Federated Learning under Arbitrary Communication Patterns

Dmitrii Avdyukhin [1] **Shiva Prasad Kasiviswanathan** [2]

Abstract

Federated Learning is a distributed learning setting where the goal is to train a centralized model with training data distributed over a large number of heterogeneous clients, each with unreliable and relatively slow network connections. A common optimization approach used in federated learning is based on the idea of *local SGD*: each client runs some number of SGD steps locally and then the updated local models are averaged to form the updated global model on the coordinating server. In this paper, we investigate the performance of an asynchronous version of local SGD wherein the clients can communicate with the server at arbitrary time intervals. Our main result shows that for smooth strongly convex and smooth nonconvex functions we achieve convergence rates that match the synchronous version that requires all clients to communicate simultaneously.

1. Introduction

Federated learning (FL) is a distributed machine learning setting that aims to collaboratively train a model under the orchestration of a central server. Practical applications of FL range from *cross-device* scenarios, where a huge number of typically unreliable clients with small quantities of data per client participate, to *cross-silo* scenarios with smaller numbers of reliable clients, each possessing larger quantities of data (Kairouz et al., 2019). Typically, in a FL application the clients perform most of the computation, and a central parameter server updates the model parameters using the information returned by the clients. Without explicit sharing of data from clients, FL can mitigate some of the privacy risks associated with traditional distributed learning techniques.

[1]Department of Computer Science, Indiana University, Bloomington, IN, USA [2]Amazon, Palo Alto, CA, USA. Correspondence to: Dmitrii Avdyukhin <davdyukh@iu.edu>, Shiva Kasiviswanathan <kasivisw@gmail.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

The canonical federated learning problem involves learning a single, global statistical model from data stored on lots of remote devices. In particular, the goal is typically to minimize the following objective function:

$$\min_x \ f(x) \ \text{ where } \ f(x) = \frac{1}{N}\sum_{i=1}^{N} f^{(i)}(x), \qquad (1)$$

where each $f^{(i)}$ is based on data available on client i. Here, N is the number of clients. Federated learning brings in some unique characteristics in solving the optimization problem posed in (1), which also makes the federated learning distinct from traditional distributed learning.

(i) Communication is a critical bottleneck in federated settings. Federated networks are potentially comprised of a massive number of devices, and communication in the network can be slower than local computation by many orders of magnitude (Kairouz et al., 2019).

(ii) Devices frequently generate and collect data in a varied manner, e.g., mobile phone users may use language differently which might affect the next word prediction task. This means that the training data are non-identically distributed, that is, a device's local data cannot be regarded as samples drawn from the overall distribution.

(iii) One final difference is that unlike traditional distributed learning systems, in the FL setting the server has no control over users' devices. For example, when a WiFi access on a device is temporarily unavailable, the device may not communicate with the server for many rounds.

1.1. Our Model and Results

We propose a federated learning algorithm that tries to addresses all the three challenges laid above. In particular, we get away from commonly used two impractical assumptions: (a) *identical data distribution* across clients and (b) all clients can synchronize and communicate *periodically or as demanded* by the server.

With the goal of increasing the compute to communication ratio, a common idea in federated/distributed setup is that instead of keeping the iterates on different clients in sync, we allow them to evolve locally on each machine, independent from each other, and only average the sequences

Table 1: Comparison of our bounds with existing results (under similar assumptions) to reach asymptotically the same statistical term as the convergence rate of minibatch SGD. Larger Δ is preferable for reducing communication. Basu et al. (2019) results also consider additional gradient compression techniques, which are ignored here for a direct comparison.

Assumption on f	Bound on Δ with sync. local SGD	Previous Bound on Δ with async. client updates	Our Bound on Δ with async. client updates
Smooth, Strongly Convex	$O(\sqrt{T/N})$ (Basu et al., 2019)	$O((T/N)^{1/4})$ (Basu et al., 2019)	$O(\sqrt{T/N})$
Smooth, Nonconvex	$O(T^{1/4}/N^{3/4})$ (Yu et al., 2019b; Basu et al., 2019)	$O(T^{1/8}/N^{3/8})$ (Basu et al., 2019)	$O(T^{1/4}/N^{3/4})$

once per several iterations. Such a strategy is commonly referred to as *local SGD* (Mangasarian, 1995; Zinkevich et al., 2010; Coppola, 2015; Stich, 2018; Zhou & Cong, 2018), but is also known in the literature under various other names (such as *parallel SGD*) (Yu et al., 2019b).

In the simplest version of synchronous local SGD, each client performs local SGD updates in parallel on the local data, and the server averages all clients iterates after every Δ updates.[1]

In this paper, we use an asynchronous model for client communication, where all the client iterates evolve at the same rate, but communicate with the server at arbitrary times decided individually by each client[2] In this asynchronous model, variants of which were recently considered by (Stich, 2018; Basu et al., 2019; Stich & Karimireddy, 2019), each client takes the same number of steps per unit time according to a global clock. So the local iterations are in synchrony with respect to the global clock, but the asynchrony comes with the communication of the clients. As we will discuss the only assumption on the client communication we make is that each client communicates to the server at least once every $\Delta \geq 1$ rounds.

In this paper, we analyze the local SGD algorithm in this asynchronous communication setting.[3] We consider both smooth strongly convex as well as smooth nonconvex objectives. Under a standard assumption of bounded second moment (Rakhlin et al., 2012; Yu et al., 2019b; Stich, 2018; Basu et al., 2019), we show that the convergence rate of our proposed local SGD with asynchronous update matches that of synchronous local SGD where all clients communicate together every Δ rounds. This is the first result showing that local SGD even with asynchronous updates from the clients performs as well as the synchronous local SGD in the heterogeneous (non-identical) data setting. Previous convergence bounds from (Basu et al., 2019) were considerably weaker under similar assumptions. In other words, while due to asynchronous nature of communication the gradient information for some of the clients at the server might be stale, somewhat surprisingly our results show that as long as this period of staleness is bounded by some Δ, we get similar convergence behavior (under our assumptions) as a synchronous local SGD where the communication happens every fixed Δ rounds.

For simplicity of discussion, in this section we ignore the dependence on various parameters such as strong convexity, smoothness, variance bound, and gradient norm bound. Table 1 summarizes our main theoretical results.

For smooth strongly convex functions, we show that an averaged iterate $\hat{x}_T$ satisfies (see Theorem 2.2 for a precise statement): $E[f(\hat{x}_T) - f(x^\star)] = O\left(1/NT + \Delta^2/T^2\right)$, where $x^\star$ is a minimizer of f. In particular, we can set $\Delta = O(\sqrt{T/N})$ and reach asymptotically the same convergence rate of minibatch SGD of $O(1/NT)$.[4] This matches the bound on Δ known with synchronous local SGD (Basu et al., 2019, Corollary 3) and improves the previously best known bound on Δ in our asynchronous update setting (Basu et al., 2019, Corollary 5) by a square factor, from $(T/N)^{1/4}$ to $\sqrt{T/N}$.

For smooth nonconvex functions, we show that iterate x_t satisfies (see Theorem 2.4 for a precise statement): $\frac{1}{T}\sum_{t=0}^{T} \mathbb{E}[\|\nabla f(x_t)\|^2] = O\left(1/\sqrt{NT} + N\Delta^2/T\right)$.

In particular, we can set $\Delta = O(T^{1/4}/N^{3/4})$ and reach asymptotically the same convergence rate of minibatch SGD of $O(1/\sqrt{NT})$. Again, this matches the bound on Δ known with synchronous local SGD, see e.g., (Basu et al., 2019, Corollary 2) or (Yu et al., 2019b, Corollary 3). Similarly, this improves the previously best known bound on Δ in our asynchronous update setting (Basu et al., 2019, Corollary 4) by a square factor, from $T^{1/8}/N^{3/8}$ to $T^{1/4}/N^{3/4}$.

[1] Local SGD is different from minibatch SGD where the averaging happens after every iteration, but more closely related to minibatch SGD with Δ times larger batchsizes on each client.

[2] This model also subsumes the standard synchronous model where all clients communicate in each round.

[3] Stich (2018) considered a different variant of asynchronous local SGD, where each client has identical data distribution; however, the clients can evolve their computation at slightly different rates resulting in delayed stochastic gradient updates.

[4] It is desirable to have larger Δ as it translates into lower communication overhead.

Finally, we also empirically evaluate multiple instantiations of our scheme to demonstrate the various factors affecting the performance in practice.

Comparison to Federated Averaging. Federated Averaging (FedAvg) (McMahan et al., 2017; Konečnỳ et al., 2016), one of the most widely used algorithm for FL applications, is a variant of local SGD. In the basic version of FedAvg, the participating devices (clients) are sampled randomly in each round, and only those devices perform the SGD steps on their local data and send back the results to the server. Notice that even though related, our model, which is the asynchronous communication version of local SGD as also discussed in (Stich, 2018; Basu et al., 2019; Stich & Karimireddy, 2019), is different in that we assume all the clients perform local updates in each round. Standard analyses of FedAvg (as defined above), in the non-identical data setting, rely on the assumption that the server gets access to a *random* subset of clients, and if the clients are unavailable, then either the assumptions are violated or we run into straggler issues (Kairouz et al., 2019). A common practical heuristic in this case is to over sample and then take the first few responding clients, but this in fact constructs a biased set at the server (since more powerful clients are selected). Another point of distinction is that, unlike our model, FedAvg does not capture scenarios where clients communicate at their convenience.

1.2. Related Work

With the increasing popularity of federated learning there has been lots of recent interest in understanding the convergence properties of local SGD. We refer the reader to recent excellent surveys (Kairouz et al., 2019; Li et al., 2020a) for a more comprehensive review of developments in federated learning algorithms. To emphasize the difference of our results from previous ones, we categorize the previous results into different (non-exclusive) groups. Also note that not all these previous results had strong theoretical convergence guarantees which is of focus in this paper.

Identical Data Distribution on Clients. A line of work has focused on analyzing local SGD under identical data distribution on clients (Zhou & Cong, 2017; Jiang & Agrawal, 2018; Wang & Joshi, 2018; Stich, 2018; Stich & Karimireddy, 2019; Haddadpour et al., 2019; Khaled et al., 2019b; Wang & Joshi, 2018). If all the clients have identical data distribution, then that would result in unbiasedness of gradients at every client, resulting in slightly easier analysis. However, as discussed earlier, this is not a reasonable assumption for FL applications where data available locally fail to represent the overall distribution. We make no assumptions on the local data distributions.

Synchronous/Random Client Communication with Non-identical Data Distributions. Another line of work has focused on synchronous local SGD with non-identical data distribution across clients wherein all clients communicate their local parameter to the server every fixed Δ rounds (Yu et al., 2019b; Haddadpour & Mahdavi, 2019; Khaled et al., 2019b;a; Wang et al., 2019c; Basu et al., 2019; Li et al., 2019b). Again as discussed, the requirement of full device (synchronous) participation is not generally practical in FL. Another variant is that a set of random clients communicate in each round (McMahan et al., 2017; Li et al., 2018; 2019a), which again is hard to ensure or verify in practice. We make no assumptions on the client communication patterns except that each client participates at least once in Δ rounds. In particular, synchronous and random client communication can be thought as special cases of our setup.

Extensions to SGD. We note that more sophisticated local stochastic gradient methods have also been considered, for example with momentum (Yu et al., 2019a; Wang et al., 2019b) with gradient compression (Jiang & Agrawal, 2018; Basu et al., 2019; Reisizadeh et al., 2020), with other various variance-reduction methods (Liang et al., 2019; Sharma et al., 2019; Karimireddy et al., 2019). Our work is complimentary to these approaches, and focuses on the vanilla version of local SGD commonly used in practice.

The most relevant result to us is that of (Basu et al., 2019), who considered an asynchronous communication setting similar to that discussed in this paper. They also study additional gradient compression techniques not addressed here. For completeness, we include a discussion of results from (Basu et al., 2019) in Section 2.1. Our convergence results are much tighter, for both strongly convex and nonconvex cases, suggesting that the clients can communicate much less frequently to achieve the same convergence guarantee.

Preliminaries. Since f in (1) is an average of $f^{(i)}$'s, we express it as $f = \text{avg}_i(f^{(i)})$ to simplify the presentation. Assume all functions $f^{(i)} : \mathbb{R}^d \to \mathbb{R}$. We review some basic optimization concepts in Appendix A. In this paper, we assume that all $f^{(i)}$'s (and therefore f) are L-smooth. See Table 2 for a full list of notation.

2. Our Algorithm

In Algorithm AsyncCommSGD (Algorithm 1), we present our local SGD approach. We assume that there are N clients, labeled $1, \ldots, N$. For each client i, we maintain two parameter vectors: $x_t^{(i)}$, the local parameter vector on the client, and $y_t^{(i)}$, the server copy of the last (at round t or earlier) parameter vector received by the server from client i. The server maintains a global parameter vector x_t which accumulates local updates.

At each round t, each client performs a stochastic gradi-

Table 2: Notation used in the paper.

Notation	Explanation
Problem parameters	
N	The number of clients
$f^{(i)}$	Function at the ith client
f	The objective function: $f = \frac{1}{N}\sum_{i\in[N]} f^{(i)}$
$f_{\max}$	Bound on the objective value: $f_{\max} = f(x_0) - f(x^\star)$
L	Smoothness parameter of f and $f^{(i)}$, $i \in [N]$: $\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|$
λ (Theorem 2.2)	Strong convexity parameter of f: $f(x) \ge f(y) + \langle \nabla f(y), x-y\rangle + \frac{\lambda}{2}\|x-y\|^2$
Δ	Maximum gap between communications of a single client
$\nabla F^{(i)}(x,\theta)$	Stochastic gradient computed by ith client at point x. θ is a parameter controlling randomness
$\mathbb{E}[\cdots]$	Expectation over stochastic randomness: $\mathbb{E}[\cdots \mid \theta_t^{(i)}, i \in [N], t \in [T]]$
$\mathbf{G}_{\max}$	Maximum stochastic gradient norm: $\mathbb{E}[\|\nabla F^{(i)}\|^2] \le \mathbf{G}_{\max}^2$
σ^2	Variance of stochastic gradients: for all i, $\mathbb{E}_\theta[\|\nabla F^{(i)}(x,\theta) - \nabla f^{(i)}(x)\|^2] \le \sigma^2$
Sequences	
$\mathrm{avg}_i(\ldots)$	Average over all clients: $\frac{1}{N}\sum_{i\in[N]}\cdots$
$x_t^{(i)}$	Value stored by ith client at tth iteration
x_t	Value stored by the server at tth iteration
$y_t^{(i)}$	The latest value communicated to ith client from the server before tth iteration
γ_t	Gradient step size (learning rate)
$\mathbf{G}_t^{(i)}$	Stochastic gradient computed by ith client at tth iteration at point $x_t^{(i)}$: $\mathbf{G}_t^{(i)} = \nabla F^{(i)}(x_t^{(i)}; \theta_t^{(i)})$
z_t	A virtual sequence $z_{t+1} = z_t - \gamma_t \,\mathrm{avg}_i(\mathbf{G}_t^{(i)})$
$\mathcal{C}_t$	A set of clients communicating at tth iteration
$\rho^{(i)}(t)$ (App. B)	Last communication round for machine i up to iteration t: $\rho^{(i)}(t) = \max(\{\tau \le t \mid i \in \mathcal{C}_\tau\} \cup \{0\})$

ent descent step. Then a subset of clients $\mathcal{C}_{t+1}$ (possibly empty) send their updates to the server. The server aggregates them, updates the global parameter vector x_{t+1} and $y_t^{(i)}$, and sends it back to clients from $\mathcal{C}_{t+1}$. The clients update their local parameter vector $x_{t+1}^{(i)}$ to global parameter vector x_{t+1}. A similar local SGD with asynchronous update algorithm was also considered by Basu et al. (2019, Algorithm 2) with the additional gradient compression operator.

For constructing x_{t+1}, we average over the latest parameter vectors of all the clients currently available on the server. This on the first glance might look problematic as some of these updates might be stale say if a client has not communicated recently. This is also different from a typical Federated Averaging scheme, where the averaging is done only over the parameter vectors of a random set of communicated clients. However, this averaging is crucial for our analysis. In fact, Federated Averaging scheme based on clients communicating randomly will not have a good convergence rates if there are rounds in which only a small set of clients communicate, something that our analysis does not suffer from. Moreover, if some clients don't communicate, then it's impossible to find a minimizer of the objective, since each client could have different data distribution. Therefore, it is crucial to have an upper bound (Δ) on the maximum delay between each clients update times.

Note that the set $\mathcal{C}_{t+1}$ doesn't need to be known to the server; for example, a round can start and end at a prespecified global clock time, and all the clients that communicate within this time window then form the set $\mathcal{C}_{t+1}$. This way the communication patterns are controlled by the clients and is not at the behest of the server. For example, in a FL setting, if a client has connectivity issue, then client could communicate back when the connectivity is restored. For simplicity of presentation, in Algorithm AsyncCommSGD, we assume that server knows $\mathcal{C}_{t+1}$.

2.1. Convergence Analysis

In this section we present our main convergence results for local SGD with asynchronous updates, obtained by running Algorithm AsyncCommSGD on smooth functions, both strongly convex and nonconvex. Missing details from this section are collected in Appendix B. We use the following standard assumptions.

i. **Smoothness:** All local functions $f^{(i)}$ ($i \in [N]$) are L-smooth (see Definition 3, Appendix A)

ii. **Bounded second moment:** There exists a $\mathbf{G}_{\max} > 0$ such that $\mathbb{E}[\|\nabla F^{(i)}(x)\|^2] \le \mathbf{G}_{\max}^2$ for all $x \in$

Algorithm 1 AsyncCommSGD

parameters: $\{\gamma_t\}$ – step sizes, T – the number of rounds, x_0 – starting point, $\{\mathcal{C}_t\}$ – for each t, which clients communicate at iteration t

On each client $i \in [N]$**:**
  $x_0^{(i)} \leftarrow x_0$ // Local parameters on the client
  $y_0^{(i)} \leftarrow x_0$ // Last parameters received by the client

for $t = 0 \dots T-1$ **do**
 ClientUpdate: // Run on each client i
  $\mathbf{G}_t^{(i)} \leftarrow$ stochastic gradient for $f^{(i)}$ at $x_t^{(i)}$
  $v_{t+1}^{(i)} \leftarrow x_t^{(i)} - \gamma_t \mathbf{G}_t^{(i)}$ // Local SGD step
  if $i \in \mathcal{C}_{t+1}$ **then**
    Send $\delta_{t+1}^{(i)} := v_{t+1}^{(i)} - y_t^{(i)}$ to the server
    Receive x_{t+1}
    $x_{t+1}^{(i)} \leftarrow x_{t+1}$
    $y_{t+1}^{(i)} \leftarrow x_{t+1}$
  else
    $x_{t+1}^{(i)} \leftarrow v_{t+1}^{(i)}$
    $y_{t+1}^{(i)} \leftarrow y_t^{(i)}$
  end if

 ServerUpdate: // Run on the server
  Receive $\delta_{t+1}^{(i)}$ from clients $i \in \mathcal{C}_{t+1}$
  $x_{t+1} \leftarrow x_t + \frac{1}{N}\sum_{i \in \mathcal{C}_{t+1}} \delta_{t+1}^{(i)}$ // Aggregate updates
  Send x_{t+1} to clients $i \in \mathcal{C}_{t+1}$
end for

$\mathbb{R}^d, i \in [N]$, where $\nabla F^{(i)}(x)$ is an unbiased stochastic gradient of $f^{(i)}$ at x. This is a standard assumption in the SGD literature (Rakhlin et al., 2012; Stich, 2018; Stich et al., 2018; Yu et al., 2019b; Basu et al., 2019) etc.[5] Relaxing this bounded gradient assumption, as achieved through different parameters in recent distributed SGD/GD literature (see, e.g., (Wang & Joshi, 2018; Khaled et al., 2019b; Haddadpour et al., 2019; Yu et al., 2019a; Li et al., 2020b; Wang et al., 2019a; Li et al., 2018)) is an interesting open problem. The second moment assumption also implies a bound on the variance, $\mathbb{E}[\|\nabla F^{(i)}(x) - \nabla f^{(i)}(x)\|^2] \leq \sigma^2$ for all $x \in \mathbb{R}^d, i \in [N]$ (where $\sigma^2 \leq G^2$).

Let $\mathbf{G}_t^{(i)} = \nabla F^{(i)}(x_t^{(i)})$ be a stochastic gradient computed by the ith client at the t-th round. Recall that $\mathcal{C}_{t+1}$ is the set of clients communicating at the t-th round. Then our update equation on the clients has the following form:

$$x_{t+1}^{(i)} = \begin{cases} x_t^{(i)} - \gamma_t \mathbf{G}_t^{(i)}, & i \notin \mathcal{C}_{t+1} \\ x_{t+1}, & i \in \mathcal{C}_{t+1}, \end{cases}$$

where x_{t+1} is the server model accumulating updates communicated to the server.

Let $\rho^{(i)}(t) = \max\{\tau \leq t | i \in \mathcal{C}_\tau\}$ be the last round before t such that the client i communicates with the server at this round. Since the server received updates from client i up to this round, we have: $x_t = x_0 - \text{avg}_i \left(\sum_{\tau=0}^{\rho^{(i)}(t)} \gamma_\tau \mathbf{G}_\tau^{(i)}\right)$.

To show convergence rates, we investigate $\|x_t - x^\star\|^2$, where $x^\star$ is a minimizer of f. Unfortunately, x_t has a rather complicated update equation. To address this issue, we define a virtual sequence $\{z_t\}_{t \in \mathbb{N}}$ the following way:

Definition 1 (Virtual sequence)

$$z_t = x_0 - \sum_{\tau=0}^{t-1} \gamma_\tau \, \text{avg}_i(\mathbf{G}_\tau^{(i)}).$$

Similar virtual sequences have been utilized before in decentralized optimization under various contexts (Lian et al., 2017; Yuan et al., 2016; Nedić et al., 2018; Stich, 2018).

We show (see Proposition 2.1) that z_t's are close to x_t and $x_t^{(i)}$'s for all clients i. The advantage of working with z_t is its simple update equation: $z_{t+1} = z_t - \gamma_t \, \text{avg}_i(\mathbf{G}_t^{(i)})$, which makes the analysis cleaner. The following proposition bounds the distance between the virtual z_t and the local $x_t^{(i)}$ in terms of the parameter Δ. This proposition and its proof is similar to (Stich, 2018, Lemma B.1).

Proposition 2.1 (Distance Bound) *Let $\{\gamma_t\}$ be a non-increasing sequence such that $\gamma_t/\gamma_{t+\Delta} \leq 2$. Then in Algorithm* AsyncCommSGD *for each client $i \in [N]$:*

$$\max\left(\mathbb{E}[\|z_t - x_t^{(i)}\|^2], \mathbb{E}[\|z_t - x_t\|^2]\right) \leq 72\gamma_t^2 \mathbf{G}_{\max}^2 \Delta^2$$

Analysis for Strongly Convex Functions. We now assume that $f^{(i)}$ for $i \in [N]$ are L-smooth functions and $f = \text{avg}_i(f^{(i)})$ is a λ-strongly convex (Definitions 2 and 3).

In Theorem 2.2, we state the convergence theorem for strongly convex functions when using Algorithm AsyncCommSGD. Instead of x_t, we consider a weighted average $\hat{x}_t = \frac{1}{S_T}\sum_{t=0}^{T} w_t x_t$ where $w_t = (\Delta + t)^2$. Therefore, the sequence $\{\hat{x}_t\}_{t \in [T]}$ can be easily computed from the sequence $\{x_t\}_{t \in [T]}$. This choice of w_t puts more weights on later rounds. A similar $\hat{x}_t$ was also considered in the previous related work of (Stich, 2018; Basu et al., 2019).

Our analysis starts by bounding $\mathbb{E}[\|z_{t+1} - x^\star\|^2]$, where $x^\star$ is the minimizer for f. Using properties of strong convexity and smoothness, along with Proposition 2.1, we establish,

$$\mathbb{E}[\|z_{t+1} - x^\star\|^2] \leq \left(1 - \frac{\gamma_t \lambda}{2}\right) \mathbb{E}[\|z_t - x^\star\|^2] - 2\gamma_t(f(z_t) - f(x^\star)) + \gamma_t^2 \frac{\sigma^2}{N} + \frac{300\gamma_t^3}{\lambda} L^2 \mathbf{G}_{\max}^2 \Delta^2.$$

[5] A consequence of this assumption is that it also bounds gradient difference. For example $\|\nabla f^{(i)}(x) - \nabla f^{(i')}(x)\| \leq \mathbf{G}_{\max}$ for any two clients $i, i' \in [N]$ and for all $x \in \mathbb{R}^d$.

This along with a recurrence relation from (Stich et al., 2018) gives a bound in terms of $\hat{z}_T$

$$\mathbb{E}[f(\hat{z}_T)] - f(x^\star) \le \frac{\mathbf{G}_{\max}^2(\Delta + 4^L/\lambda)^3}{4\lambda S_T} + \frac{2T(T + 2\Delta + 8^L/\lambda)}{\lambda S_T}\frac{\sigma^2}{N} + \frac{10^4 T}{\lambda^3 S_T} L^2 \mathbf{G}_{\max}^2 \Delta^2.$$

Now using Proposition 2.1, that shows that the virtual sequence $\hat{z}_T$ is close to $\hat{x}_T$, yields the following result.

Theorem 2.2 *Let $f^{(i)}$'s for $i \in [N]$ be L-smooth functions and $f = \sum_{i\in[N]} f^{(i)}$ be a L-smooth λ-strongly convex function. Let $w_t = (\Delta + t)^2$, $S_t = \sum_{t=0}^T w_t \ge T^3$, $\hat{x}_t = \frac{1}{S_T}\sum_{t=0}^T w_t x_t$. After T rounds of Algorithm* AsyncCommSGD *with $\gamma_t = \frac{8}{\lambda(t+\Delta+8^L/\lambda)}$, we have*

$$\mathbb{E}[f(\hat{x}_T)] - f(x^\star) = O\Bigg(\frac{L\mathbf{G}_{\max}^2(\Delta + {}^L/\lambda)^3}{\lambda^2 S_T} + \frac{LT(T + \Delta + {}^L/\lambda)}{\lambda^2 S_T}\frac{\sigma^2}{N} + \frac{L^3\mathbf{G}_{\max}^2\Delta^2 T}{\lambda^4 S_T}\Bigg).$$

In particular, for a fixed Δ, we get a convergence rate of $O({}^1/\sqrt{NT} + {}^N/_T)$ (ignoring other terms). Using the fact that $S_T \ge T^3$, we get the following result:

Corollary 2.3 *Under assumptions of Theorem 2.2, if $T \ge N$, $\Delta \le \sqrt{{}^T/_N}$ and $\Delta \ge {}^L/\lambda$, then*

$$\mathbb{E}[f(x_T) - f(x^\star)] = O\left(\frac{L\sigma^2}{\lambda^2 TN} + \frac{L\mathbf{G}_{\max}^2}{\lambda^2 TN}\left(\frac{1}{\sqrt{TN}} + \frac{L^2}{\lambda^2}\right)\right).$$

In particular, in terms of T and N we recover the standard minibatch SGD convergence rate for strongly convex functions of $O({}^1/_{TN})$. In other words, for achieving this convergence rate within T rounds, we need each client to communicate ${}^T/_\Delta = \Omega(\sqrt{TN})$ many times.

Compare this with the corresponding result from (Basu et al., 2019, Corollary 5) (we change notation to match ours and consider the case without compression):

$$\mathbb{E}[f(\hat{x}_T) - f(x^\star)] = O\left(\frac{\mathbf{G}_{\max}^2\Delta^3}{\lambda^2 T^3} + \frac{(T+\Delta)}{\lambda^2 T^2}\frac{\sigma^2}{N} + \frac{\mathbf{G}_{\max}^2\Delta^4}{\lambda^3 T^2}\right)$$

In our case, the last term has much better dependence on Δ (Δ^2 instead of Δ^4). As a consequence, we can improve bound on Δ from $({}^T/_N)^{1/4}$ to square of that: $\sqrt{{}^T/_N}$.

Analysis for Nonconvex Functions. We now assume that $f^{(i)}$ for $i \in [N]$ are L-smooth functions (see Definition 3), but f is not necessarily convex. We use a fixed step size γ, and therefore the condition of Proposition 2.1 is always satisfied, and we can directly use the iterates x_t produced by Algorithm AsyncCommSGD.

We again consider the virtual sequence z_t per Definition 1. In this case, our analysis is based on bounding $\frac{1}{T}\sum_{t=0}^T \mathbb{E}[\|\nabla f(z_t)\|^2]$ with

$$\frac{4(f(z_0) - \mathbb{E}[f(z_T)])}{\gamma T} + 4\gamma(10\gamma L^2\mathbf{G}_{\max}^2\Delta^2 + \frac{L\sigma^2}{2N}).$$

Then we bound $\|\nabla f(x_t)\|$ in terms of $\|\nabla f(z_t)\|$ again using Proposition 2.1 as,

$$\mathbb{E}[\|\nabla f(x_t)\|^2] \le 2\mathbb{E}[\|\nabla f(z_t)\|^2] + 36\gamma_t^2 L^2 \mathbf{G}_{\max}^2\Delta^2.$$

The following theorem follows from these inequalities.

Theorem 2.4 *Let $f_{\max} = f(x_0) - f(x^\star)$. After T rounds of Algorithm* AsyncCommSGD *with step size $0 \le \gamma \le {}^1/_{(18L)}$, we have*

$$\frac{1}{T}\sum_{t=0}^T \mathbb{E}[\|\nabla f(x_t)\|^2] = O\left(\frac{f_{\max}}{\gamma T} + \gamma^2 L^2 \mathbf{G}_{\max}^2\Delta^2 + \gamma\frac{L\sigma^2}{N}\right).$$

The next corollary follows by substituting suitable γ and other parameters.

Corollary 2.5 *Let $f_{\max} = f(x_0) - f(x^\star)$. In Theorem 2.4, using step size $\gamma = {}^{\sqrt{N}}/_{(L\sqrt{T})}$, we get*

$$\frac{1}{T}\sum_{t=0}^T \mathbb{E}[\|\nabla f(x_t)\|^2] = O\left(\frac{Lf_{\max}}{\sqrt{NT}} + \frac{N}{T}\mathbf{G}_{\max}^2\Delta^2 + \frac{L\sigma^2}{\sqrt{NT}}\right).$$

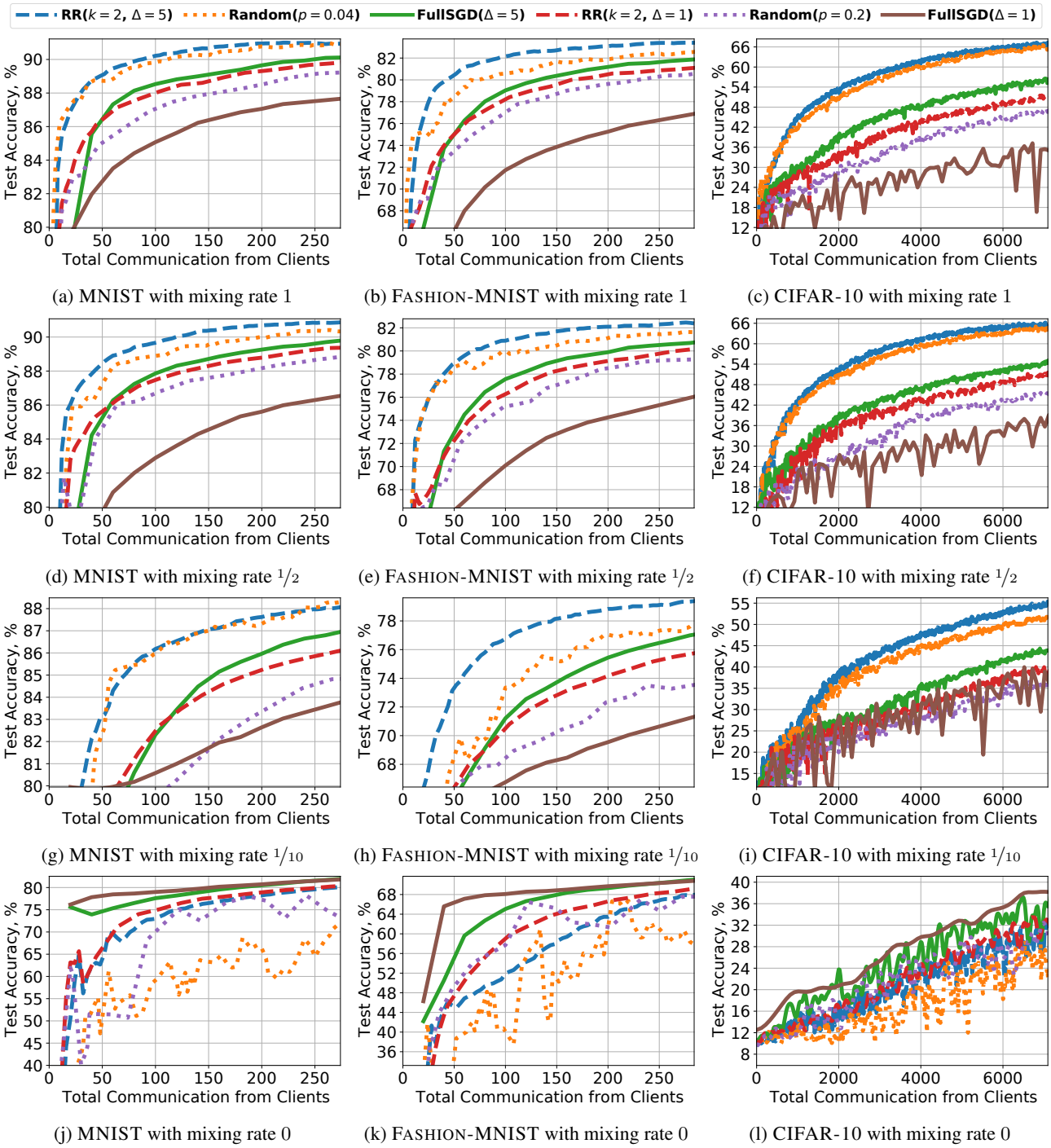
Using step size $\gamma = {}^{\sqrt{N}}/_{(L\sqrt{T})}$, if $T > N^3$ and $\Delta \le {}^{T^{1/4}}/_{N^{3/4}}$, we get

$$\frac{1}{T}\sum_{t=0}^T \mathbb{E}[\|\nabla f(x_t)\|^2] = O\left(\frac{Lf_{\max}}{\sqrt{NT}} + \frac{\mathbf{G}_{\max}^2}{\sqrt{NT}} + \frac{L\sigma^2}{\sqrt{NT}}\right).$$

In particular, for a fixed Δ, we get a convergence rate of $O({}^1/\sqrt{NT} + {}^N/_T)$ (ignoring other terms). If $T > N^3$ and $\Delta \le {}^{T^{1/4}}/_{N^{3/4}}$, then we recover the standard minibatch SGD convergence rate of $\approx O({}^1/\sqrt{NT})$. In other words, for achieving this convergence rate within T rounds, we need each client to communicate $\Omega(N^{3/4}T^{3/4})$ many times.

Again, compare this with the corresponding result from (Basu et al., 2019, Corollary 4):

$$\mathbb{E}[\|\nabla f(\hat{x}_t)\|^2] = O\left(\frac{\sigma\sqrt{f_{\max}}}{\sqrt{NT}} + \frac{f_{\max}N\mathbf{G}_{\max}^2\Delta^4}{\sigma^2 T}\right).$$



(a) MNIST with mixing rate 1 (b) FASHION-MNIST with mixing rate 1 (c) CIFAR-10 with mixing rate 1

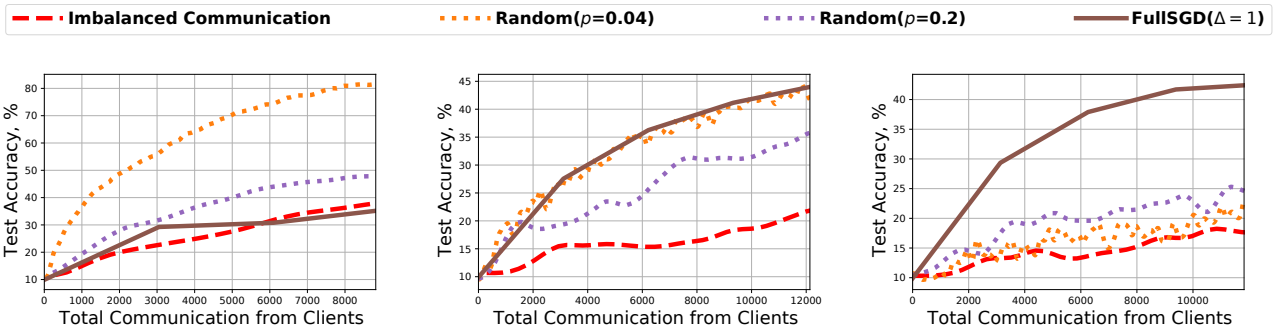
(d) MNIST with mixing rate $1/2$ (e) FASHION-MNIST with mixing rate $1/2$ (f) CIFAR-10 with mixing rate $1/2$

(g) MNIST with mixing rate $1/10$ (h) FASHION-MNIST with mixing rate $1/10$ (i) CIFAR-10 with mixing rate $1/10$

(j) MNIST with mixing rate 0 (k) FASHION-MNIST with mixing rate 0 (l) CIFAR-10 with mixing rate 0

Figure 1: For each mixing rate $\mu \in \{1, 1/2, 1/10, 0\}$, we show the test accuracy as a function of total communication (the number of communicated client models). Left column corresponds to MNIST dataset, middle – to FASHION-MNIST, right – to CIFAR-10 (to improve the presentation, the results for CIFAR-10 with mixing rate 1 are slightly smoothed). We omit IMBALANCED COMMUNICATION here for clarity. The results suggest that a full synchronous update of all the clients to the server is unnecessary as long as the local data distributions are not completely disjoint.

In our case, the second term has much better dependence on Δ (Δ^2 instead of Δ^4). As a consequence, we can improve bound on Δ from $T^{1/8}/N^{3/8}$ to square of that: $T^{1/4}/N^{3/4}$.



(a) CIFAR-10 with mixing rate $1/2$ (b) CIFAR-10 with mixing rate $1/10$ (c) CIFAR-10 with mixing rate 0

Figure 2: Results on CIFAR-10 for Resnet-34 with $\mu \in \{1/2, 1/10, 0\}$. The conclusions are same as in Figure 1 even with this bigger network. We omit FULLSGD($\Delta = 5$), FULLSGD($\Delta = 25$), RR($k = 2$, $\Delta = 1$), and RR($k = 2$, $\Delta = 1$) since their results are again similar to considered algorithms, as also observed in Figure 1. To improve the presentation, the plots were slightly smoothed.

3. Experimental Evaluation

In this section, we demonstrate the effectiveness of local SGD with asynchronous updates as compared to regular SGD or a synchronous local SGD where all communication takes place together. Our focus will be on illustrating the dependence of model accuracy on the total communication. We also investigate the role of (non)iidness of the clients data distributions.

3.1. Datasets and Models

We perform our evaluation on the following datasets: MNIST, FASHION-MNIST, CIFAR-10[6]. We use a single-machine simulation of FL computation with 10 clients. In this setup, the running time doesn't take into account the actual communication time, hence not informative.

Each of our dataset has 10 classes, and we split our data across 10 clients in the following manner. Each client is associated with a class. We define a mixing rate μ which measures how identical the data distributions across different clients are: for each client, $(1-\mu)$ fraction of data is selected from the class corresponding to the client, while μ fraction is selected from a random class. In particular, for $\mu = 1$ the data for all clients is identically distributed, and for $\mu = 0$ each client holds only data corresponding to its class. We consider $\mu \in \{0, 1/10, 1/2, 1\}$ and show how μ affects convergence of our compared algorithms.

MNIST and FASHION-MNIST. We use a one-layer neural network with softmax activation. At each round, clients process 10^3 samples (within the round, the client locally performs minibatch gradient descent with batch size 20).

[6]Dataset and detailed network descriptions are given in Appendix C The entire code is provided in supplementary material.

CIFAR-10. Here we use two networks. The first one is a shallow convolutional neural network with 3 convolutional layers with ReLU activation and max pooling and two dense layers with ReLU and softmax activations. The second network is the deep ResNet-34 (He et al., 2016) without batch normalization.

3.2. Compared Approaches

Since Algorithm 1 is quite general it covers multiple client communication scenarios depending on the selection of $\mathcal{C}_t$. Here, we select a few of them to compare (all of whom are captured by our Algorithm 1) to understand the role of various factors involved in FL.

Synchronous Local SGD. Each Δ rounds, all clients communicate with the server. Denoted by FULLSGD(Δ). Case $\Delta = 1$ corresponds to a regular distributed SGD.

Round Robin. Each Δ rounds, k clients communicate with the server: first clients $1, \ldots, k$ communicate, then clients $k+1, \ldots, 2k$ communicate, etc. In our experiments, we select $k = 2$, i.e. $1/5$ fraction of clients communicates each Δ rounds. Denoted by RR(k, Δ).

Random Communication. At each round, each client communicates with the server with probability p. This scheme is closely related to Federated Averaging where the participating clients are sampled randomly (McMahan et al., 2017). In our experiments, we select $p = 1/5$ and $p = 1/25$. These approaches have the same expected amount of communication as RR$(2, 1)$ and RR$(2, 5)$ respectively, but it is not guaranteed that each client communicates at least once each Δ rounds. Denoted by RANDOM(p).

Imbalanced Communication. Client i communicates every i rounds. i.e. client 1 communicates every round and client 10 communicates every 10 rounds, leading to very imbalanced communication.

Among these scenarios, Round Robin, Random Communication, and Imbalanced Communication are all examples of asynchronous communication scenarios.

3.3. Discussion of Results

We present our results in Figures 1 and 2. For each mixing rate μ, we show how accuracy as a function of the total communication, which is measured as the number of communicated local models. We note that, while the plots are cropped to match the lowest total communication among the algorithm, for most approaches their accuracy continue increasing. We make the following observations.

Synchronous vs. Asynchronous: Among our implemented methods, FULLSGD(1) has the largest communication cost per round, FULLSGD(5), RR$(2,1)$ and RANDOM($1/5$) have approximately $1/5$ of that communication per round, and RR$(2,5)$ and RANDOM($1/25$) have approximately $1/25$ of that communication per round. In Figure 1, approaches with similar communication per round achieve similar accuracy. For example, Figure 1e shows that FULLSGD(1) achieves 76% accuracy, approaches with $1/5$ of communication per round achieve $80\% - 81\%$ accuracy, and approaches with $1/25$ communication per round achieve $81.5\% - 82.5\%$ accuracy. The fact that RR$(2,1)$ achieves similar guarantees as FULLSGD(5) supports our theoretical conclusions that local SGD with asynchronous communication (like RR) performs as well as the synchronous local SGD (FULLSGD). The same conclusions also hold with Resnet-34 experiments (Figure 2). With IMBALANCED COMMUNICATION, even though some machine communicate much more often than others, for $\mu = 1/2$ (Figure 2a) it outperforms FULLSGD(1) (38% against 30%). However, it's noticeably outperformed by RANDOM approaches which have much lower communication per round while having similar communication gap.

Role of Δ: FULLSGD(5), RR$(2,1)$ and RANDOM($1/5$) have similar communication requirements; however RANDOM($1/5$) is slightly outperformed by the other two. For example, 80.9% accuracy for RANDOM($1/5$) against 81.2% (for RR$(2,1)$) and 82% (for FULLSGD(5)) in Figure 1b. Similarly, in the same plot, RANDOM($1/25$) is outperformed by RR$(2,5)$ (82.4% against 83.4% accuracies). The gap becomes more prominent when μ decreases. For example, in Figure 1h, RANDOM($1/5$) achieves 74.6% accuracy, while FULLSGD(5) and RR$(2,1)$ achieve 76% and 77.5% accuracy, respectively.

Accuracy of RANDOM shows large fluctuations during training (see e.g. Figure 1k). We suspect that the reason for this behavior is that, unlike other approaches, RANDOM doesn't always guarantee that each client communicates within Δ rounds (the guarantee only holds in expectation). When the data is non-iid, IMBALANCED COMMUNICATION has the worst accuracy to communication ratio (see Figure 2), which is expected: while its communication gap is 10, the average communication per round is reduced only by the factor of ≈ 3.5.

Role of (non)iidness: Comparing Figures 1a and 1d, 1b and 1e, 1c and 1f, we see that difference between cases $\mu = 1$ and $\mu = 1/2$ is minor. When further decreasing μ to $1/10$, for RR$(2,5)$ we observe 3% drop in accuracy for MNIST and FASHION-MNIST and 11% drop for CIFAR-10. However, the largest accuracy drop for RR$(2,5)$ is experienced when μ changes from $1/10$ to 0: 8%, 12% and 20% respectively. Note that when $\mu = 0$ all the local distributions have support on different classes.

In general, for $\mu \in \{1, 1/2, 1/10\}$, approaches with less communication show better convergence, and the gap is more prominent at larger values of μ. The fact that this happens even when $\mu = 1/10$ is interesting, suggesting that approaches like RR$(2,5)$ works well even when data distributions are far from identical. However, when the data distribution is completely non-iid ($\mu = 0$), we observe the opposite behavior: approaches with less communication achieve lower accuracy (e.g., 68% accuracy for RR$(2,5)$ on FASHION-MNIST) compared to approaches with full communication (e.g., 71% accuracy for FULLSGD(1) on FASHION-MNIST). Again, the conclusions are the same with Resnet-34 (Figure 2). Due to its large communication gap, IMBALANCED COMMUNICATION suffers the most when μ decreases: it reaches 38% when $\mu = 1/2$ but drops to 20% when $\mu = 1/10$. Somewhat surprisingly, it doesn't drop further when $\mu = 0$ (18.5%).

4. Concluding Remarks

In this paper we demonstrated that we can significantly relax the communication requirements on the clients to capture practical scenarios and still achieve the standard convergence rates in a Federated Learning setting. We emphasize that the remaining assumption on the communication gap is necessary: when it's unbounded, learning algorithms can easily diverge.

One possible future direction is to eliminate bounded gradient assumption: $\mathbb{E}[\|\nabla F^{(i)}t\|^2] \leq \mathbf{G}_{\max}^2$. One possible alternative is the assumption that gradients of local functions are not much different from that of the global function: $\|\nabla f^{(i)}(x) - \nabla f(x)\| < \kappa$ (Yu et al., 2019a; Li et al., 2020b; Wang et al., 2019a). Our experiments support the idea that deviation from the global data distribution is an important parameter. However, this above condition with κ doesn't fully capture our observations as our experiments show the accuracy improves dramatically even when only a small fraction of global data is present at every client, which is not enough to substantially decrease κ.

References

Basu, D., Data, D., Karakus, C., and Diggavi, S. Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations. *arXiv preprint arXiv:1906.02367 (Also in NeruIPS 2019)*, 2019.

Coppola, G. F. Iterative parameter mixing for distributed large-margin training of structured predictors for natural language processing. 2015.

Haddadpour, F. and Mahdavi, M. On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*, 2019.

Haddadpour, F., Kamani, M. M., Mahdavi, M., and Cadambe, V. Local sgd with periodic averaging: Tighter analysis and adaptive synchronization. In *Advances in Neural Information Processing Systems*, pp. 11082–11094, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jiang, P. and Agrawal, G. A linear speedup analysis of distributed deep learning with sparse and quantized communication. In *Advances in Neural Information Processing Systems*, pp. 2525–2536, 2018.

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. Scaffold: Stochastic controlled averaging for on-device federated learning. *arXiv preprint arXiv:1910.06378*, 2019.

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

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local sgd on identical and heterogeneous data. *arXiv*, pp. arXiv–1909, 2019b.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 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.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020a.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019a.

Li, X., Yang, W., Wang, S., and Zhang, Z. Communication-efficient local decentralized sgd methods. *arXiv preprint arXiv:1910.09126*, 2019b.

Li, X., Yang, W., Wang, S., and Zhang, Z. Communication-efficient local decentralized sgd methods, 2020b.

Lian, X., Zhang, C., Zhang, H., Hsieh, C.-J., Zhang, W., and Liu, J. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *arXiv preprint arXiv:1705.09056*, 2017.

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.

Mangasarian, L. Parallel gradient distribution in unconstrained optimization. *SIAM Journal on Control and Optimization*, 33(6):1916–1925, 1995.

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.

Nedić, A., Olshevsky, A., and Rabbat, M. G. Network topology and communication-computation tradeoffs in decentralized optimization. *Proceedings of the IEEE*, 106(5):953–976, 2018.

Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*, 2011.

Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. 2012.

Reisizadeh, A., Mokhtari, A., Hassani, H., Jadbabaie, A., and Pedarsani, R. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International Conference on Artificial Intelligence and Statistics*, pp. 2021–2031, 2020.

Sharma, P., Khanduri, P., Bulusu, S., Rajawat, K., and Varshney, P. K. Parallel restarted spider–communication efficient distributed nonconvex optimization with optimal computation complexity. *arXiv preprint arXiv:1912.06036*, 2019.

Stich, S. U. Local sgd converges fast and communicates little. *arXiv preprint arXiv:1805.09767*, 2018.

Stich, S. U. and Karimireddy, S. P. The error-feedback framework: Better rates for sgd with delayed gradients and compressed communication. *arXiv preprint arXiv:1909.05350*, 2019.

Stich, S. U., Cordonnier, J.-B., and Jaggi, M. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pp. 4447–4458, 2018.

Wang, J. and Joshi, G. Cooperative sgd: A unified framework for the design and analysis of communication-efficient sgd algorithms. *arXiv preprint arXiv:1808.07576*, 2018.

Wang, J., Sahu, A. K., Yang, Z., Joshi, G., and Kar, S. Matcha: Speeding up decentralized sgd via matching decomposition sampling. *arXiv preprint arXiv:1905.09435*, 2019a.

Wang, J., Tantia, V., Ballas, N., and Rabbat, M. Slowmo: Improving communication-efficient distributed sgd with slow momentum. *arXiv preprint arXiv:1910.00643*, 2019b.

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, 2019c.

Yu, H., Jin, R., and Yang, S. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. *arXiv preprint arXiv:1905.03817*, 2019a.

Yu, H., Yang, S., and Zhu, S. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5693–5700, 2019b.

Yuan, K., Ling, Q., and Yin, W. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

Zhou, F. and Cong, G. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*, 2017.

Zhou, F. and Cong, G. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 3219–3227, 2018.

Zinkevich, M., Weimer, M., Smola, A. J., and Li, L. Parallelized stochastic gradient descent. In *NIPS*, volume 4, pp. 4. Citeseer, 2010.