Achieving Optimal Sample and Communication Complexities for Non-IID Federated Learning

Prashant Khanduri [1] **Pranay Sharma** [2] **Haibo Yang** [1] **Mingyi Hong** [3] **Jia Liu** [1] **Ketan Rajawat** [4]
Pramod K. Varshney [2]

Abstract

In Federated Learning (FL) multiple worker nodes (WNs) jointly aim to build a combined model by using only local data. The goal of FL is to design algorithms such that the WNs require minimum number of samples and communication rounds to achieve the desired solution. This work addresses the above concern and considers a class of stochastic algorithms where the WNs perform a few local updates before communication. We show that when both the WN's and the server's directions are chosen based on a stochastic momentum estimator, the algorithm requires $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ samples and $\tilde{\mathcal{O}}(\epsilon^{-1})$ communication rounds to compute an ϵ-stationary solution. To the best of our knowledge, this is the first FL algorithm that achieves such *near-optimal* sample and communication complexities simultaneously. Further, we show a trade-off curve between the number of local updates and minibatch sizes, on which the above sample and communication complexities are maintained. Our insights on this trade-off provides guidelines for choosing the four important design elements for FL algorithms, WN and server's update directions, number of local updates, and the minibatch sizes to achieve the best performance.

1. Introduction

Federated Learning (FL) is a distributed optimization framework where multiple worker nodes (WNs) collaborate with the goal of learning a joint model, by only using local data (Konečnỳ et al., 2016). A classical distributed optimization problem that K WNs aim to solve:

$$\min_{x\in\mathbb{R}^d} \Big\{ f(x) \coloneqq \frac{1}{K}\sum_{k=1}^{K} f^{(k)}(x) \Big\}. \tag{1}$$

with $f^{(k)}(x) \coloneqq \mathbb{E}_{\xi^{(k)}\sim\mathcal{D}^{(k)}}[f^{(k)}(x;\xi^{(k)})]$, and where $f^{(k)} : \mathbb{R}^d \to \mathbb{R}$ denotes the smooth (possibly non-convex) objective function and $\xi^{(k)} \sim \mathcal{D}^{(k)}$ represents the sample/s drawn from distribution $\mathcal{D}^{(k)}$ at the k^{th} WN with $k \in [K]$. When the distributions $\mathcal{D}^{(k)}$ are different across the WNs, it is referred to as the *heterogeneous* data setting.

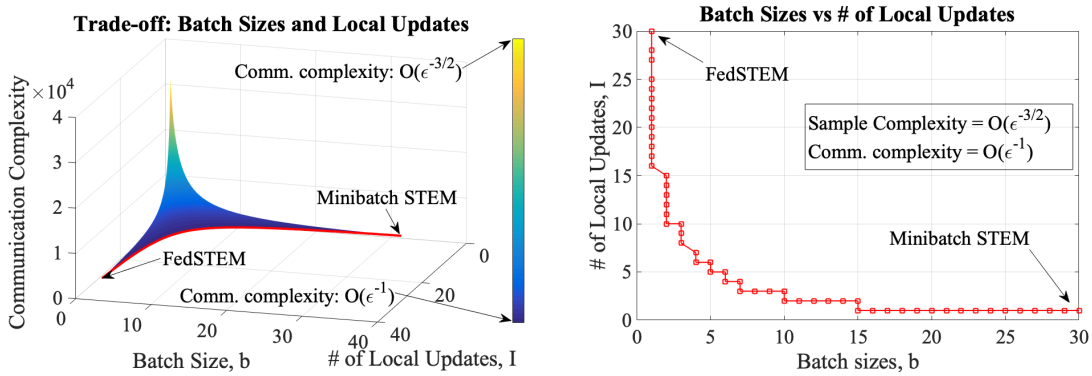
The optimization performance of non-convex FL algorithms is typically measured by the total number of samples accessed (cf. Definition 2.2) and the total rounds of communication (cf. Definition 2.3) required by each WN to achieve an ϵ-stationary solution (cf. Definition 2.1). To minimize the sample and the communication complexities, FL algorithms rely on the following *four* key design elements: (i) the WNs' local model update directions, (ii) Minibatch size to compute each local direction, (iii) the number of local updates before WNs share their parameters, and (iv) the SN's update direction. How to find effective FL algorithms by (optimally) designing these parameters has received significant research interest recently.

Contributions. In this work, we propose Stochastic Two-Sided Momentum (STEM) algorithm, that utilizes momentum-assisted stochastic gradient directions for *both* the WNs and server node's (SN) updates. We show that there exists an *optimal* trade off between the minibatch sizes and the number of local updates, such that on the trade-off curve STEM requires $\tilde{\mathcal{O}}(\epsilon^{-3/2})$[1] samples and $\tilde{\mathcal{O}}(\epsilon^{-1})$ communication rounds to reach an ϵ-stationary solution; see Figure 1 for an illustration. These complexity results are the best achievable for first-order stochastic FL algorithms (under certain assumptions, cf. Assumption 1); see (Fang et al., 2018) and (Zhang et al., 2020), as well as Remark 1 of this paper for discussions regarding optimality. To the best of our knowledge, STEM is the first algorithm which

[1]Dept. of ECE, The Ohio State University, OH, USA, [2]EECS Dept., Syracuse University, NY, USA, [3]Dept. of ECE, University of Minnesota, MN, USA, [4]Dept. of EE, Indian Institute of Technology Kanpur, India. Correspondence to: Prashant Khanduri <khand095@umn.edu>.

This work was presented at the International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021 (FL-ICML'21). This workshop does not have official proceedings and this paper is non-archival.

[1]The notation $\tilde{\mathcal{O}}(\cdot)$ hides the logarithmic factors.



(a) Communication complexity. (b) Minibatch sizes vs Local Updates.

Figure 1. The 3D surface in (a) plots the communication complexity of the proposed STEM for different minibatch sizes and number of local updates. The surface is generated such that each point represents STEM with a particular choice of (b, I), so that it requires $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ samples to achieve ϵ-stationarity. Plot (b) shows the optimal trade off between the minibatch sizes and the number of local updates at each WN (i.e., achieving the lowest communication and sample complexities). Both plots are generated for an accuracy of $\epsilon = 10^{-3}$ and all the constants dependent on system parameters (variance of stochastic gradients, heterogeneity parameter, optimality gap, Lipschitz constants, etc.) are assumed to be 1. Fed STEM is a special case of STEM where $\mathcal{O}(1)$ minibatch is used; Minibatch STEM is a special case of STEM where $\mathcal{O}(1)$ local updates are used.

– (i) *simultaneously* achieves the optimal sample and communication complexities and (ii) can optimally trade off the minibatch sizes and the number of local updates. Collectively, our insights on the trade-offs provide practical guidelines for choosing different design elements for FL algorithms.

Related Works. FL algorithms were first proposed in the form of FedAvg (McMahan et al., 2017), where the local update directions at each WN were chosen to be the SGD updates. Recent studies have focused on designing new algorithms to deal with heterogeneous data settings, as well as problems where the local loss functions are non-convex (Zhang et al., 2020; Li et al., 2018; Yu et al., 2018; 2019; Karimireddy et al., 2020b; Das et al., 2020; Liang et al., 2019; Reddi et al., 2019). In (Yu et al., 2018), the authors showed that Parallel Restarted SGD (Local SGD or FedAvg) achieves linear speed up while requiring $\mathcal{O}(\epsilon^{-2})$ samples and $\mathcal{O}(\epsilon^{-3/2})$ rounds of communication to reach an ϵ-stationary solution. In (Yu et al., 2019), a Momentum SGD was proposed, which achieved the same sample and communication complexities as Parallel Restarted SGD (Yu et al., 2018). The works in (Karimireddy et al., 2020b; Yang et al., 2021) conducted tighter analysis for FedAvg with partial WN participation with $\mathcal{O}(1)$ local updates and batch sizes. Their analysis showed that FedAvg's sample and communication complexities are both $\mathcal{O}(\epsilon^{-2})$. Additionally, SCAFFOLD was proposed in (Karimireddy et al., 2020b), which utilized variance reduction based local update directions (Johnson & Zhang, 2013) to achieve the same sample and communication complexities as FedAvg. Similarly, VRL-SGD proposed in (Liang et al., 2019) also utilized variance reduction and showed improved communication complexity of $\mathcal{O}(\epsilon^{-1})$, while requiring the same computations as FedAvg. Importantly, both SCAFFOLD and VRL-SGD's guarantees were independent of the data heterogeneity. FedProx and FedPD proposed in (Li et al., 2018) and (Zhang et al., 2020), resp., improved the communication complexity of FedAvg to $\mathcal{O}(\epsilon^{-1})$. Recently, (Karimireddy et al., 2020a; Das et al., 2020) proposed to utilize hybrid momentum gradient estimators (Cutkosky & Orabona, 2019; Tran-Dinh et al., 2019). Both MIME (Karimireddy et al., 2020a) and FedGLOMO (Das et al., 2020) matched the optimal sample complexity (under certain smoothness assumptions) of $\mathcal{O}(\epsilon^{-3/2})$ of the centralized non-convex stochastic optimization algorithms (Fang et al., 2018; Zhou et al., 2018; Cutkosky & Orabona, 2019; Tran-Dinh et al., 2019), while requiring $\mathcal{O}(\epsilon^{-3/2})$ communication rounds to achieve an ϵ-stationary solution. Please see Table 1 for a summary of the above discussion.

The comparison of Local SGD (FedAvg) to Minibatch SGD for convex and strongly convex problems with heterogeneous data setting was conducted in (Woodworth et al., 2020). It was shown that Minibatch SGD almost always dominates the Local SGD. In contrast, it was shown in (Lin et al., 2018) that Local SGD dominates Minibatch SGD in terms of generalization performance. Although existing FL results are rich, but they are somehow ad hoc and there is a lack of principled understanding of the algorithms. We note that the proposed STEM algorithmic framework provides a theoretical framework that unifies federated and minibatch algorithms while achieving near optimal sample and communication complexities.

Algorithm	Work	Sample	Comm.	Minibatch (b)	Updates (I)
FedAvg$^\diamond$	(Yu et al., 2018) /(Yu et al., 2019)		$\mathcal{O}(\epsilon^{-3/2})$	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-1/2})$
	(Karimireddy et al., 2020b)/(Yang et al., 2021)	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
	this work		$\mathcal{O}(\epsilon^{-3/2})$	$\mathcal{O}\big(\epsilon^{-\frac{2(1-\nu)}{(4-\nu)}}\big)$	$\mathcal{O}\big(\epsilon^{-\frac{3\nu}{2(4-\nu)}}\big)$
SCAFFOLD*	(Karimireddy et al., 2020b)	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
FedPD/FedProx‡	(Zhang et al., 2020)/(Li et al., 2018)	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\epsilon^{-1})$	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-1})$
MIME†/FedGLOMO	(Karimireddy et al., 2020a)/(Das et al., 2020)	$\mathcal{O}(\epsilon^{-3/2})$	$\mathcal{O}(\epsilon^{-3/2})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
STEM$^\diamond$	this work	$\tilde{\mathcal{O}}(\epsilon^{-3/2})$	$\tilde{\mathcal{O}}(\epsilon^{-1})$	$\mathcal{O}\big(\epsilon^{-\frac{3(1-\nu)}{2(3-\nu)}}\big)$	$\mathcal{O}\big(\epsilon^{-\frac{\nu}{(3-\nu)}}\big)$
Fed STEM				$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-1/2})$
Minibatch STEM*				$\mathcal{O}(\epsilon^{-1/2})$	$\mathcal{O}(1)$

Table 1. Comparison of FedAvg and STEM with different FL algorithms for various choices of the minibatch sizes (b) and the number of per node local updates between two rounds of communication (I).
$^\diamond$ $\nu \in [0,1]$ trades off b and I; $\nu = 1$ (resp. $\nu = 0$) uses multiple (resp. $\mathcal{O}(1)$) local updates and $\mathcal{O}(1)$ (resp. multiple) samples. Fed STEM and Minibatch STEM are two variants of the proposed STEM.
‡The data heterogeneity assumption is weaker than Assumption 2 (please see (Zhang et al., 2020) for details).
†Requires bounded Hessian dissimilarity to model data heterogeneity across WNs.
*Guarantees for Minibatch STEM with $I = 1$ and SCAFFOLD are independent of the data heterogeneity.

Notations. The expected value of a random variable X is denoted by $\mathbb{E}[X]$ and its expectation conditioned on an Event A is denoted as $\mathbb{E}[X|\text{Event } A]$. We denote by $\mathbb{R}$ (and $\mathbb{R}^d$) the real line (and the d-dimensional Euclidean space). The set of natural numbers is denoted by $\mathbb{N}$. Given a positive integer $K \in \mathbb{N}$, we denote $[K] \triangleq \{1, 2, \ldots, K\}$. Notation $\|\cdot\|$ denotes the ℓ_2-norm and $\langle \cdot, \cdot \rangle$ the Euclidean inner product. For a discrete set $\mathcal{B}$, $|\mathcal{B}|$ denotes the cardinality of the set. We denote by $\bar{x} = \frac{1}{K}\sum_{k=1}^K x^{(k)}$ the empirical mean of a set of vectors.

2. Preliminaries

Before we proceed to the the algorithms, we make the following assumptions about problem (1).

Assumption 1 (Sample Gradient Lipschitz Smoothness)**.** The stochastic functions $f^{(k)}(\cdot, \xi^{(k)})$ with $\xi^{(k)} \sim \mathcal{D}^{(k)}$ for all $k \in [K]$, satisfy the mean squared smoothness property, i.e, $\forall x, y \in \mathbb{R}^d$ we have

$$\mathbb{E}\|\nabla f^{(k)}(x; \xi^{(k)}) - \nabla f^{(k)}(y; \xi^{(k)})\|^2 \leq L^2 \|x - y\|^2.$$

Assumption 2 (Unbiased gradient and Variance Bounds)**.** (i) Unbiased Gradient. The stochastic gradients computed at each WN are unbiased

$$\mathbb{E}[\nabla f^{(k)}(x; \xi^{(k)})] = \nabla f^{(k)}(x), \ \forall\, \xi^{(k)} \sim \mathcal{D}^{(k)}, \ \forall\, k \in [K].$$

(ii) Intra- and inter- node Variance Bound. The following bounds hold:

$$\mathbb{E}\|\nabla f^{(k)}(x; \xi^{(k)}) - \nabla f^{(k)}(x)\|^2 \leq \sigma^2, \ \forall \xi^{(k)} \sim \mathcal{D}^{(k)},$$
$$\|\nabla f^{(k)}(x) - \nabla f^{(\ell)}(x)\|^2 \leq \zeta^2, \ \ \forall k, \ell \in [K].$$

Note that Assumption 1 is stronger than directly assuming $f^{(k)}$'s are Lipschitz smooth (which we will refer to as the *averaged* gradient Lipschitz smooth condition), but it is still a rather standard assumption in SGD analysis. For example it has been used in analyzing centralized SGD algorithms such as SPIDER (Fang et al., 2018), SNVRG (Zhou et al., 2018), STORM (Cutkosky & Orabona, 2019) (and many others) as well as in FL algorithms such as MIME (Karimireddy et al., 2020a) and Fed-GLOMO (Das et al., 2020). The second relation in Assumption 2-(ii) quantifies the data heterogeneity, and we call $\zeta > 0$ as the *heterogeneity parameter*. This is a typical assumption required to evaluate the performance of FL algorithms. If data distributions across individual WNs are identical, i.e., $\mathcal{D}^{(k)} = \mathcal{D}^{(\ell)}$ for all $k, \ell \in [K]$ then we have $\zeta = 0$. Next, we define the ϵ-stationary solution for non-convex optimization problems, as well as quantify the sample and communication complexities.

Definition 2.1 (ϵ-Stationary Point)**.** A point x is called ϵ-stationary if $\|\nabla f(x)\|^2 \leq \epsilon$. Moreover, a stochastic algorithm is said to achieve an ϵ-stationary point in t iterations if $\mathbb{E}[\|\nabla f(x_t)\|^2] \leq \epsilon$, where the expectation is over the stochasticity of the algorithm until time instant t.

Definition 2.2 (Sample complexity)**.** We assume an Incremental First-order Oracle (IFO) framework (Bottou et al., 2018), where, given a sample $\xi^{(k)} \sim \mathcal{D}^{(k)}$ at the k^{th} node and iterate x, the oracle returns $(f^{(k)}(x; \xi^{(k)}), \nabla f^{(k)}(x; \xi^{(k)}))$. Each access to the oracle is counted as a single IFO operation. We measure the sample (and computational) complexity in terms of the total number of calls to the IFO by all WNs to achieve an ϵ-stationary point given in Definition 2.1.

Definition 2.3 (Communication complexity)**.** We define a communication round as a one back-and-forth sharing of pa-

rameters between the WNs and the SN. The communication complexity is defined to be the total number of communication rounds between any WN and the SN required to achieve an ϵ-stationary point given in Definition 2.1.

3. The STEM algorithm and the trade-off analysis

In this section, we discuss the proposed algorithm and present the main results. The key in the algorithm design is to carefully balance *all the four* design elements, WNs and SNs update directions, local updates and the minibatch sizes, mentioned in Sec. 1, so that sufficient and useful progress can be made between two rounds of communication.

Let us discuss the key steps of STEM, listed in Algorithm 1. In Step 10, each node locally updates its model parameters using the local direction d_t^k, computed by using b stochastic gradients at two consecutive iterates $x_{t+1}^{(k)}$ and $x_t^{(k)}$ as

$$d_{t+1}^{(k)} = \frac{1}{b}\sum_{\xi_{t+1}^{(k)}\in\mathcal{B}_{t+1}^{(k)}} \nabla f^{(k)}(x_{t+1}^{(k)};\xi_{t+1}^{(k)}) + (1-a_{t+1})\Big(d_t^{(k)} - \frac{1}{b}\sum_{\xi_{t+1}^{(k)}\in\mathcal{B}_{t+1}^{(k)}} \nabla f^{(k)}(x_t^{(k)};\xi_{t+1}^{(k)})\Big) \quad (2)$$

with $|\mathcal{B}_{t+1}^{(k)}| = b$, and $a_{t+1} = c\cdot\eta_t^2$. After every I local steps, the WNs share their current local models $\{x_{t+1}^{(k)}\}_{k=1}^K$ and directions $\{d_{t+1}^{(k)}\}_{k=1}^K$ with the SN. The SN aggregates these quantities, and performs a server-side momentum step, before returning $x_{t+2}^{(k)}$ and $\bar{d}_{t+1}$ to all the WNs. Because both the WNs and the SN perform momentum based updates, we call the algorithm a stochastic *two-sided* momentum algorithm. The key parameters are: b the minibatch size, I the local update steps between two communication rounds, $\{\eta_t\}$ the stepsizes, and $\{a_t\}$ the momentum parameters.

One key technical innovation of our algorithm design is to identify the most suitable way to incorporate momentum based directions in FL algorithms. Although the momentum-based gradient estimator itself is not new and has been used in the literature before (see e.g., in (Cutkosky & Orabona, 2019; Tran-Dinh et al., 2019) and (Karimireddy et al., 2020a; Das et al., 2020) to improve the sample complexities of centralized and decentralized stochastic optimization problems, respectively), it is by no means clear if and how it can contribute to improve the communication complexity of FL algorithms. We show that in the FL setting, the local directions together with the local models have to be aggregated by the SN so to avoid being influenced too much by the local data. More importantly, besides the WNs, the SN also needs to perform updates using the (aggregated) momentum directions. Finally, such *two-sided* momentum updates have to be done carefully with the correct choice of minibatch size

Algorithm 1 STEM Algorithm

1: **Input**: Parameters: $c > 0$, the number of local updates I, batch size b, stepsizes $\{\eta_t\}$.
2: **Initialize**: Iterate $x_1^{(k)} = \bar{x}_1 = \frac{1}{K}\sum_{k=1}^K x_1^{(k)}$, descent direction $d_1^{(k)} = \bar{d}_1 = \frac{1}{K}\sum_{k=1}^K d_1^{(k)}$ with $d_1^{(k)} = \frac{1}{B}\sum_{\xi_1^{(k)}\in\mathcal{B}_1^{(k)}} \nabla f^{(k)}(x_1^{(k)};\xi_1^{(k)})$ and $|\mathcal{B}_1^{(k)}| = B$ for $k\in[K]$.
Perform: $x_2^{(k)} = x_1^k - \eta_1 d_1^{(k)},\ \forall\, k\in[K]$
3: **for** $t = 1$ to T **do**
4: **for** $k = 1$ to K **do**
5: Compute $d_{t+1}^{(k)}$ using (2)
6: **if** t mod $I = 0$ **then**
7: $d_{t+1}^{(k)} = \bar{d}_{t+1}$
8: $x_{t+2}^{(k)} := \bar{x}_{t+1} - \eta_{t+1}\bar{d}_{t+1}$ #SN momentum
9: **else**
10: $x_{t+2}^{(k)} = x_{t+1}^{(k)} - \eta_{t+1} d_{t+1}^{(k)}$ #WN momentum
11: **end if**
12: **end for**
13: **end for**
14: $\bar{x}_a$ chosen uniformly randomly from $\{\bar{x}_t\}_{t=1}^T$

b, and the number of local updates I. Overall, it is the judicious choice of all these design elements that results in the optimal sample and communication complexities. Next, we present the convergence guarantees of the STEM algorithm.

Theorem 3.1. *Under the Assumptions 1 and 2, suppose the stepsize sequence is chosen as:* $\eta_t = \frac{\bar{\kappa}}{(w_t+\sigma^2 t)^{1/3}}$, *where we define :* $\bar{\kappa} = \frac{(bK)^{2/3}\sigma^{2/3}}{L}$ *and* $w_t = \max\Big\{2\sigma^2, 4096L^3I^3\bar{\kappa}^3 - \sigma^2 t, \frac{c^3\bar{\kappa}^3}{4096L^3I^3}\Big\}$. *Further, let us set* $c = \frac{64L^2}{bK} + \frac{\sigma^2}{24\bar{\kappa}^3 LI} = L^2\Big(\frac{64}{bK} + \frac{1}{24(bK)^2 I}\Big)$, *and set the initial batch size as* $B = bI$; *set the local updates* I *and minibatch size* b *as follows:*

$$I = \mathcal{O}\big((T/K^2)^{\nu/3}\big), \quad b = \mathcal{O}\big((T/K^2)^{1/2-\nu/2}\big) \quad (3)$$

where ν *satisfies* $\nu\in[0,1]$. *For STEM the following holds:*

(i) For $\bar{x}_a$ *chosen according to Algorithm 1, we have:*

$$\mathbb{E}\|\nabla f(\bar{x}_a)\|^2 = \mathcal{O}\Big(\frac{f(\bar{x}_1)-f^*}{K^{2\nu/3}T^{1-\nu/3}}\Big) + \tilde{\mathcal{O}}\Big(\frac{\sigma^2}{K^{2\nu/3}T^{1-\nu/3}}\Big) + \tilde{\mathcal{O}}\Big(\frac{\zeta^2}{K^{2\nu/3}T^{1-\nu/3}}\Big). \quad (4)$$

(ii) For any $\nu\in[0,1]$, *we have*
Sample Complexity: *The sample complexity of STEM is* $\tilde{\mathcal{O}}(\epsilon^{-3/2})$. *This implies that each WN requires at most* $\tilde{\mathcal{O}}(K^{-1}\epsilon^{-3/2})$ *gradient computations, thereby achieving linear speedup with the number of*

WNs present in the network.

Communication Complexity: *The communication complexity of* STEM *is* $\tilde{\mathcal{O}}(\epsilon^{-1})$.

A few remarks are in order.

Remark 1 (Near-Optimal sample and communication complexities). Theorem 3.1 suggests that when I and b are selected appropriately, then STEM achieves $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ and $\tilde{\mathcal{O}}(\epsilon^{-1})$ sample and communication complexities. Taking them separately, these complexity bounds are the best achievable by the existing FL algorithms (upto logarithmic factors regardless of sample or batch Lipschitz smooth assumption) (Drori & Shamir, 2020); see Table 1. We note that the $\mathcal{O}(\epsilon^{-3/2})$ complexity is the best possible that can be achieved by centralized SGD with the sample Lipschitz gradient assumption; see (Fang et al., 2018). On the other hand, the $\mathcal{O}(\epsilon^{-1})$ complexity bound is also *likely* to be the optimal, since in (Zhang et al., 2020) the authors showed that even when the local steps use a class of (deterministic) first-order algorithms, $\mathcal{O}(\epsilon^{-1})$ is the best achievable communication complexity. The only difference is that (Zhang et al., 2020) does not explicitly assume the intra-node variance bound (i.e., the second relation in Assumption 2-(ii)). We leave the precise characterization of the communication lower bound with intra-node variance as future work. □

Remark 2 (The Optimal Batch Sizes and Local Updates Trade-off). The parameter $\nu \in [0, 1]$ balances the local minibatch sizes b, and the number of local updates I. Eqs. in (3) suggest that when ν increases from 0 to 1, b decreases and I increases. Specifically, if $\nu = 1$, then b is a $\mathcal{O}(1)$ but $I = \mathcal{O}(T^{1/3}/K^{2/3})$. In this case, each WN chooses a small minibatch while executing multiple local updates, and STEM resembles FedAvg (a.k.a. Local SGD) but with double-sided momentum update directions, and is referred to as FedSTEM. In contrast, if $\nu = 0$, then $b = \mathcal{O}(T^{1/2}/K)$ but I is $\mathcal{O}(1)$. In this case, each WN chooses a large batch size while executing only a few, or even one, local updates, and STEM resembles the Minibatch SGD, but again with different update directions, and is referred to as Minibatch STEM. Such a trade-off can be seen in Fig. 1(b). □

Remark 3 (Sub-Optimal batch sizes & local updates trade-off). STEM requires $\tilde{\mathcal{O}}\big(\max\{(b \cdot I)\epsilon^{-1}, K^{-1}\epsilon^{-3/2}\}\big)$ samples and $\tilde{\mathcal{O}}\big(\max\{\epsilon^{-1}, (b \cdot I)^{-1}K^{-1}\epsilon^{-3/2}\}\big)$ communication rounds. Above expressions imply if $b \cdot I$ increases beyond $\mathcal{O}(K^{-1}\epsilon^{-1/2})$, then the sample complexity will increase from the optimal $\tilde{\mathcal{O}}(\epsilon^{-3/2})$; otherwise, the optimal sample complexity $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ is maintained. On the other hand, if $b \cdot I$ decreases beyond $\mathcal{O}(K^{-1}\epsilon^{-1/2})$, the communication complexity increases from $\tilde{\mathcal{O}}(\epsilon^{-1})$. For instance, if we choose $b = \mathcal{O}(1)$ and $I = \mathcal{O}(1)$ the communication complexity becomes $\tilde{\mathcal{O}}(\epsilon^{-3/2})$. This trade-off is illustrated in Figure 1(a), where we maintain the optimal sample complexity, while changing b and I to generate the trade-off surface. □

Remark 4 (Data Heterogeneity). The term $\tilde{\mathcal{O}}\Big(\frac{\zeta^2}{K^{2\nu/3}T^{1-\nu/3}}\Big)$ in the gradient bound (4) captures the effect of the heterogeneity of data across WNs, where ζ is the parameter characterizing the intra-node variance and has been defined in Assumption 2-(ii). Highly heterogeneous data with large ζ^2 can adversely impact the performance of STEM. Note that such a dependency on ζ also appears in other existing FL algorithms, such as (Zhang et al., 2020; Yu et al., 2019; Das et al., 2020). However, there is one special case of STEM that does not depend on the parameter ζ. This is the case where $I = 1$, i.e., the minibatch SGD counterpart of STEM where only a single local iteration is performed between two communication rounds. We have the following corollary. □

Corollary 1 (Minibatch STEM)**.** *Under Assumptions 1 and 2 , and choose the algorithm parameters as in Theorem 3.1. At each WN, choose* $I = 1$, $b = (T/K^2)^{1/2}$, *and the initial batch size* $B = b \cdot I$. *Then* STEM *satisfies:*

(i) For $\bar{x}_a$ *chosen according to Algorithm 1, we have*

$$\mathbb{E}\|\nabla f(\bar{x}_a)\|^2 = \mathcal{O}\Big(\frac{f(\bar{x}_1) - f^*}{T}\Big) + \tilde{\mathcal{O}}\Big(\frac{\sigma^2}{T}\Big).$$

(ii) Minibatch STEM *achieves* $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ *sample and* $\tilde{\mathcal{O}}(\epsilon^{-1})$ *communication complexity.*

This section established that once the WNs' and the SNs' update directions are fixed, there exists a sequence of optimal choices of the number of local updates I, and the batch sizes b, which guarantees the best possible sample and communication complexities for STEM. The trade-off analysis presented in this section provides some useful guidelines for how to best select b and I in practice. Our subsequent numerical results verify that if b or I are not chosen judiciously, then the practical performance of the algorithms can degrade significantly.

4. Numerical results

In this section, we validate the proposed STEM algorithm and compare its performance with the de facto standard FedAvg (McMahan et al., 2017) and recently proposed SCAFFOLD (Karimireddy et al., 2020b). The goal of our experiments are three-fold: (1) To show that STEM performs better compared to other algorithms, (2) there are multiple ways to reach the desired solution accuracy, one can either choose a large batch size and perform only a few local updates or select a smaller batch size and perform multiple local updates, and finally, (3) if the local updates and the batch sizes are not chosen appropriately, the WNs might

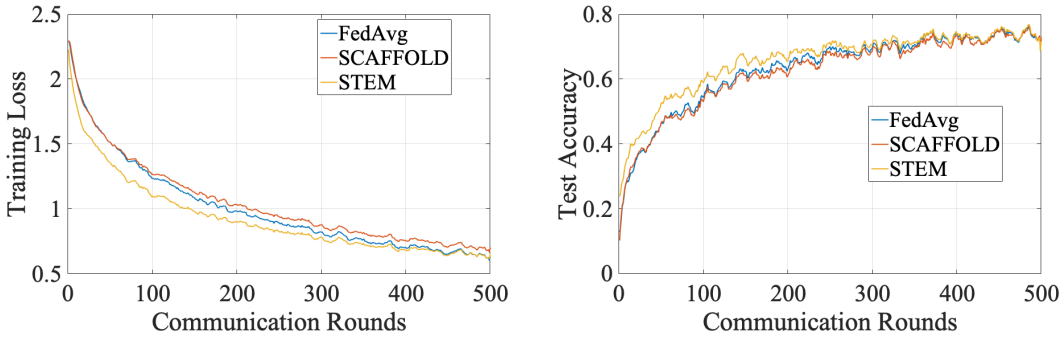


Figure 2. Training loss and testing accuracy for classification on CIFAR-10 dataset against the number of communication rounds for moderate heterogeneity setting with $b = 8$ and $I = 61$.

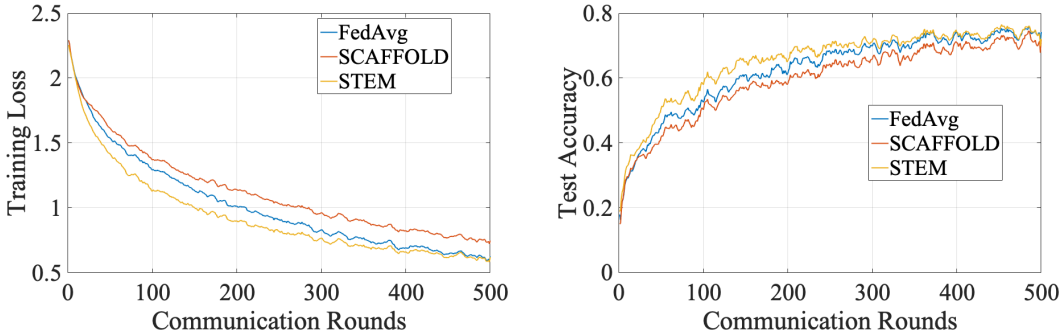


Figure 3. Training loss and testing accuracy for classification on CIFAR-10 dataset against the number of communication rounds for moderate heterogeneity setting with $b = 64$ and $I = 7$.

need to perform excessive computations, thereby slowing down convergence.

Data and Parameter Settings: We compare the algorithms for image classification tasks on CIFAR-10 and MNIST data sets with 100 WNs in the network. For both CIFAR-10 and MNIST, each WN implements a two-hidden-layer convolutional neural network (CNN) architecture followed by three linear layers for CIFAR-10 and two for MNIST. All the experiments are implemented on a single NVIDIA Quadro RTX 5000 GPU. We consider two settings, one with moderate and the other with high heterogeneity. For both settings, the data is partitioned into disjoint sets among the WNs. In the moderate heterogeneity setting, the WNs have access to partitioned data from all the classes but for the high heterogeneity setting the data is partitioned such that each WN can access data from only a subset (5 out of 10 classes) of classes. For CIFAR-10 (resp. MNIST), each WN has access to 490 (resp. 540) samples for training and 90 (resp. 80) samples for testing purposes.

For STEM, we set $w_t = 1$, $c = \bar{c}/\bar{\kappa}^2$ and tune for $\bar{\kappa}$ and $\bar{c}$ in the range $\bar{\kappa} \in [0.01, 0.5]$ and $\bar{c} \in [1, 10]$, respectively (cf. Theorem 3.1). We note that for small batch sizes $\bar{\kappa} \in [0.01, 0.1]$, whereas for larger batch sizes $\bar{\kappa} \in [0.3, 0.5]$ perform well. We diminish η_t as given in Theorem 3.1 in each epoch[2]. For SCAFFOLD and FedAvg, the stepsize choices of 0.1 and 0.01 perform well for large and smaller batch sizes, respectively. We use cross entropy as the loss function and evaluate the algorithm performance under a few settings discussed next.

Discussion: In Figures 2 and 3, we compare the training and testing performance of STEM with FedAvg and SCAFFOLD for CIFAR-10 dataset under moderate heterogeneity setting. For Figure 2, we choose $b = 8$ and $I = 61$, whereas for Figure 3, we choose $b = 64$ and $I = 7$. We first note that for both cases STEM performs better than FedAvg and SCAFFOLD. Moreover, observe that for both settings, small batches with multiple local updates (Figure 2) and large batches with few local updates (Figure 3), the algorithms converge with approximately similar performance, corroborating the theoretical analysis (see Discussion in Section 1). Next, in Figure 4 we evaluate the performance of the proposed algorithms on CIFAR-10 with high heterogeneity setting for $b = 8$ and $I = 61$. We note that STEM outperforms FedAvg and SCAFFOLD in this setting

[2]We define epoch as a single pass over the whole data.

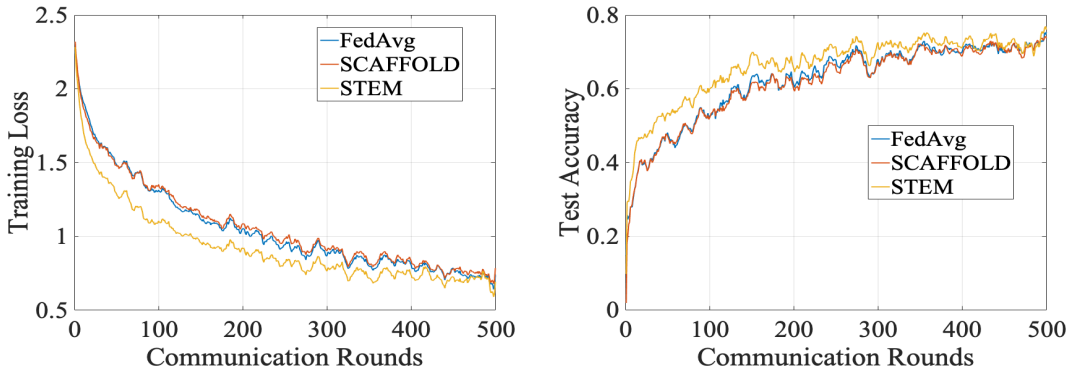


Figure 4. Training loss and testing accuracy for classification on CIFAR-10 dataset against the number of communication rounds for high heterogeneity setting with $b = 8$ and $I = 61$.

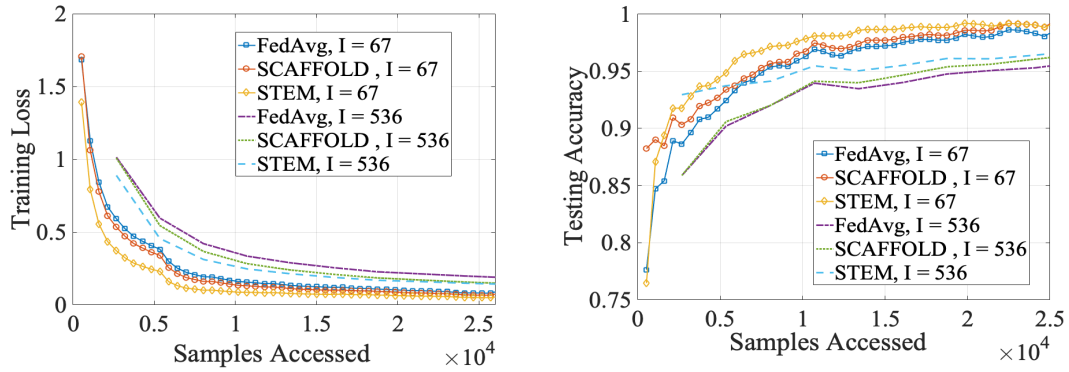


Figure 5. Training loss and the testing accuracy for classification on MNIST data set against the number of samples accessed at each WN for high heterogeneity setting with $b = 8$.

as well. Finally, with the next set of experiments we emphasize the importance of choosing b and I carefully. In Figure 5, we compare the training and testing performance of the algorithms against the number of samples accessed at each WN for the classification task on MNIST dataset with high heterogeneity. We fix $b = 8$ and conduct experiments under two settings, one with $I = 67$, and the other with $I = 536$ local updates at each WN. Note that although a large number of local updates might lead to fewer communication rounds but it can make the sample complexity extremely high as is demonstrated by Figure 5. For example, Figure 5 shows that to reach testing accuracy of $96 - 97\%$ with $I = 67$, STEM requires approximately $5000 - 6000$ samples, in contrast with $I = 536$ it requires more than 25000 samples at each WN. Similar behavior can be observed if we fix $I > 1$ and increase the local batch sizes. This implies not choosing the local updates and the batch sizes judiciously might lead to increased sample complexity.

5. Conclusion

In this work, we proposed a novel algorithm STEM, for distributed stochastic non-convex optimization with applications to FL. We showed that STEM reaches an ϵ-stationary point with $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ sample complexity while achieving linear speed-up with the number of WNs. Moreover, the algorithm achieves a communication complexity of $\tilde{\mathcal{O}}(\epsilon^{-1})$. We established a (optimal) trade-off that allows interpolation between varying choices of local updates and the batch sizes at each WN while maintaining (near optimal) sample and communication complexities. Our results provide guidelines to carefully choose the number of local updates, directions, and minibatch sizes to achieve the best performance. The future directions of this work include developing lower bounds on communication complexity that establishes the tightness of the analysis conducted in this work.

References

Bottou, L., Curtis, F. E., and Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018. doi: 10.1137/16M1080173.

Cutkosky, A. and Orabona, F. Momentum-based variance reduction in non-convex SGD. In *Advances in Neural Information Processing Systems 32*, pp. 15236–15245. Curran Associates, Inc., 2019.

Das, R., Hashemi, A., Sanghavi, S., and Dhillon, I. S. Improved convergence rates for non-convex federated learning with compression. *arXiv preprint arXiv:2012.04061*, 2020.

Drori, Y. and Shamir, O. The complexity of finding stationary points with stochastic gradient descent. In *International Conference on Machine Learning*, pp. 2658–2667. PMLR, 2020.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pp. 689–699, 2018.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pp. 315–323. Curran Associates, Inc., 2013.

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020a.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pp. 5132–5143. PMLR, 2020b.

Konečnỳ, J., McMahan, H. B., Ramage, D., and Richtárik, P. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Liang, X., Shen, S., Liu, J., Pan, Z., Chen, E., and Cheng, Y. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. Don't use large mini-batches, use local sgd, 2018.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Reddi, S. J., Kale, S., and Kumar, S. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

Tran-Dinh, Q., Pham, N. H., Phan, D. T., and Nguyen, L. M. Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. *arXiv preprint arXiv:1905.05920*, 2019.

Woodworth, B., Patel, K. K., and Srebro, N. Minibatch vs local sgd for heterogeneous distributed learning, 2020.

Yang, H., Fang, M., and Liu, J. Achieving linear speedup with partial worker participation in non-iid federated learning, 2021.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning, 2018.

Yu, H., Jin, R., and Yang, S. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization, 2019.

Zhang, X., Hong, M., Dhople, S., Yin, W., and Liu, Y. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data, 2020.

Zhou, D., Xu, P., and Gu, Q. Stochastic nested variance reduction for nonconvex optimization. *arXiv preprint arXiv:1806.07811*, 2018.