf{x}^t(2)-\overline{\mathbf{x}^t}\\ \vdots\\ \mathbf{x}^t(m)-\overline{\mathbf{x}^t}\end{pmatrix}$ then proves the result.

5.2.4 Proof of Lemma 5

Let $\widetilde{\mathbf{Z}}^t:=\mathbf{Y}^{t,K}$. Obviously, it holds

$$\mathbf{X}^{t+1}=\mathbf{W}\mathbf{X}^t-\widetilde{\zeta}^t, \tag{16}$$

where $\widetilde{\zeta}^t=\mathbf{W}\mathbf{X}^t-\mathbf{W}(\mathbf{Q}(\widetilde{\mathbf{Z}}^t-\mathbf{X}^t)+\mathbf{X}^t)$. We just need to bound $\mathbb{E}\|\widetilde{\zeta}^t\|^2$,

$$\begin{aligned}
&\mathbb{E}\|\widetilde{\zeta}^t\|^2\le 2\mathbb{E}\|\mathbf{W}\mathbf{X}^t-\mathbf{W}\widetilde{\mathbf{Z}}^t\|^2\\
&\quad+2\mathbb{E}\|\mathbf{W}\widetilde{\mathbf{Z}}^t-\mathbf{W}(\mathbf{Q}(\widetilde{\mathbf{Z}}^t-\mathbf{X}^t)+\mathbf{X}^t)\|^2\\
&\le 2\mathbb{E}\|\mathbf{X}^t-\widetilde{\mathbf{Z}}^t\|^2+2\mathbb{E}\|\mathbf{W}(\widetilde{\mathbf{Z}}^t-\mathbf{X}^t)-\mathbf{W}(\mathbf{Q}(\widetilde{\mathbf{Z}}^t-\mathbf{X}^t))\|^2\\
&\le 2\mathbb{E}\|\mathbf{X}^t-\widetilde{\mathbf{Z}}^t\|^2+2mds^2\\
&\le 2m\eta^2(8K\sigma_l^2+32K^2\sigma_g^2+\frac{64K\theta^2}{(1-\theta)^2}(\sigma_l^2+B^2)+32K^2B^2)\\
&\quad+2mds^2,
\end{aligned}$$

the last inequality uses Lemma 3.

5.3 Proof of Theorem 1

Noting that $\mathbf{P}\mathbf{X}^{t+1}=\mathbf{P}\mathbf{W}\mathbf{Z}^t=\mathbf{P}\mathbf{Z}^t$, that is also

$$\overline{\mathbf{x}^{t+1}}=\overline{\mathbf{z}^t},$$

we have

$$\overline{\mathbf{x}^{t+1}}-\overline{\mathbf{x}^t}=\overline{\mathbf{x}^{t+1}}-\overline{\mathbf{z}^t}+\overline{\mathbf{z}^t}-\overline{\mathbf{x}^t}=\overline{\mathbf{z}^t}-\overline{\mathbf{x}^t}, \tag{17}$$

where $\overline{\mathbf{z}^t}:=\frac{\sum_{i=1}^m\mathbf{z}^t(i)}{m}$. With the local scheme in each node,

$$\begin{aligned}
\overline{\mathbf{z}^t}-\overline{\mathbf{x}^t}&=\frac{\sum_{i=1}^m(\mathbf{z}^t(i)-\mathbf{x}^t(i))}{m}\\
&=\frac{\sum_{i=1}^m(\sum_{k=0}^{K-1}\mathbf{y}^{t,k+1}(i)-\mathbf{y}^{t,k}(i))}{m}\\
&=-\eta\frac{\sum_{i=1}^m\sum_{k=0}^{K-1}\nabla f_i(\mathbf{y}^{t,k}(i))}{m}+\theta(\overline{\mathbf{z}^t}-\overline{\mathbf{x}^t})\\
&\quad+\theta\eta\frac{\sum_{i=1}^m\nabla f_i(\mathbf{y}^{t,K-1}(i))}{m}.
\end{aligned}$$

Thus, we get

$$\begin{aligned}
\overline{\mathbf{z}^t}-\overline{\mathbf{x}^t}&=\frac{1}{1-\theta}(-\eta\frac{\sum_{i=1}^m\sum_{k=0}^{K-2}\nabla f_i(\mathbf{y}^{t,k}(i))}{m})\\
&=\frac{1}{1-\theta}(-\eta\frac{\sum_{i=1}^m(1-\theta)\nabla f_i(\mathbf{y}^{t,K-1}(i))}{m}).
\end{aligned} \tag{18}$$

The Lipschitz continuity of ∇f gives us

$$\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) \leq \mathbb{E}f(\overline{\mathbf{x}^t}) + \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \overline{\mathbf{z}^t} - \overline{\mathbf{x}^t}\rangle + \frac{L}{2}\mathbb{E}\|\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t}\|^2, \tag{19}$$

where we used (17). Let

$$\tilde{K} = \frac{K-\theta}{1-\theta},$$

we can derive

$$\begin{aligned}
&\mathbb{E}\langle \tilde{K}\nabla f(\overline{\mathbf{x}^t}), (\overline{\mathbf{z}^t} - \overline{\mathbf{x}^t})/\tilde{K}\rangle \\
&= \mathbb{E}\langle \tilde{K}\nabla f(\overline{\mathbf{x}^t}), -\eta\nabla f(\overline{\mathbf{x}^t}) + \eta\nabla f(\mathbf{x}^t) + (\overline{\mathbf{z}^t} - \overline{\mathbf{x}^t})/\tilde{K}\rangle \\
&= -\eta\tilde{K}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 + \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \eta\nabla f(\mathbf{x}^t) + (\overline{\mathbf{z}^t} - \overline{\mathbf{x}^t})/\tilde{K}\rangle \\
&\overset{a)}{\leq} -\eta\tilde{K}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \\
&\quad + \eta\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\| \cdot \Big\| \frac{\sum_{i=1}^m \sum_{k=0}^{K-2}[\nabla f_i(\overline{\mathbf{x}^t}) - \nabla f_i(\mathbf{y}^{t,k}(i))]}{m} \\
&\quad + \frac{\sum_{i=1}^m (1-\theta)[\nabla f_i(\overline{\mathbf{x}^t}) - \nabla f_i(\mathbf{y}^{t,K-1}(i))]}{m}\Big) \Big\| \\
&\leq -\eta\tilde{K}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 + \frac{\eta L}{m}\sum_{i=1}^m\sum_{k=0}^{K-1}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\| \cdot \|\overline{\mathbf{x}^t} - \mathbf{y}^{t,k}(i)\| \\
&\leq -\eta\tilde{K}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 + \frac{\eta\tilde{K}}{2}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \\
&\quad + \frac{\eta L^2K^2}{2\tilde{K}}(C_1\eta^2 + 32K^2\eta^2\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m}),
\end{aligned}$$

where $a)$ uses (18). Similarly, we can get

$$\begin{aligned}
&\frac{L}{2}\mathbb{E}(\|\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t}\|^2) = \frac{L}{2}\mathbb{E}(\|\overline{\mathbf{z}^t} - \overline{\mathbf{x}^t}\|^2) \\
&\leq \frac{L}{2}\frac{1}{m}\sum_{i=1}^m \|\mathbf{z}^t(i) - \mathbf{x}^t(i)\|^2 \\
&\leq \frac{L}{2}C_1\eta^2 + 16LK^2\eta^2\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m}.
\end{aligned}$$

Thus, (19) can be represented as

$$\begin{aligned}
&\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) \leq \mathbb{E}f(\overline{\mathbf{x}^t}) - \frac{\eta\tilde{K}}{2}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 + \frac{L^2K^2}{2\tilde{K}}C_1\eta^3 \\
&+ \frac{L}{2}C_1\eta^2 + (\frac{16L^2K^4\eta^3}{\tilde{K}} + 16LK^2\eta^2)\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m}.
\end{aligned}$$

Direct computation together with Lemma 4 gives us

$$\begin{aligned}
&\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m} \\
&= \frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i)) - \nabla f(\overline{\mathbf{x}^t}) + \nabla f(\overline{\mathbf{x}^t})\|^2}{m} \\
&\leq \frac{\sum_{i=1}^m 2\mathbb{E}\|\nabla f(\mathbf{x}^t(i)) - \nabla f(\overline{\mathbf{x}^t})\|^2 + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2}{m} \\
&\leq 2L^2\frac{\sum_{i=1}^m \|\mathbf{x}^t(i) - \overline{\mathbf{x}^t}\|^2}{m} + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \\
&\leq \frac{2L^2C_2\eta^2}{1-\lambda} + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2.
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
&\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) \leq \mathbb{E}f(\overline{\mathbf{x}^t}) \\
&- (\frac{\eta(K-\theta)}{2(1-\theta)} - \frac{32(1-\theta)L^2K^4\eta^3}{K-\theta} - 32LK^2\eta^2) \\
&\quad \times \mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 + (\frac{(1-\theta)L^2K^2}{2(K-\theta)}\eta^3 + \frac{L}{2}\eta^2) \\
&\quad \times (8K\sigma_l^2 + 32K^2\sigma_g^2 + \frac{64K^2\theta^2}{(1-\theta)^2}(\sigma_l^2 + B^2)) \\
&+ (32(1-\theta)L^4K^4\eta^5/(K-\theta) + 32L^3K^2\eta^4)/(1-\lambda) \\
&\quad \times (8K\sigma_l^2 + 32K^2\sigma_g^2 + 32K^2B^2 + \frac{64K^2\theta^2}{(1-\theta)^2}(\sigma_l^2 + B^2)).
\end{aligned} \tag{20}$$

Summing the inequality (20) from $t = 1$ to T, we then proved the result.

5.4 Proof of Theorem 2

With the PŁ condition,

$$\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \geq 2\nu\mathbb{E}(f(\overline{\mathbf{x}^t}) - \min f).$$

We start from (20),

$$\begin{aligned}
\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) &\leq \mathbb{E}f(\overline{\mathbf{x}^t}) - \nu\gamma(K,\eta)\mathbb{E}(f(\overline{\mathbf{x}^t}) - \min f) \\
&+ \frac{\gamma(K,\eta)\alpha(K,\eta)}{2} + \frac{\gamma(K,\eta)\beta(K,\eta,\lambda)}{2}.
\end{aligned} \tag{21}$$

By defining $\xi_t := \mathbb{E}(f(\overline{\mathbf{x}^t}) - \min f)$, it then follows

$$\begin{aligned}
\xi_{t+1} &\leq [1 - \nu\gamma(K,\eta)]\xi_t \\
&+ \frac{\gamma(K,\eta)\alpha(K,\eta)}{2} + \frac{\gamma(K,\eta)\beta(K,\eta,\lambda)}{2}.
\end{aligned} \tag{22}$$

Thus, we are then led to

$$\begin{aligned}
\xi_T &\leq [1 - \nu\gamma(K,\eta)]^T\xi_0 \\
&+ \Big(\frac{\gamma(K,\eta)\alpha(K,\eta)}{2} + \frac{\gamma(K,\eta)\beta(K,\eta,\lambda)}{2}\Big) \\
&\quad \times (\sum_{t=0}^{T-1}[1 - \nu\gamma(K,\eta)]^t) \\
&\leq [1 - \nu\gamma(K,\eta)]^T\xi_0 \\
&+ \Big(\frac{\gamma(K,\eta)\alpha(K,\eta)}{2} + \frac{\gamma(K,\eta)\beta(K,\eta,\lambda)}{2}\Big)\frac{1}{\nu\gamma(K,\eta)} \\
&= [1 - \nu\gamma(K,\eta)]^T\xi_0 + \frac{\alpha(K,\eta)}{2\nu} + \frac{\beta(K,\eta,\lambda)}{2\nu}.
\end{aligned}$$

The result is then proved.

5.5 Proof of Proposition 2

A quick calculation gives us

$$\begin{cases} \gamma(K,\eta) = \Theta(\frac{1}{T^{c_2}}), \\ \frac{\alpha(K,\eta)}{2\nu} + \frac{\beta(K,\eta,\lambda)}{2\nu} = O(\frac{1}{T^{c_2}}). \end{cases}$$

Thus, we just need to bound the first term in Theorem 2. As T is large, $\gamma(K,\eta) \to 0$. Its logarithm is then

$$T\log[1 - \nu\gamma(K,\eta)] = T\log[1 - \nu\gamma(K,\eta)] = \Theta(-T\nu\gamma(K,\eta)).$$

With our setting, it follows

$$T\nu\gamma(K,\eta) \approx \frac{\nu c_1 \ln^{c_3} T}{LT^{c_2-1}}.$$

Then we have

$$\mathbb{E}f(\overline{\mathbf{x}^T}) - \min f = \mathcal{O}(\exp(-\frac{\nu c_1 \ln^{c_3} T}{LKT^{c_2-1}}) + \frac{1}{T^{c_2}}).$$

We first consider how to choose c_2. From the L'Hospital's rule, for any $\delta > 0$, as $T \to +\infty$

$$\exp(-\frac{1}{T^{\delta}}) \to 1.$$

Thus, we need to set $c_2 \leq 1$ and the fast rate is slower than $O(\frac{1}{T})$. To this end, we choose $c_1 = \frac{L}{\nu}$, $c_2 = 1$, and $c_3 = -1$.

5.6 Proof of Theorem 3

Let $\widetilde{\mathbf{y}}^t := \frac{\sum_{i=1}^m \mathbf{y}^{t,K(i)}}{m}$, in the quantized DFedAvgM, it follows $\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t} = Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})$. With the Lipschitz continuity of ∇f,

$$\begin{aligned}\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) &\leq \mathbb{E}f(\overline{\mathbf{x}^t}) \\ &\quad + \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})\rangle + \frac{L}{2}\mathbb{E}\|\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t}\|^2,\end{aligned}$$

We have

$$\begin{aligned}&\mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})\rangle \\ &= \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t}\rangle + \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t} - Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})\rangle \\ &\leq \mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t}\rangle + B\sqrt{d}s\end{aligned}$$

and

$$\begin{aligned}&\frac{L}{2}\mathbb{E}\|\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t}\|^2 = \frac{L}{2}\mathbb{E}\|Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})\|^2 \\ &\leq L\mathbb{E}\|\widetilde{\mathbf{y}^t} - \overline{\mathbf{x}^t}\|^2 + L\mathbb{E}\|Q(\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t}) - (\widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t})\|^2 \\ &\leq L\mathbb{E}\|\widetilde{\mathbf{y}^t} - \overline{\mathbf{x}^t}\|^2 + \frac{Lds^2}{m}.\end{aligned}$$

Note that both $\mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \widetilde{\mathbf{y}}^t - \overline{\mathbf{x}^t}\rangle$ and $\mathbb{E}\|\widetilde{\mathbf{y}^t} - \overline{\mathbf{x}^t}\|^2$ can inherit the bounds of $\mathbb{E}\langle \nabla f(\overline{\mathbf{x}^t}), \overline{\mathbf{z}^t} - \overline{\mathbf{x}^t}\rangle$ and $\mathbb{E}\|\overline{\mathbf{x}^{t+1}} - \overline{\mathbf{x}^t}\|^2$ in the proof of Theorem 1. Thus, we obtain

$$\begin{aligned}\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) &\leq \mathbb{E}f(\overline{\mathbf{x}^t}) - \frac{\eta\tilde{K}}{2}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \\ &+ \frac{L^2K^2}{2\tilde{K}}C_1\eta^3 + LC_1\eta^2 + \frac{Lds^2}{m} + B\sqrt{d}s \\ &+ (\frac{32L^2K^4\eta^3}{\tilde{K}} + 32LK^2\eta^2)\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m}.\end{aligned}$$

With Lemma 5, we can get

$$\begin{aligned}&\frac{\sum_{i=1}^m \mathbb{E}\|\nabla f(\mathbf{x}^t(i))\|^2}{m} \\ &\leq \frac{\sum_{i=1}^m 2\mathbb{E}\|\nabla f(\mathbf{x}^t(i)) - \nabla f(\overline{\mathbf{x}^t})\|^2 + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2}{m} \\ &\leq 2L^2\frac{\sum_{i=1}^m \|\mathbf{x}^t(i) - \overline{\mathbf{x}^t}\|^2}{m} + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 \\ &\leq \frac{2L^2C_3\eta^2}{1-\lambda} + \frac{4L^2ds^2}{1-\lambda} + 2\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2.\end{aligned}$$

Combining the inequalities together,

$$\begin{aligned}\mathbb{E}f(\overline{\mathbf{x}^{t+1}}) &\leq \mathbb{E}f(\overline{\mathbf{x}^t}) + \zeta(K,\eta,\lambda,s) \\ &\quad - (\frac{\eta\tilde{K}}{2} - \frac{64L^2K^4\eta^3}{\tilde{K}} - 64LK^2\eta^2)\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2,\end{aligned}$$

where $\zeta(K,\eta,\lambda,s) := \frac{L^2K^2}{2\tilde{K}}C_1\eta^3 + LC_1\eta^2 + (\frac{32L^2K^4\eta^3}{\tilde{K}} + 32LK^2\eta^2)(\frac{2L^2C_3\eta^2}{1-\lambda} + \frac{4L^2ds^2}{1-\lambda}) + \frac{Lds^2}{m} + B\sqrt{d}s$. Given the stepsize $\eta = \Theta(\frac{1}{LK\sqrt{T}})$, we can see that $\frac{\eta\tilde{K}}{2} - \frac{64L^2K^4\eta^3}{\tilde{K}} - 64LK^2\eta^2 > 0$ as T is large. When $s > 0$ is small, $s^2 = O(s)$. And it then follows $\frac{\eta\tilde{K}}{2} - \frac{64L^2K^4\eta^3}{\tilde{K}} - 64LK^2\eta^2 = \Theta(\frac{1}{(1-\theta)\sqrt{T}})$. We now consider

$$\begin{aligned}&\Big[(\frac{32L^2K^4\eta^3}{\tilde{K}} + 32LK^2\eta^2)(\frac{2L^2C_3\eta^2}{1-\lambda} + \frac{4L^2ds}{1-\lambda}) \\ &\quad + LC_1\eta^2 + \frac{L^2K^2}{2\tilde{K}}C_1\eta^3 + \frac{Lds^2}{m} + B\sqrt{d}s\Big]\Big/\Big[\frac{\eta\tilde{K}}{2} \\ &\quad - \frac{64L^2K^4\eta^3}{\tilde{K}} - 64LK^2\eta^2\Big],\end{aligned} \tag{23}$$

which is at the order as $O\Big(\frac{1}{\sqrt{T}} + \frac{\sigma_l^2 + K\sigma_g^2 + \frac{\theta^2}{(1-\theta)^2}K(\sigma_l^2+B^2)}{K\sqrt{T}} + \frac{\sigma_l^2 + K\sigma_g^2 + KB^2 + \frac{\theta^2}{(1-\theta)^2}K(\sigma_l^2+B^2)}{(1-\lambda)KT^{3/2}} + \sqrt{T}s\Big)$. If the function f satisfies the PŁ-ν, we have

$$\begin{aligned}&\mathbb{E}(f(\overline{\mathbf{x}^t}) - \min f) \\ &\leq [1-\nu\gamma(K,\eta)]^T(f(\overline{\mathbf{x}^0}) - \min f) + \frac{\zeta(K,\eta,\lambda,s)}{\nu\gamma(K,\eta)}.\end{aligned}$$

When $\eta = \frac{1}{\nu TK\ln T}$, $[1-\nu\gamma(K,\eta)]^T = \widetilde{\mathcal{O}}(\frac{1}{T})$ and

$$\frac{\zeta(K,\eta,\lambda,s)}{\nu\gamma(K,\eta)} = \widetilde{\mathcal{O}}(\frac{1}{T} + Ts).$$

5.7 Proof of Proposition 3

We calculate to reach the same error, the communication costs by both algorithms. Omitting the order larger than 1 for η, from (20), we have

$$\begin{aligned}\min_{1\leq t\leq T}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 &\approx \frac{2(f(\overline{\mathbf{x}^0}) - \min f)}{\eta KT} \\ &+ L\eta(8\sigma_l^2 + 32K\sigma_g^2 + \frac{64K\theta^2}{(1-\theta)^2}(\sigma_l^2 + B^2)) \\ &= \frac{2(1-\theta)(f(\overline{\mathbf{x}^0}) - \min f)}{\sqrt{T}} \\ &+ \frac{8(1-\theta)\sigma_l^2 + 32(1-\theta)K\sigma_g^2 + \frac{64K\theta^2}{(1-\theta)}(\sigma_l^2 + B^2)}{K\sqrt{T}}.\end{aligned}$$

for DFedAvgM. From (23),

$$\begin{aligned}\min_{1\leq t\leq T}\mathbb{E}\|\nabla f(\overline{\mathbf{x}^t})\|^2 &\approx \frac{2(1-\theta)(f(\overline{\mathbf{x}^0}) - \min f)}{\sqrt{T}} \\ &+ \frac{8(1-\theta)\sigma_l^2 + 32(1-\theta)K\sigma_g^2 + \frac{64K\theta^2}{(1-\theta)}(\sigma_l^2 + B^2)}{K\sqrt{T}} \\ &+ 2(1-\theta)LB\sqrt{d}\sqrt{T}s.\end{aligned}$$

Given the $\epsilon > 0$, assume T_ϵ obeys

$$\begin{aligned}&\frac{8(1-\theta)\sigma_l^2 + 32(1-\theta)K\sigma_g^2 + \frac{64K\theta^2}{(1-\theta)}(\sigma_l^2 + B^2)}{K\sqrt{T_\epsilon}} \\ &+ \frac{2(1-\theta)(f(\overline{\mathbf{x}^0}) - \min f)}{\sqrt{T_\epsilon}} = \epsilon.\end{aligned}$$

That means DFedAvgM can output an ϵ error in T_ϵ iterations. However, due to the error caused by the quantization, we have to increase the iteration number for quantized DFedAvgM. We set the iteration number as $\frac{9}{4}T_\epsilon$. To get the ϵ error for quantized DFedAvgM, we also need $3(1-\theta)LB\sqrt{d}\sqrt{T_\epsilon}s \leq \epsilon$, which yields

$$3(1-\theta)^2 LB\sqrt{d}s\Big[2(f(\overline{\mathbf{x}^0}) - \min f) + \frac{8\sigma_l^2}{K} + 32\sigma_g^2 + \frac{64\theta^2}{(1-\theta)^2}(\sigma_l^2 + B^2)\Big] \leq \epsilon^2.$$

The total communication cost of DFedAvgM to reach ϵ is

$$32dT_\epsilon \sum_{i=1}^{m}[\deg(\mathcal{N}(i))].$$

While for quantized version, the total communication cost is

$$(32+db)\frac{9}{4}T_\epsilon \sum_{i=1}^{m}[\deg(\mathcal{N}(i))].$$

Thus, the communications can be reduced if

$$(32+db)\frac{9}{4} < 32d.$$

6 Numerical Results

We apply the proposed DFedAvgM with communication quantization to train DNNs for both image classification and language modeling, where we consider a simple ring structured communication network. We aim to verify that DFedAvgM can train DNNs effectively, especially in communication efficiency. Moreover, we consider the membership privacy protection when DFedAvgM is used for training DNNs. We apply the membership inference attack (MIA) [49] to test the efficiency of (quantized) DFedAvgM in protecting the training data's MP. In MIA, the attack model is a binary classifier [3], which is to decide if a data point is in the training set of the target model. For each of the following dataset, to perform MIA we first split its training set into D_{shadow} and D_{target} with the same size. Furthermore, we split D_{shadow} into two halves with the same size and denote them as $D_{\text{shadow}}^{\text{train}}$ and $D_{\text{shadow}}^{\text{out}}$, and we split D_{target} by half into $D_{\text{target}}^{\text{train}}$ and $D_{\text{target}}^{\text{out}}$. MIA proceeds as follows: 1) train the shadow model by using $D_{\text{shadow}}^{\text{train}}$; 2) apply the trained shadow model to predict all data points in D_{shadow} and train the corresponding classification probabilities of belonging to each class. Then we take the top three classification probabilities (or two in the case of binary classification) to form the feature vector for each data point.A feature vector is tagged as1if the corresponding data point is in $D_{\text{shadow}}^{\text{train}}$, and 0 otherwise. Then we train the attack model by leveraging all the labeled feature vectors; 3) train the target model by using $D_{\text{target}}^{\text{train}}$ and obtain feature vector for each point in D_{target}. Finally, we leverage the attack model to decide if a data point is in $D_{\text{target}}^{\text{train}}$. Note the attack model we build is a binary classifier, which is to decide if a data point is in the training set of the target model. For any data $\xi \in D_{\text{target}}$, we apply the attack model to predict its probability (p) of belonging to the training set of the target model. Given any fixed threshold t if $p \geq t$, we classify ξ as a member of the training set (positive sample), and if $p < t$, we conclude that ξ is not in the training set (negative sample); so we can obtain different attack results with different thresholds. We can plot the ROC curve for different threshold, and regard the area under the ROC curve (AUC) as an evaluation of the membership inference attack.The target model protects perfect membership privacy if the AUC is 0.5 (attack model performs random guess), and the higher AUC is, the less private the target model is.

3. We use a multilayer perceptron with a hidden layer of 64 nodes, followed by a softmax output function as the attack model, which is adapted from [49].

6.1 MNIST Classification

6.1.0.1 The efficiency of DFedAvgM.: We train two DNNs for MNIST classification using 100 clients: 1) A simple multilayer-perceptron with 2-hidden layers with 200 units each using ReLU activation (199,210 total parameters), which we refer to as 2NN. 2) A CNN with two 5×5 convolution layers (the first with 32 channels, the second with 64, each followed with 2×2 max pooling), a fully connected layer with 512 units and ReLU activation, and the final output layer (1,663,370 total parameters). We study two partitioning of the MNIST data over clients, i.e., IID and Non-IID. In IID setting, the data is shuffled, and then partitioned into 20 clients each receiving 3000 examples. In Non-IID, we first sort the data by digit label, divide it into 40 shards of size 1500, and assign each of 20 clients 2 shards. In training, we set the local batch size (batch size of the training data on clients) to be 50, learning rate 0.01, and momentum 0.9. Figures 2 and 3 show the results of training CNN for MNIST classification (Fig. 2: IID and Fig. 3 Non-IID) by DFedAvgM using different communication bits and different local epochs. These results confirm the efficiency of DFedAvgM for training DNNs; in particular, when the clients' data are IID. For both IID and Non-IID settings, the communication bits do not affect the performance of DFedAvgM; as we see that the training loss, test accuracy, and AUC under the membership inference attack are almost identical. Increasing local training epochs can accelerate training for IID setting at the cost of faster privacy leakage. However, for Non-IID, increasing local training epochs does not help DFedAvgM in either training or privacy protection. Training 2NN by DFedAvgM behaves similarly, see Figs. 4 and 5.

6.1.0.2 Comparison between DFedAvgM, FedAvg, and DSGD.: Now, we compare the DFedAvgM, FedAvg, and DSGD in training 2NNs for MNIST classification. We use the same local batch size 50 for both FedAvg and DSGD, and the learning rates are both set to 0.1[4]. For FedAvg, we select all clients to get involved in training and communication in each round. Figure 6 compares three algorithms in terms of test loss and test accuracy for IID MNIST over communication round and communication cost. In terms of communication rounds, DFedAvgM converges as fast as FedAvg, and both are much faster than DSGD. DFedAvgM has a significant advantage over FedAvg and DSGD in communication costs. For Non-IID MNIST, training 2NN by

4. We note that DFedAvg requires smaller learning rates than FedAvg and DSGD for numerical convergence.

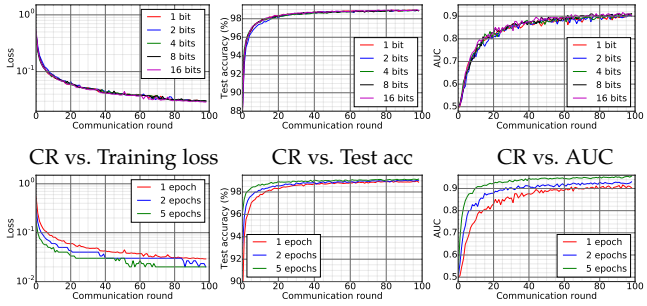


Fig. 2. Training CNN for IID MNIST classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Different quantized DFedAvgM performs almost similar, and more local epoch can accelerate training at the cost of faster privacy leakage. CR: communication round.

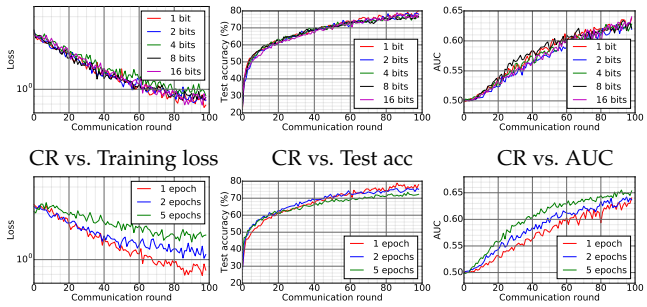


Fig. 3. Training CNN for Non-IID MNIST classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Different quantized DFedAvgM does not lead to much difference in performance. More local epoch does not help in accelerating training or protect data privacy.

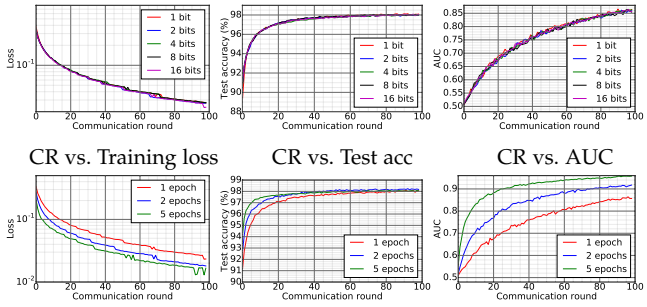


Fig. 4. Training 2NN for IID MNIST classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Different quantized DFedAvgM performs almost similar, and more local epoch can accelerate training at the cost of faster privacy leakage.

FedAvg can achieve 96.81% test accuracy, but both DFedAvg and DSGD cannot bypass 85%. This disadvantage is because both DSGD and DFedAvgM only communicate with their neighbors, while the neighbors and itself may not contain enough training data to cover all possible classes. One feasible solution to resolve the issues of DFedAvgM for the Non-IID setting is by designing a new graph structure for more efficient global communication.

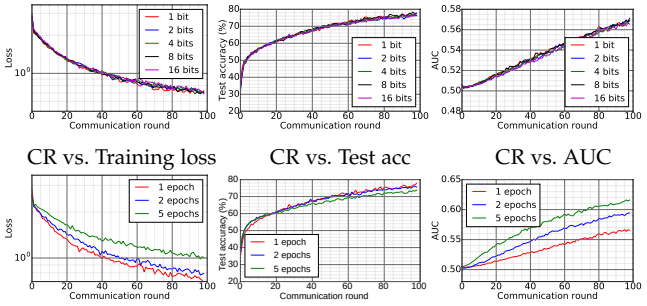


Fig. 5. Training 2NN for Non-IID MNIST classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Different quantized DFedAvgM does not lead to much difference in performance. More local epoch does not help in accelerating training or protect data privacy.

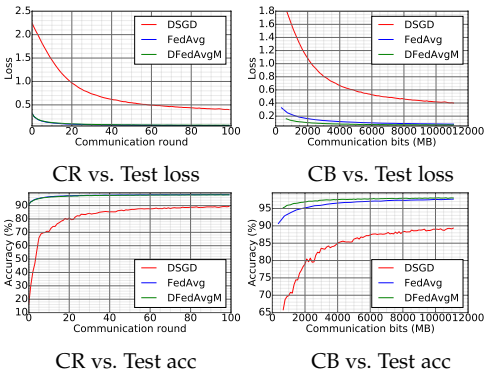
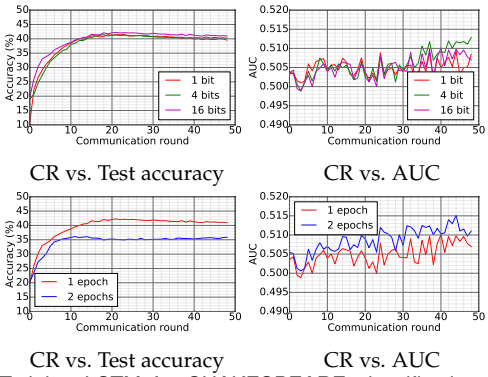


Fig. 6. The efficiency comparison between DSGD, FedAvg, and DFedAvgM in training 2NN for MNIST classification. (a) and (c): test loss and test accuracy vs. communication round. (b) and (d): test loss and test accuracy vs. communication bits. DFedAvgM performs on par with FedAvg in terms of communication rounds, but DFedAvgM is significantly more efficient than FedAvg from the communication cost viewpoint. CR: communication round; CB: communication bits.

6.2 LSTM for Language Modeling

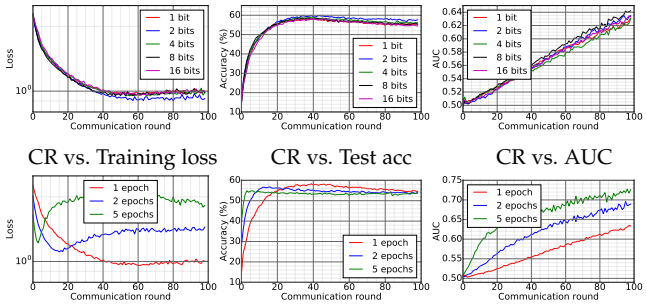
We consider the SHAKESPEARE dataset and we follow the processing as that used in [1], resulting in a dataset distributed over 1146 clients in the Non-IID fashion. On this data, we use DFedAvgM to train a stacked character-level LSTM language model, which predicts the next character after reading each character in a line. The model takes a series of characters as input and embeds each of these into a learned 8-dimensional space. The embedded characters are then processed through 2 LSTM layers, each with 256 nodes. Finally, the output of the second LSTM layer is sent to a softmax output layer with one node per character. The full model has 866,578 parameters, and we trained using an unroll length of 80 characters. We set the local batch size to 10, and we use a learning rate of 1.47, which is the same as [1]. The momentum is selected to be 0.9. Figure 7 plots the communication round vs. test accuracy and AUC under MIA for different quantization and different local epochs. These results show that 1) both the accuracy and MIA increase as training goes; 2) higher communication cost can lead to faster convergence; 3) increase local epochs deteriorate the performance of DFedAvgM.



CR vs. Test accuracy CR vs. AUC

CR vs. Test accuracy CR vs. AUC

Fig. 7. Training LSTM for SHAKESPEARE classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Using higher precision communication can slightly improve the performance. More local epoch does not help in accelerating training or protect data privacy.



CR vs. Training loss CR vs. Test acc CR vs. AUC

CR vs. Training loss CR vs. Test acc CR vs. AUC

Fig. 8. Training CNN for IID CIFAR10 classification with DFedAvgM using: different communication bits but fix local epoch to one (first row) and different local epochs but fix the communication bits to 16 (second row). Different quantized DFedAvgM performs almost similar and more local epochs can accelerate training at the beginning but does not perform very well as training continues.

6.3 CIFAR10 Classification

Finally, we use DFedAvgM to train ResNet20 for CIFAR10 classification, which consists of 10 classes of 32×32 images with three channels. There are 50,000 training and 10,000 testing examples, which we partitioned into 20 clients uniformly, and we only consider the IID setting following [1]. We use the same data augmentation and DNN as that used in [1]. The local batch size is set to 50, the learning rate is set to 0.01, and the momentum is set to 0.9. Figure 8 shows the communication round vs. test accuracy and AUC under MIA for different quantization and different local epochs. If the local epoch is set to 1, different communication bits does not lead to a significant difference in training loss, test accuracy, and AUC under MIA. However, for the fixed communication bits 16, increase the local epochs from 1 to 2 or to 5 will make training not even converge.

7 Concluding Remarks

In this paper, we proposed a DFedAvgM and its quantized version. There two major benefits of the DFedAvgM over the existing FedAvg: 1) In FedAvg, communication between the central parameter server and local clients is required in each communication round, and this communication will be very expensive as the number of clients is very large. On the contrary, in DFedAvgM communications are between clients which are significantly less than FedAvg. 2) In FedAvg, the central server collects the updated models from clients, and attack the central server can break the privacy of the whole system. In contrast, conceptually, it is harder to break the privacy in DFedAvgM than FedAvg. Furthermore, we established the theoretical convergence for DFedAvgM and its quantized version under general nonconvex assumptions, and we showed that the worst-case convergence rate of (quantized) DFedAvgM is the same as that of DSGD. In particular, we proved a sublinear convergence rate of (quantized) DFedAvgM when the objective functions satisfy the PŁ condition. We perform extensive numerical experiments to verify the efficacy of DFedAvgM and its quantized version in training ML models and protect membership privacy.

References

[1] H. B. Mcmahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," pp. 1273–1282, 2017.

[2] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[3] M. Zinkevich, M. Weimer, L. Li, and A. J. Smola, "Parallelized stochastic gradient descent," in *Advances in neural information processing systems*, pp. 2595–2603, 2010.

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

[5] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar, "Fully decentralized federated learning," in *Third workshop on Bayesian Deep Learning (NeurIPS)*, 2018.

[6] A. Lalitha, O. C. Kilinc, T. Javidi, and F. Koushanfar, "Peer-to-peer federated learning on graphs," *arXiv preprint arXiv:1901.11173*, 2019.

[7] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*, pp. 1139–1147, 2013.

[8] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.

[9] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," in *International Conference on Learning Representations*, 2021.

[10] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," in *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2018.

[11] J. Sun, T. Chen, G. Giannakis, and Z. Yang, "Communication-efficient distributed learning via lazily aggregated quantized gradients," in *Advances in Neural Information Processing Systems*, pp. 3365–3375, 2019.

[12] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smithy, "Feddane: A federated newton-type method," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 1227–1231, IEEE, 2019.

[13] A. Khaled, K. Mishchenko, and P. Richtárik, "First analysis of local gd on heterogeneous data," *arXiv preprint arXiv:1909.04715*, 2019.

[14] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-IID data," in *International Conference on Learning Representations*, 2020.

[15] H. B. McMahan *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1, 2021.

[16] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions.," *arXiv: Learning*, 2019.

[17] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3, pp. 1653–1664, IEEE, 2005.

[18] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.
[19] L. Schenato and G. Gamba, "A distributed consensus protocol for clock synchronization in wireless sensor network," in *2007 46th ieee conference on decision and control*, pp. 2289–2294, IEEE, 2007.
[20] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Transactions on Signal processing*, vol. 57, no. 7, pp. 2748–2761, 2009.
[21] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.
[22] A. I. Chen and A. Ozdaglar, "A fast distributed proximal-gradient method," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pp. 601–608, IEEE, 2012.
[23] D. Jakovetić, J. Xavier, and J. M. Moura, "Fast distributed gradient methods," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1131–1146, 2014.
[24] I. Matei and J. S. Baras, "Performance evaluation of the consensus-based distributed subgradient method under random communication topologies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 754–771, 2011.
[25] K. Yuan, Q. Ling, and W. Yin, "On the convergence of decentralized gradient descent," *SIAM Journal on Optimization*, vol. 26, no. 3, pp. 1835–1854, 2016.
[26] J. Zeng and W. Yin, "On nonconvex decentralized gradient descent," *IEEE Transactions on Signal Processing*, vol. 66, no. 11, pp. 2834–2848, 2018.
[27] B. Sirb and X. Ye, "Consensus optimization with delayed and stochastic gradients on decentralized networks," in *Big Data (Big Data), 2016 IEEE International Conference on*, pp. 76–85, IEEE, 2016.
[28] G. Lan, S. Lee, and Y. Zhou, "Communication-efficient algorithms for decentralized and stochastic optimization," *Mathematical Programming*, vol. 180, no. 1, pp. 237–284, 2020.
[29] X. Lian, W. Zhang, C. Zhang, and J. Liu, "Asynchronous decentralized parallel stochastic gradient descent," in *Proceedings of the 35th International Conference on Machine Learning*, pp. 3043–3052, 2018.
[30] T. Sun, P. Yin, D. Li, C. Huang, L. Guan, and H. Jiang, "Non-ergodic convergence analysis of heavy-ball algorithms," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5033–5040, 2019.
[31] R. Xin and U. A. Khan, "Distributed heavy-ball: A generalization and acceleration of first-order methods with gradient tracking," *IEEE Transactions on Automatic Control*, 2019.
[32] A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani, "Quantized decentralized consensus optimization," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 5838–5843, IEEE, 2018.
[33] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, *Federated learning*. Morgan & Claypool Publishers, 2019.
[34] H. Xing, O. Simeone, and S. Bi, "Decentralized federated learning via SGD over wireless D2D networks," in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5, IEEE, 2020.
[35] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of the 1 st Adaptive & Multitask Learning Workshop, Long Beach, California*, 2019.
[36] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.
[37] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing markov chain on a graph," *SIAM review*, vol. 46, no. 4, pp. 667–689, 2004.
[38] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *Ussr Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.
[39] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Advances in Neural Information Processing Systems*, pp. 19–27, 2014.
[40] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Advances in Neural Information Processing Systems*, pp. 1709–1720, 2017.
[41] S. Magnússon, H. Shokri-Ghadikolaei, and N. Li, "On maintaining linear convergence of distributed learning and optimization under limited communication," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6101–6116, 2020.
[42] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 795–811, Springer, 2016.
[43] S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. Smola, "Stochastic variance reduction for nonconvex optimization," in *International conference on machine learning*, pp. 314–323, 2016.
[44] D. J. Foster, A. Sekhari, and K. Sridharan, "Uniform convergence of gradients for non-convex learning and optimization," in *Advances in Neural Information Processing Systems*, pp. 8745–8756, 2018.
[45] B. T. Polyak, "Gradient methods for minimizing functionals," *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, vol. 3, no. 4, pp. 643–653, 1963.
[46] S. Lojasiewicz, "A topological property of real analytic subsets," *Coll. du CNRS, Les équations aux dérivées partielles*, vol. 117, pp. 87–89, 1963.
[47] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical programming*, vol. 127, no. 1, pp. 3–30, 2011.
[48] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM Journal on optimization*, vol. 19, no. 4, pp. 1574–1609, 2009.
[49] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," *In Annual Network and Distributed System Security Symposium (NDSS 2019)*, 2019.