SCAFFOLD: Stochastic Controlled Averaging for Federated Learning

Sai Praneeth Karimireddy [1 2] **Satyen Kale** [3] **Mehryar Mohri** [3 4] **Sashank J. Reddi** [3] **Sebastian U. Stich** [1]
Ananda Theertha Suresh [3]

Abstract

Federated Averaging (FEDAVG) has emerged as the algorithm of choice for federated learning due to its simplicity and low communication cost. However, in spite of recent research efforts, its performance is not fully understood. We obtain tight convergence rates for FEDAVG and prove that it suffers from 'client-drift' when the data is heterogeneous (non-iid), resulting in unstable and slow convergence.

As a solution, we propose a new algorithm (SCAFFOLD) which uses control variates (variance reduction) to correct for the 'client-drift' in its local updates. We prove that SCAFFOLD requires significantly fewer communication rounds and is not affected by data heterogeneity or client sampling. Further, we show that (for quadratics) SCAFFOLD can take advantage of similarity in the client's data yielding even faster convergence. The latter is the first result to quantify the usefulness of local-steps in distributed optimization.

1. Introduction

Federated learning has emerged as an important paradigm in modern large-scale machine learning. Unlike in traditional centralized learning where models are trained using large datasets stored in a central server (Dean et al., 2012; Iandola et al., 2016; Goyal et al., 2017), in federated learning, the training data remains distributed over a large number of clients, which may be phones, network sensors, hospitals, or alternative local information sources (Konečnỳ et al., 2016b;a; McMahan et al., 2017; Mohri et al., 2019; Kairouz et al., 2019). A centralized model (referred to as server model) is then trained without ever transmitting client data over the network, thereby ensuring a basic level of privacy. In this work, we investigate stochastic optimization algorithms for federated learning.

The key challenges for federated optimization are 1) dealing with unreliable and relatively slow network connections between the server and the clients, 2) only a small subset of clients being available for training at a given time, and 3) large heterogeneity (non-iid-ness) in the data present on the different clients (Konečnỳ et al., 2016a). The most popular algorithm for this setting is FEDAVG (McMahan et al., 2017). FEDAVG tackles the communication bottleneck by performing multiple local updates on the available clients before communicating to the server. While it has shown success in certain applications, its performance on heterogeneous data is still an active area of research (Li et al., 2018; Yu et al., 2019; Li et al., 2019b; Haddadpour & Mahdavi, 2019; Khaled et al., 2020). We prove that indeed such heterogeneity has a large effect on FEDAVG—it introduces a *drift* in the updates of each client resulting in slow and unstable convergence. Further, we show that this client-drift persists even if full batch gradients are used and all clients participate throughout the training.

As a solution, we propose a new Stochastic Controlled Averaging algorithm (SCAFFOLD) which tries to correct for this client-drift. Intuitively, SCAFFOLD estimates the update direction for the server model (c) and the update direction for each client c_i.[1] The difference $(c - c_i)$ is then an estimate of the client-drift which is used to correct the local update. This strategy successfully overcomes heterogeneity and converges in significantly fewer rounds of communication. Alternatively, one can see heterogeneity as introducing 'client-variance' in the updates across the different clients and SCAFFOLD then performs 'client-variance reduction' (Schmidt et al., 2017; Johnson & Zhang, 2013; Defazio et al., 2014). We use this viewpoint to show that SCAFFOLD is relatively unaffected by client sampling.

Finally, while accommodating heterogeneity is important, it is equally important that a method can take advantage of similarities in the client data. We prove that SCAFFOLD indeed has such a property, requiring fewer rounds of com-

[1]EPFL, Lausanne [2]Based on work performed at Google Research, New York. [3]Google Research, New York [4]Courant Institute, New York. Correspondence to: Sai Praneeth Karimireddy <sai.karimireddy@epfl.ch>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

[1]We refer to these estimates as *control variates* and the resulting correction technique as stochastic controlled averaging.

munication when the clients are more similar.

Contributions. We summarize our main results below.

- We derive tighter convergence rates for FEDAVG than previously known for convex and non-convex functions with client sampling and heterogeneous data.
- We give matching lower bounds to prove that even with no client sampling and full batch gradients, FEDAVG can be slower than SGD due to client-drift.
- We propose a new Stochastic Controlled Averaging algorithm (SCAFFOLD) which corrects for this client-drift. We prove that SCAFFOLD is at least as fast as SGD and converges for arbitrarily heterogeneous data.
- We show SCAFFOLD can additionally take advantage of similarity between the clients to further reduce the communication required, proving the advantage of taking local steps over large-batch SGD for the first time.
- We prove that SCAFFOLD is relatively unaffected by the client sampling obtaining variance reduced rates, making it especially suitable for federated learning.

Finally, we confirm our theoretical results on simulated and real datasets (extended MNIST by Cohen et al. (2017)).

Related work. For identical clients, FEDAVG coincides with parallel SGD analyzed by (Zinkevich et al., 2010) who proved asymptotic convergence. Stich (2018) and, more recently Stich & Karimireddy (2019); Patel & Dieuleveut (2019); Khaled et al. (2020), gave a sharper analysis of the same method, under the name of local SGD, also for identical functions. However, there still remains a gap between their upper bounds and the lower bound of Woodworth et al. (2018). The analysis of FEDAVG for heterogeneous clients is more delicate due to the afore-mentioned client-drift, first empirically observed by Zhao et al. (2018). Several analyses bound this drift by assuming bounded gradients (Wang et al., 2019; Yu et al., 2019), or view it as additional noise (Khaled et al., 2020), or assume that the client optima are ϵ-close (Li et al., 2018; Haddadpour & Mahdavi, 2019). In a concurrent work, (Liang et al., 2019) propose to use variance reduction to deal with client heterogeneity but still show rates slower than SGD and do not support client sampling. Our method SCAFFOLD can also be seen as an improved version of the distributed optimization algorithm DANE by (Shamir et al., 2014), where a fixed number of (stochastic) gradient steps are used in place of a proximal point update. A more in-depth discussion of related work is given in Appendix A. We summarize the complexities of different methods for heterogeneous clients in Table 2.

2. Setup

We formalize the problem as minimizing a sum of stochastic functions, with only access to stochastic samples:

Table 1. Summary of notation used in the paper

N, S, and i	total num., sampled num., and index of clients
R, r	number, index of communication rounds
K, k	number, index of local update steps
$\boldsymbol{x}^r$	aggregated server model after round r
$\boldsymbol{y}_{i,k}^r$	ith client's model in round r and step k
$\boldsymbol{c}^r$, $\boldsymbol{c}_i^r$	control variate of server, ith client after round r

$$\min_{\boldsymbol{x}\in\mathbb{R}^d}\Big\{f(\boldsymbol{x}) := \frac{1}{N}\sum_{i=1}^{N}(f_i(\boldsymbol{x}) := \mathbb{E}_{\zeta_i}[f_i(\boldsymbol{x};\zeta_i)])\Big\}.$$

The functions f_i represents the loss function on client i. All our results can be easily extended to the weighted case.

We assume that f is bounded from below by $f^\star$ and f_i is β-smooth. Further, we assume $g_i(\boldsymbol{x}) := \nabla f_i(\boldsymbol{x};\zeta_i)$ is an unbiased stochastic gradient of f_i with variance bounded by σ^2. For some results, we assume $\mu \geq 0$ (strong) convexity. Note that σ only bounds the variance *within* clients. We also define two non-standard terminology below.

(A1) (G, B)**-BGD** or bounded gradient dissimilarity: there exist constants $G \geq 0$ and $B \geq 1$ such that

$$\frac{1}{N}\sum_{i=1}^{N}\|\nabla f_i(\boldsymbol{x})\|^2 \leq G^2 + B^2\|\nabla f(\boldsymbol{x})\|^2\,,\ \forall \boldsymbol{x}\,.$$

If $\{f_i\}$ are convex, we can relax the assumption to

$$\frac{1}{N}\sum_{i=1}^{N}\|\nabla f_i(\boldsymbol{x})\|^2 \leq G^2 + 2\beta B^2(f(\boldsymbol{x}) - f^\star)\,,\ \forall \boldsymbol{x}\,.$$

(A2) δ**-BHD** or bounded Hessian dissimilarity:

$$\|\nabla^2 f_i(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{x})\| \leq \delta\,,\ \forall \boldsymbol{x}\,.$$

Further, f_i is δ-weakly convex i.e. $\nabla^2 f_i(\boldsymbol{x}) \succeq -\delta I$.

The assumptions A1 and A2 are orthogonal—it is possible to have $G = 0$ and $\delta = 2\beta$, or $\delta = 0$ but $G \gg 1$.

3. Convergence of FedAvg

In this section we review FEDAVG and improve its convergence analysis by deriving tighter rates than known before. The scheme consists of two main parts: local updates to the model (1), and aggregating the client updates to update the server model (2). In each round, a subset of clients $\mathcal{S} \subseteq [N]$ are sampled uniformly. Each of these clients $i \in \mathcal{S}$ copies the current sever model $\boldsymbol{y}_i = \boldsymbol{x}$ and performs K local updates of the form:

$$\boldsymbol{y}_i \leftarrow \boldsymbol{y}_i - \eta_l g_i(\boldsymbol{y}_i)\,. \tag{1}$$

Here η_l is the local step-size. Then the clients' updates $(\boldsymbol{y}_i - \boldsymbol{x})$ are aggregated to form the new server model using a global step-size η_g as:

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \frac{\eta_g}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}(\boldsymbol{y}_i - \boldsymbol{x})\,. \tag{2}$$

Table 2. Number of communication rounds required to reach ϵ accuracy for μ strongly convex and non-convex functions (log factors are ignored). Set $\mu = \epsilon$ for general convex rates. (G, B) bounds gradient dissimilarity (A1), and δ bounds Hessian dissimilarity (A2). Our rates for FEDAVG are more general and tighter than others, even matching the lower bound. However, SGD is still faster ($B \geq 1$). SCAFFOLD does not require any assumptions, is faster than SGD, and is robust to client sampling. Further, when clients become more similar (small δ), SCAFFOLD converges even faster.

Method	Strongly convex	Non-convex	Sampling	Assumptions
SGD (large batch)	$\frac{\sigma^2}{\mu NK\epsilon} + \frac{1}{\mu}$	$\frac{\sigma^2}{NK\epsilon^2} + \frac{1}{\epsilon}$	$\times$	–
FedAvg				
(Li et al., 2019b)	$\frac{\sigma^2}{\mu^2 NK\epsilon} + \frac{G^2 K}{\mu^2\epsilon}$	–	$\times$	$(G, 0)$-BGD
(Yu et al., 2019)	–	$\frac{\sigma^2}{NK\epsilon^2} + \frac{G^2 NK}{\epsilon}$	$\times$	$(G, 0)$-BGD
(Khaled et al., 2020)	$\frac{\sigma^2+G^2}{\mu NK\epsilon} + \frac{\sigma+G}{\mu\sqrt{\epsilon}} + \frac{NB^2}{\mu}$	–	$\times$	(G, B)-BGD
Ours (Thm. I)[1]	$\frac{M^2}{\mu SK\epsilon} + \frac{G}{\mu\sqrt{\epsilon}} + \frac{B^2}{\mu}$	$\frac{M^2}{SK\epsilon^2} + \frac{G}{\epsilon^{3/2}} + \frac{B^2}{\epsilon}$	✓	(G, B)-BGD
Lower-bound (Thm. II)	$\Omega\big(\frac{\sigma^2}{\mu NK\epsilon} + \frac{G}{\sqrt{\mu\epsilon}}\big)$	?	$\times$	$(G, 1)$-BGD
FedProx (Li et al., 2018)[2]	$\frac{B^2}{\mu}$	$\frac{B^2}{\epsilon}$ (weakly convex)	✓	$\sigma = 0$, $(0, B)$-BGD
DANE (Shamir et al., 2014)[2,3]	$\frac{\delta^2}{\mu^2}$	–	$\times$	$\sigma = 0$, δ-BHD
VRL-SGD (Liang et al., 2019)	–	$\frac{N\sigma^2}{K\epsilon^2} + \frac{N}{\epsilon}$	$\times$	–
SCAFFOLD				
Theorem III	$\frac{\sigma^2}{\mu SK\epsilon} + \frac{1}{\mu} + \frac{N}{S}$	$\frac{\sigma^2}{SK\epsilon^2} + \frac{1}{\epsilon}\big(\frac{N}{S}\big)^{\frac{2}{3}}$	✓	–
Theorem IV[3]	$\frac{\sigma^2}{\mu NK\epsilon} + \frac{1}{\mu K} + \frac{\delta}{\mu}$	$\frac{\sigma^2}{NK\epsilon^2} + \frac{1}{K\epsilon} + \frac{\delta}{\epsilon}$	$\times$	δ-BHD

[1] $M^2 := \sigma^2 + K(1 - \frac{S}{N})G^2$. Note that $\frac{M^2}{S} = \frac{\sigma^2}{N}$ when no sampling ($S = N$).
[2] proximal point method i.e. $K \gg 1$.
[3] proved only for quadratic functions.

3.1. Rate of convergence

We now state our novel convergence results for functions with bounded dissimilarity (proofs in Appendix D.2).

Theorem I. *For β-smooth functions $\{f_i\}$ which satisfy (A1), the output of* FEDAVG *has expected error smaller than ϵ in each of the below three cases for some values of η_l and η_g, with the following bound on R*

- μ ***Strongly convex:***

$$R = \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\mu KS\epsilon} + \left(1 - \tfrac{S}{N}\right)\frac{G^2}{\mu S\epsilon} + \frac{\sqrt{\beta}G}{\mu\sqrt{\epsilon}} + \frac{B^2\beta}{\mu}\right),$$

- ***General convex:***

$$R = \mathcal{O}\left(\frac{\sigma^2 D^2}{KS\epsilon^2} + \left(1 - \tfrac{S}{N}\right)\frac{G^2 D^2}{S\epsilon^2} + \frac{\sqrt{\beta}G}{\epsilon^{\frac{3}{2}}} + \frac{B^2\beta D^2}{\epsilon}\right),$$

- ***Non-convex:***

$$R = \mathcal{O}\left(\frac{\beta\sigma^2 F}{KS\epsilon^2} + \left(1 - \tfrac{S}{N}\right)\frac{G^2 F}{S\epsilon^2} + \frac{\sqrt{\beta}G}{\epsilon^{\frac{3}{2}}} + \frac{B^2\beta F}{\epsilon}\right),$$

where $D := \|\boldsymbol{x}^0 - \boldsymbol{x}^\star\|^2$ and $F := f(\boldsymbol{x}^0) - f^\star$.

The exact values of η_l and η_g decreases with the number of rounds R and can be found in the proofs in the Appendix. It is illuminating to compare our rates with those of the simpler iid. case i.e. with $G = 0$ and $B = 1$. Our strongly-convex rates become $\frac{\sigma^2}{\mu SK\epsilon} + \frac{1}{\mu}$. In comparison, the best previously known rate for this case was by Stich & Karimireddy (2019) who show a rate of $\frac{\sigma^2}{\mu SK\epsilon} + \frac{S}{\mu}$. The main source of improvement in the rates came from the use of *two separate step-sizes* (η_l and η_g). By having a larger global step-size η_g, we can use a smaller local step-size η_l thereby reducing the client-drift while still ensuring progress. However, even our improved rates do not match the lower-bound for the identical case of $\frac{\sigma^2}{\mu SK\epsilon} + \frac{1}{K\mu}$ (Woodworth et al., 2018). We bridge this gap for quadratic functions in Section 6.

We now compare FEDAVG to two other algorithms FedProx by (Li et al., 2018) (aka EASGD by (Zhang et al., 2015)) and to SGD. Suppose that $G = 0$ and $\sigma = 0$ i.e. we use full batch gradients and all clients have very similar optima. In such a case, FEDAVG has a complexity of $\frac{B^2}{\mu}$ which is identical to that of FedProx (Li et al., 2018). Thus, FedProx does not have any theoretical advantage.

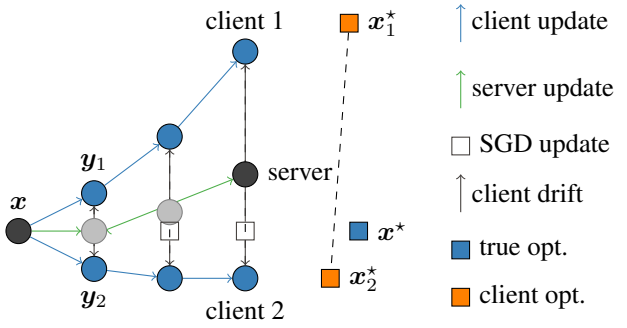

Figure 1. Client-drift in FEDAVG is illustrated for 2 clients with 3 local steps ($N = 2$, $K = 3$). The local updates $\boldsymbol{y}_i$ (in blue) move towards the individual client optima $\boldsymbol{x}_i^\star$ (orange square). The server updates (in red) move towards $\frac{1}{N}\sum_i \boldsymbol{x}_i^\star$ instead of to the true optimum $\boldsymbol{x}^\star$ (black square).

Next, suppose that all clients participate (no sampling) with $S = N$ and there is no variance $\sigma = 0$. Then, the above for strongly-convex case simplifies to $\frac{G}{\mu\sqrt{\epsilon}} + \frac{B^2}{\mu}$. In comparison, extending the proof of (Khaled et al., 2020) using our techniques gives a worse dependence on G of $\frac{G^2}{\mu KN\epsilon} + \frac{G}{\mu\sqrt{\epsilon}}$. Similarly, for the non-convex case, our rates are tighter and have better dependence on G than (Yu et al., 2019). However, simply running SGD in this setting would give a communication complexity of $\frac{\beta}{\mu}$ which is faster, and independent of similarity assumptions. In the next section we examine the necessity of such similarity assumptions.

3.2. Lower bounding the effect of heterogeneity

We now show that when the functions $\{f_i\}$ are distinct, the local updates of FEDAVG on each client experiences *drift* thereby slowing down convergence. We show that the amount of this client drift, and hence the slowdown in the rate of convergence, is exactly determined by the gradient dissimilarity parameter G in (A1).

We now examine the mechanism by which the client-drift arises (see Fig. 1). Let $\boldsymbol{x}^\star$ be the global optimum of $f(\boldsymbol{x})$ and $\boldsymbol{x}_i^\star$ be the optimum of each client's loss function $f_i(\boldsymbol{x})$. In the case of heterogeneous data, it is quite likely that each of these $\boldsymbol{x}_i^\star$ is far away from the other, and from the global optimum $\boldsymbol{x}^\star$. Even if all the clients start from the same point $\boldsymbol{x}$, each of the $\boldsymbol{y}_i$ will move towards their client optimum $\boldsymbol{x}_i^\star$. This means that the average of the client updates (which is the server update) moves towards $\frac{1}{N}\sum_{i=1}^N \boldsymbol{x}_i^\star$. This difference between $\frac{1}{N}\sum_{i=1}^N \boldsymbol{x}_i^\star$ and the true optimum $\boldsymbol{x}^\star$ is exactly the cause of client-drift. To counter this drift, FEDAVG is forced to use much smaller step-sizes which in turn hurts convergence. We can formalize this argument to prove a lower-bound (see Appendix D.4 for proof).

Theorem II. *For any positive constants G and μ, there exist μ-strongly convex functions satisfying A1 for which* FEDAVG *with $K \geq 2$, $\sigma = 0$ and $N = S$ has an error*

$$f(\boldsymbol{x}^r) - f(\boldsymbol{x}^\star) \geq \Omega\left(\frac{G^2}{\mu R^2}\right).$$

This implies that the $\frac{G}{\sqrt{\epsilon}}$ term is unavoidable even if there is no stochasticity. Further, because FEDAVG uses RKN stochastic gradients, we also have the statistical lower-bound of $\frac{\sigma^2}{\mu KN\epsilon}$. Together, these lower bounds prove that the rate derived in Theorem I is nearly optimal (up to dependence on μ). In the next section, we introduce a new method SCAFFOLD to mitigate this client-drift.

Algorithm 1 SCAFFOLD: Stochastic Controlled Averaging for federated learning

server input: initial $\boldsymbol{x}$ and $\boldsymbol{c}$, and global step-size η_g
client i's input: $\boldsymbol{c}_i$, and local step-size η_l
for each round $r = 1, \ldots, R$ **do**
  sample clients $\mathcal{S} \subseteq \{1, \ldots, N\}$
  communicate $(\boldsymbol{x}, \boldsymbol{c})$ to all clients $i \in \mathcal{S}$
  on client $i \in \mathcal{S}$ **in parallel do**
    initialize local model $\boldsymbol{y}_i \leftarrow \boldsymbol{x}$
    for $k = 1, \ldots, K$ **do**
      compute mini-batch gradient $g_i(\boldsymbol{y}_i)$
      $\boldsymbol{y}_i \leftarrow \boldsymbol{y}_i - \eta_l \left(g_i(\boldsymbol{y}_i) - \boldsymbol{c}_i + \boldsymbol{c}\right)$
    end for
    $\boldsymbol{c}_i^+ \leftarrow$ (i) $g_i(\boldsymbol{x})$, or (ii) $\boldsymbol{c}_i - \boldsymbol{c} + \frac{1}{K\eta_l}(\boldsymbol{x} - \boldsymbol{y}_i)$
    communicate $(\Delta\boldsymbol{y}_i, \Delta\boldsymbol{c}_i) \leftarrow (\boldsymbol{y}_i - \boldsymbol{x}, \boldsymbol{c}_i^+ - \boldsymbol{c}_i)$
    $\boldsymbol{c}_i \leftarrow \boldsymbol{c}_i^+$
  end on client
  $(\Delta\boldsymbol{x}, \Delta\boldsymbol{c}) \leftarrow \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}(\Delta\boldsymbol{y}_i, \Delta\boldsymbol{c}_i)$
  $\boldsymbol{x} \leftarrow \boldsymbol{x} + \eta_g \Delta\boldsymbol{x}$ and $\boldsymbol{c} \leftarrow \boldsymbol{c} + \frac{|\mathcal{S}|}{N}\Delta\boldsymbol{c}$
end for

4. SCAFFOLD algorithm

In this section we first describe SCAFFOLD and then discuss how it solves the problem of client-drift.

Method. SCAFFOLD has three main steps: local updates to the client model (3), local updates to the client control variate (4), and aggregating the updates (5). We describe each in more detail.

Along with the server model $\boldsymbol{x}$, SCAFFOLD maintains a state for each client (client control variate $\boldsymbol{c}_i$) and for the server (server control variate $\boldsymbol{c}$). These are initialized to ensure that $\boldsymbol{c} = \frac{1}{N}\sum \boldsymbol{c}_i$ and can safely all be initialized to 0. In each round of communication, the server parameters $(\boldsymbol{x}, \boldsymbol{c})$ are communicated to the participating clients $\mathcal{S} \subset [N]$. Each participating client $i \in \mathcal{S}$ initializes its local model with the server model $\boldsymbol{y}_i \leftarrow \boldsymbol{x}$. Then it makes a pass

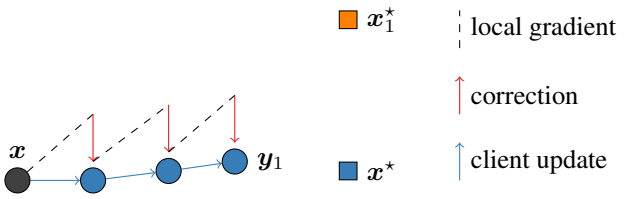

Figure 2. Update steps of SCAFFOLD on a single client. The local gradient (dashed black) points to $x_1^\star$ (orange square), but the correction term $(c - c_i)$ (in red) ensures the update moves towards the true optimum $x^\star$ (black square).

over its local data performing K updates of the form:

$$y_i \leftarrow y_i - \eta_l (g_i(y_i) + c - c_i)\,. \tag{3}$$

Then, the local control variate c_i is also updated. For this, we provide two options:

$$c_i^+ \leftarrow \begin{cases} \text{Option I.} & g_i(x)\,, \text{or} \\ \text{Option II.} & c_i - c + \frac{1}{K\eta_l}(x - y_i)\,. \end{cases} \tag{4}$$

Option I involves making an additional pass over the local data to compute the gradient at the server model x. Option II instead re-uses the previously computed gradients to update the control variate. Option I can be more stable than II depending on the application, but II is cheaper to compute and usually suffices (all our experiments use Option II). The client updates are then aggregated and used to update the server parameters:

$$\begin{aligned} x &\leftarrow x + \frac{\eta_g}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}(y_i - x)\,, \\ c &\leftarrow c + \frac{1}{N}\sum_{i\in\mathcal{S}}(c_i^+ - c_i)\,. \end{aligned} \tag{5}$$

This finishes one round of communication. Note that the clients in SCAFFOLD are *stateful* and retain the value of c_i across multiple rounds. Further, if c_i is always set to 0, then SCAFFOLD becomes equivalent to FEDAVG. The full details are summarized in Algorithm 1.

Usefulness of control variates. If communication cost was not a concern, the ideal update on client i would be

$$y_i \leftarrow y_i + \frac{1}{N}\sum_j g_j(y_i)\,. \tag{6}$$

Such an update essentially computes an unbiased gradient of f and hence becomes equivalent to running FEDAVG in the iid case (which has excellent performance). Unfortunately such an update requires communicating with all clients for every update step. SCAFFOLD instead uses control variates such that

$$c_j \approx g_j(y_i) \text{ and } c \approx \frac{1}{N}\sum_j g_j(y_i)\,.$$

Then, SCAFFOLD (3) mimics the ideal update (6) with

$$(g_i(y_i) - c_i + c) \approx \frac{1}{N}\sum_j g_j(y_i)\,.$$

Thus, the local updates of SCAFFOLD remain synchronized and converge for arbitrarily heterogeneous clients.

5. Convergence of SCAFFOLD

We state the rate of SCAFFOLD without making any assumption on the similarity between the functions. See Appendix E for the full proof.

Theorem III. *For any β-smooth functions $\{f_i\}$, the output of* SCAFFOLD *has expected error smaller than ϵ for in each of the below three cases for some values of η_l and η_g, with the following bound on R*

- μ ***Strongly convex:***

$$R = \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\mu K S\epsilon} + \frac{\beta}{\mu} + \frac{N}{S}\right),$$

- ***General convex:***

$$R = \tilde{\mathcal{O}}\left(\frac{\sigma^2 D^2}{K S\epsilon^2} + \frac{\beta D^2}{\epsilon} + \frac{NF}{S}\right),$$

- ***Non-convex:***

$$R = \mathcal{O}\left(\frac{\beta\sigma^2 F}{KS\epsilon^2} + \left(\frac{N}{S}\right)^{\frac{2}{3}}\frac{\beta F}{\epsilon}\right),$$

where $D := \|x^0 - x^\star\|^2$ and $F := f(x^0) - f^\star$.

The exact values of η_l and η_g decreases with the number of rounds R and can be found in the proofs in the Appendix. Let us first examine the rates without client sampling ($S = N$). For the strongly convex case, the number of rounds becomes $\frac{\sigma^2}{\mu NK\epsilon} + \frac{1}{\mu}$. This rate holds for arbitrarily heterogeneous clients unlike Theorem I and further matches that of SGD with K times larger batch-size, proving that SCAFFOLD is at least as fast as SGD. These rates also match known lower-bounds for distributed optimization (Arjevani & Shamir, 2015) (up to acceleration) and are unimprovable in general. However in certain cases SCAFFOLD is provably faster than SGD. We show this fact in Section 6.

Now let $\sigma = 0$. Then our rates in the strongly-convex case are $\frac{1}{\mu} + \frac{N}{S}$ and $\left(\frac{N}{S}\right)^{\frac{2}{3}}\frac{1}{\epsilon}$ in the non-convex case. These exactly match the rates of SAGA (Defazio et al., 2014; Reddi et al., 2016c). In fact, when $\sigma = 0$, $K = 1$ and $S = 1$, the update of SCAFFOLD with option I reduces to SAGA where in each round consists of sampling one client f_i. Thus SCAFFOLD can be seen as an extension of variance reduction techniques for federated learning, and one

could similarly extend SARAH (Nguyen et al., 2017), SPIDER (Fang et al., 2018), etc. Note that standard SGD with client sampling is provably slower and converges at a sublinear rate even with $\sigma = 0$.

Proof sketch. For simplicity, assume that $\sigma = 0$ and consider the ideal update of (6) which uses the full gradient $\nabla f(\boldsymbol{y})$ every step. Clearly, this would converge at a linear rate even with $S = 1$. FEDAVG would instead use an update $\nabla f_i(\boldsymbol{y})$. The difference between the ideal update (6) and the FEDAVG update (1) is $\|\nabla f_i(\boldsymbol{y}) - \nabla f(\boldsymbol{y})\|$. We need a bound on the gradient-dissimilarity as in (A1) to bound this error. SCAFFOLD instead uses the update $\nabla f_i(\boldsymbol{y}) - \boldsymbol{c}_i + \boldsymbol{c}$, and the difference from ideal update becomes

$$\sum_i \|(\nabla f_i(\boldsymbol{y}) - \boldsymbol{c}_i + \boldsymbol{c}) - \nabla f(\boldsymbol{y})\|^2 \leq \sum_i \|\boldsymbol{c}_i - \nabla f_i(\boldsymbol{y})\|^2 .$$

Thus, the error is independent of how similar or dissimilar the functions f_i are, and instead only depends on the quality of our approximation $\boldsymbol{c}_i \approx \nabla f_i(\boldsymbol{y})$. Since f_i is smooth, we can expect that the gradient $\nabla f_i(\boldsymbol{y})$ does not change too fast and hence is easy to approximate. Appendix E translates this intuition into a formal proof.

6. Usefulness of local steps

In this section we investigate when and why taking local steps might be useful over simply computing a large-batch gradient in distributed optimization. We will show that when the functions across the clients share some similarity, local steps can take advantage of this and converge faster. For this we consider quadratic functions and express their similarity with the δ parameter introduced in (A2).

Theorem IV. *For any β-smooth quadratic functions $\{f_i\}$ with δ bounded Hessian dissimilarity (A2), the output of* SCAFFOLD *with $S = N$ (no sampling) has error smaller than ϵ in each of the following two cases with $\eta_g = 1$, some value of η_l, and R satisfying*

- ***Strongly convex:***

$$R = \tilde{\mathcal{O}}\left(\frac{\beta\sigma^2}{\mu K N \epsilon} + \frac{\beta + \delta K}{\mu K}\right),$$

- ***Weakly convex:***

$$R = \mathcal{O}\left(\frac{\beta\sigma^2 F}{K N \epsilon^2} + \frac{(\beta + \delta K)F}{K\epsilon}\right),$$

where we define $F := (f(\boldsymbol{x}^0) - f^\star)$.

Here again the exact value of η_l decreases with the number of rounds R and can be found in the proofs in the Appendix. When $\sigma = 0$ and K is large, the complexity of SCAFFOLD becomes $\frac{\delta}{\mu}$. In contrast DANE, which being a proximal point method also uses large K, requires $(\frac{\delta}{\mu})^2$ rounds (Shamir et al., 2014) which is significantly slower, or needs an additional backtracking-line search to match the rates of SCAFFOLD (Yuan & Li, 2019). Further, Theorem IV is the first result to demonstrate improvement due to similairty for non-convex functions as far as we are aware.

Suppose that $\{f_i\}$ are identical. Recall that δ in (A2) measures the Hessian dissimilarity between functions and so $\delta = 0$ for this case. Then Theorem IV shows that the complexity of SCAFFOLD is $\frac{\sigma^2}{\mu K N \epsilon} + \frac{1}{\mu K}$ which (up to acceleration) matches the i.i.d. lower bound of (Woodworth et al., 2018). In contrast, SGD with K times larger batch-size would require $\frac{\sigma^2}{\mu K N \epsilon} + \frac{1}{\mu}$ (note the absence of K in the second term). Thus, for identical functions, SCAFFOLD (and in fact even FEDAVG) improves linearly with increasing number of local steps. In the other extreme, if the functions are arbitrarily different, we may have $\delta = 2\beta$. In this case, the complexity of SCAFFOLD and large-batch SGD match the lower bound of Arjevani & Shamir (2015) for the heterogeneous case.

The above insights can be generalized to when the functions are only somewhat similar. If the Hessians are δ-close and $\sigma = 0$, then the complexity is $\frac{\beta + \delta K}{\mu K}$. This bound implies that the optimum number of local steps one should use is $K = \frac{\beta}{\delta}$. Picking a smaller K increases the communication required whereas increasing it further would only waste computational resources. While this result is intuitive—if the functions are more 'similar', local steps are more useful—Theorem IV shows that it is the similarity of the *Hessians* which matters. This is surprising since the Hessians of $\{f_i\}$ may be identical even if their individual optima $\boldsymbol{x}_i^\star$ are arbitrarily far away from each other and the gradient-dissimilarity (A1) is unbounded.

Proof sketch. Consider a simplified SCAFFOLD update with $\sigma = 0$ and no sampling ($S = N$):

$$\boldsymbol{y}_i = \boldsymbol{y}_i - \eta(\nabla f_i(\boldsymbol{y}_i) + \nabla f(\boldsymbol{x}) - \nabla f_i(\boldsymbol{x})) .$$

We would ideally want to perform the update $\boldsymbol{y}_i = \boldsymbol{y}_i - \eta\nabla f(\boldsymbol{y}_i)$ using the full gradient $\nabla f(\boldsymbol{y}_i)$. We reinterpret the correction term of SCAFFOLD $(\boldsymbol{c} - \boldsymbol{c}_i)$ as performing the following first order correction to the local gradient $\nabla f_i(\boldsymbol{y}_i)$ to make it closer to the full gradient $\nabla f(\boldsymbol{y}_i)$:

$$\begin{aligned}
&\underbrace{\nabla f_i(\boldsymbol{y}_i) - \nabla f_i(\boldsymbol{x})}_{\approx \nabla^2 f_i(\boldsymbol{x})(\boldsymbol{y}_i - \boldsymbol{x})} + \underbrace{\nabla f(\boldsymbol{x})}_{\approx \nabla f(\boldsymbol{y}_i) + \nabla^2 f(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{y}_i)} \\
&\qquad \approx \nabla f(\boldsymbol{y}_i) + (\nabla^2 f_i(\boldsymbol{x}) - \nabla^2 f(\boldsymbol{x}))(\boldsymbol{y}_i - \boldsymbol{x}) \\
&\qquad \approx \nabla f(\boldsymbol{y}_i) + \delta(\boldsymbol{y}_i - \boldsymbol{x})
\end{aligned}$$

Thus the SCAFFOLD update approximates the ideal update up to an error δ. This intuition is proved formally for

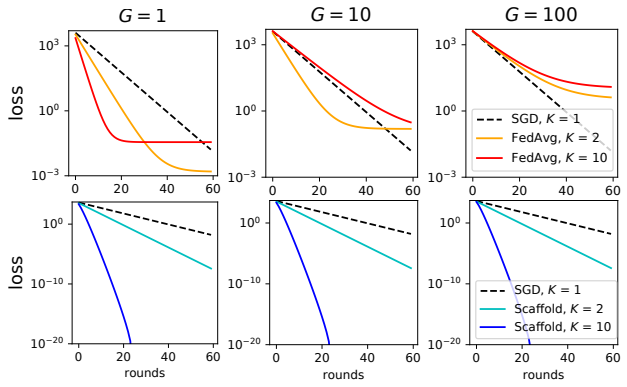

Figure 3. SGD (dashed black), FedAvg (above), and SCAFFOLD (below) on simulated data. FedAvg gets worse as local steps increases with $K = 10$ (red) worse than $K = 2$ (orange). It also gets slower as gradient-dissimilarity (G) increases (to the right). SCAFFOLD significantly improves with more local steps, with $K = 10$ (blue) faster than $K = 2$ (light blue) and SGD. Its performance is identical as we vary heterogeneity (G).

quadratic functions in Appendix F. Generalizing these results to other functions is a challenging open problem.

7. Experiments

We run experiments on both simulated and real datasets to confirm our theory. Our main findings are i) SCAFFOLD consistently outperforms SGD and FEDAVG across all parameter regimes, and ii) the benefit (or harm) of local steps depends on both the algorithm and the similarity of the clients data.

7.1. Setup

Our simulated experiments uses $N = 2$ quadratic functions based on our lower-bounds in Theorem II. We use full-batch gradients ($\sigma = 0$) and no client sampling. Our real world experiments run logistic regression (convex) and 2 layer fully connected network (non-convex) on the EMNIST (Cohen et al., 2017). We divide this dataset among $N = 100$ clients as follows: for $s\%$ similar data we allocate to each client $s\%$ i.i.d. data and the remaining $(100 - s)\%$ by sorting according to label (cf. Hsu et al. (2019)).

We consider four algorithms: SGD, FEDAVG SCAFFOLD and FEDPROX with SGD as the local solver (Li et al., 2018). On each client SGD uses the full local data to compute a single update, whereas the other algorithms take 5 steps per epoch (batch size is 0.2 of local data). We always use global step-size $\eta_g = 1$ and tune the local step-size η_l individually for each algorithm. SCAFFOLD uses option II (no extra gradient computations) and FEDPROX has fixed regularization $= 1$ to keep comparison fair. Additional tuning of the regularization parameter may sometimes yield improved empirical performance.

7.2. Simulated results

The results are summarized in Fig. 3. Our simulated data has Hessian difference $\delta = 1$ (A2) and $\beta = 1$. We vary the gradient heterogeneity (A1) as $G \in [1, 10, 100]$. For all valued of G, FEDAVG gets slower as we increase the number of local steps. This is explained by the fact that client-drift increases as we increase the number of local steps, hindering progress. Further, as we increase G, FEDAVG continues to slow down exactly as dictated by Thms. I and II. Note that when heterogeneity is small ($G = \beta = 1$), FEDAVG can be competitive with SGD.

SCAFFOLD is consistently faster than SGD, with $K = 2$ being twice as fast and $K = 10$ about 5 times faster. Further, its convergence is completely unaffected by G, confirming our theory in Thm. III. The former observation that we do not see linear improvement with K is explained by Thm. IV since we have $\delta > 0$. This sub linear improvement is still significantly faster than both SGD and FEDAVG.

7.3. EMNIST results

We run extensive experiments on the EMNIST dataset to measure the interplay between the algorithm, number of epochs (local updates), number of participating clients, and the client similarity. Table 3 measures the benefit (or harm) of using more local steps, Table 4 studies the resilience to client sampling, and Table 5 reports preliminary results on neural networks. We are mainly concerned with minimizing the number of *communication rounds*. We observe that

SCAFFOLD is consistently the best. Across all range of values tried, we observe that SCAFFOLD outperforms SGD, FEDAVG, and FEDPROX. The latter FEDPROX is always slower than the other local update methods, though in some cases it outperforms SGD. Note that it is possible to improve FEDPROX by carefully tuning the regularization parameter (Li et al., 2018). FEDAVG is always slower than SCAFFOLD and faster than FEDPROX.

SCAFFOLD $>$ SGD $>$ FedAvg for heterogeneous clients. When similarity is 0%, FEDAVG gets slower with increasing local steps. If we take more than 5 epochs, its performance is worse than SGD's. SCAFFOLD initially worsens as we increase the number of epochs but then flattens. However, its performance is always better than that of SGD, confirming that it can handle heterogeneous data.

SCAFFOLD and FedAvg get faster with more similarity, but not SGD. As similarity of the clients increases, the performance of SGD remains relatively constant. On the other hand, SCAFFOLD and FEDAVG get significantly faster as similarity increases. Further, local steps become much more useful, showing monotonic improvement with the increase in number of epochs. This is because with increasing the i.i.d.ness of the data, both the gradient and

Table 3. Communication rounds to reach 0.5 test accuracy for logistic regression on EMNIST as we vary number of epochs. 1k+ indicates 0.5 accuracy was not reached even after 1k rounds, and similarly an arrowhead indicates that the barplot extends beyond the table. 1 epoch for local update methods corresponds to 5 local steps (0.2 batch size), and 20% of clients are sampled each round. We fix $\mu = 1$ for FEDPROX and use variant (ii) for SCAFFOLD to ensure all methods are comparable. Across all parameters (epochs and similarity), SCAFFOLD is the fastest method. When similarity is 0 (sorted data), FEDAVG consistently gets worse as we increase the number of epochs, quickly becoming slower than SGD. SCAFFOLD initially gets worse and later stabilizes, but is always at least as fast as SGD. As similarity increases (i.e. data is more shuffled), both FEDAVG and SCAFFOLD significantly outperform SGD though SCAFFOLD is still better than FEDAVG. Further, with higher similarity, both methods benefit from increasing number of epochs.

	Epochs	0% similarity (sorted)		10% similarity		100% similarity (i.i.d.)	
		Num. of rounds	Speedup	Num. of rounds	Speedup	Num. of rounds	Speedup
SGD	1	317	(1×)	365	(1×)	416	(1×)
SCAFFOLD	1	**77**	(4.1×)	62	(5.9×)	60	(6.9×)
	5	152	(2.1×)	20	(18.2×)	10	(41.6×)
	10	286	(1.1×)	16	(22.8×)	7	(59.4×)
	20	266	(1.2×)	**11**	(33.2×)	**4**	(104×)
FEDAVG	1	258	(1.2×)	74	(4.9×)	83	(5×)
	5	428	(0.7×)	34	(10.7×)	10	(41.6×)
	10	711	(0.4×)	25	(14.6×)	6	(69.3×)
	20	1k+	(< 0.3×)	18	(20.3×)	**4**	(104×)
FEDPROX	1	1k+	(< 0.3×)	979	(0.4×)	459	(0.9×)
	5	1k+	(< 0.3×)	794	(0.5×)	351	(1.2×)
	10	1k+	(< 0.3×)	894	(0.4×)	308	(1.4×)
	20	1k+	(< 0.3×)	916	(0.4×)	351	(1.2×)

Table 4. Communication rounds to reach 0.45 test accuracy for logistic regression on EMNIST as we vary the number of sampled clients. Number of epochs is kept fixed to 5. SCAFFOLD is consistently faster than FEDAVG. As we decrease the number of clients sampled in each round, the increase in number of rounds is sub-linear. This slow-down is better for more similar clients.

	Clients	0% similarity		10% similarity	
SCAFFOLD	20%	143	(1.0×)	9	(1.0×)
	5%	290	(2.0×)	13	(1.4×)
	1%	790	(5.5×)	28	(3.1×)
FEDAVG	20%	179	(1.0×)	12	(1.0×)
	5%	334	(1.9×)	17	(1.4×)
	1%	1k+	(5.6+×)	35	(2.9×)

Table 5. Best test accuracy after 1k rounds with 2-layer fully connected neural network (non-convex) on EMNIST trained with 5 epochs per round (25 steps) for the local methods, and 20% of clients sampled each round. SCAFFOLD has the best accuracy and SGD has the least. SCAFFOLD again outperforms other methods. SGD is unaffected by similarity, whereas the local methods improve with client similarity.

	0% similarity	10% similarity
SGD	0.766	0.764
FEDAVG	0.787	0.828
SCAFFOLD	**0.801**	**0.842**

Hessian dissimilarity decrease.

SCAFFOLD is resilient to client sampling. As we decrease the fraction of clients sampled, SCAFFOLD ,and FEDAVG only show a sub-linear slow-down. They are more resilient to sampling in the case of higher similarity.

SCAFFOLD outperforms FedAvg on non-convex experiments. We see that SCAFFOLD is better than FEDAVG in terms of final test accuracy reached, though interestingly FEDAVG seems better than SGD even when similarity is 0. However, much more extensive experiments (beyond current scope) are needed before drawing conclusions.

8. Conclusion

Our work studied the impact of heterogeneity on the performance of optimization methods for federated learning. Our careful theoretical analysis showed that FEDAVG can be severely hampered by *gradient dissimilarity*, and can be even slower than SGD. We then proposed a new stochastic algorithm (SCAFFOLD) which overcomes gradient dis-

similarity using control variates. We demonstrated the effectiveness of SCAFFOLD via strong convergence guarantees and empirical evaluations. Further, we showed that while SCAFFOLD is always at least as fast as SGD, it can be much faster depending on the *Hessian dissimilarity* in our data. Thus, different algorithms can take advantage of (and are limited by) different notions of dissimilarity. We believe that characterizing and isolating various dissimilarities present in real world data can lead to further new algorithms and significant impact on distributed, federated, and decentralized learning.

Acknowledgments. We thank Filip Hanzely and Jakub Konečnỳ for discussions regarding variance reduction techniques and Blake Woodworth, Virginia Smith and Kumar Kshitij Patel for suggestions which improved the writing.

References

Agarwal, N., Suresh, A. T., Yu, F. X., Kumar, S., and McMahan, B. cpSGD: Communication-efficient and differentially-private distributed SGD. In *Proceedings of NeurIPS*, pp. 7575–7586, 2018.

Arjevani, Y. and Shamir, O. Communication complexity of distributed convex learning and optimization. In *Advances in neural information processing systems*, pp. 1756–1764, 2015.

Bassily, R., Smith, A., and Thakurta, A. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pp. 464–473. IEEE, 2014.

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*, 2019.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191. ACM, 2017.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., and Shi, W. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

Cen, S., Zhang, H., Chi, Y., Chen, W., and Liu, T.-Y. Convergence of distributed stochastic variance reduced methods without sampling extra data. *arXiv preprint arXiv:1905.12648*, 2019.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

Chen, M., Mathews, R., Ouyang, T., and Beaufays, F. Federated learning of out-of-vocabulary words. *arXiv preprint arXiv:1903.10635*, 2019a.

Chen, M., Suresh, A. T., Mathews, R., Wong, A., Beaufays, F., Allauzen, C., and Riley, M. Federated learning of N-gram language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 2019b.

Cohen, G., Afshar, S., Tapson, J., and Van Schaik, A. Emnist: Extending mnist to handwritten letters. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., Le, Q. V., and Ng, A. Y. Large scale distributed deep networks. In *Advances in neural information processing systems*, pp. 1223–1231, 2012.

Defazio, A. and Bottou, L. On the ineffectiveness of variance reduced optimization for deep learning. In *Advances in Neural Information Processing Systems*, pp. 1753–1763, 2019.

Defazio, A., Bach, F., and Lacoste-Julien, S. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pp. 1646–1654, 2014.

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.

Glasserman, P. *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media, 2013.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch SGD: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

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

Hanzely, F. and Richtárik, P. One method to rule them all: Variance reduction for data, parameters and many new methods. *arXiv preprint arXiv:1905.11266*, 2019.

Hard, A., Rao, K., Mathews, R., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., and Ramage, D. Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*, 2018.

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

Iandola, F. N., Moskewicz, M. W., Ashraf, K., and Keutzer, K. Firecaffe: near-linear acceleration of deep neural network training on compute clusters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2592–2600, 2016.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

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.

Karimi, H., Nutini, J., and Schmidt, M. 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.

Karimireddy, S. P., Rebjock, Q., Stich, S. U., and Jaggi, M. Error feedback fixes SignSGD and other gradient compression schemes. *arXiv preprint arXiv:1901.09847*, 2019.

Khaled, A., Mishchenko, K., and Richtárik, P. Tighter theory for local SGD on indentical and heterogeneous data. In *Proceedings of AISTATS*, 2020.

Kifer, D., Smith, A., and Thakurta, A. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pp. 25–1, 2012.

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

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*, 2016b.

Kulunchakov, A. and Mairal, J. Estimate sequences for stochastic composite optimization: Variance reduction, acceleration, and robustness to noise. *arXiv preprint arXiv:1901.08788*, 2019.

Lee, J. D., Lin, Q., Ma, T., and Yang, T. Distributed stochastic variance reduced gradient methods and a lower bound for communication complexity. *arXiv preprint arXiv:1507.07595*, 2015.

Lei, L. and Jordan, M. Less than a single pass: Stochastically controlled stochastic gradient. In *AISTATS*, pp. 148–156, 2017.

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

Li, T., Sanjabi, M., and Smith, V. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019a.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. Feddane: A federated newton-type method. *arXiv preprint arXiv:2001.01920*, 2020.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. On the convergence of FedAvg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019b.

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.

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

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

Mohri, M., Sivek, G., and Suresh, A. T. Agnostic federated learning. *arXiv preprint arXiv:1902.00146*, 2019.

Nedich, A., Olshevsky, A., and Shi, W. A geometrically convergent method for distributed optimization over time-varying graphs. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 1023–1029. IEEE, 2016.

Nesterov, Y. *Lectures on convex optimization*, volume 137. Springer, 2018.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2613–2621. JMLR. org, 2017.

Nguyen, L. M., Scheinberg, K., and Takáč, M. Inexact SARAH algorithm for stochastic optimization. *arXiv preprint arXiv:1811.10105*, 2018.

Patel, K. K. and Dieuleveut, A. Communication trade-offs for synchronized distributed SGD with large step size. *arXiv preprint arXiv:1904.11325*, 2019.

Ramaswamy, S., Mathews, R., Rao, K., and Beaufays, F. Federated learning for emoji prediction in a mobile keyboard. *arXiv preprint arXiv:1906.04329*, 2019.

Reddi, S. J., Hefny, A., Sra, S., Poczos, B., and Smola, A. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323, 2016a.

Reddi, S. J., Konečnỳ, J., Richtárik, P., Póczós, B., and Smola, A. Aide: Fast and communication efficient distributed optimization. *arXiv preprint arXiv:1608.06879*, 2016b.

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. Fast incremental method for smooth nonconvex optimization. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 1971–1977. IEEE, 2016c.

Safran, I. and Shamir, O. How good is sgd with random shuffling? *arXiv preprint arXiv:1908.00045*, 2019.

Samarakoon, S., Bennis, M., Saad, W., and Debbah, M. Federated learning for ultra-reliable low-latency v2v communications. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7. IEEE, 2018.

Schmidt, M., Le Roux, N., and Bach, F. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

Shamir, O., Srebro, N., and Zhang, T. Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pp. 1000–1008, 2014.

Shi, W., Ling, Q., Wu, G., and Yin, W. EXTRA: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

Smith, V., Forte, S., Ma, C., Takáč, M., Jordan, M., and Jaggi, M. CoCoA: A general framework for communication-efficient distributed optimization. *Journal of Machine Learning Research*, 18(230):1–49, 2018.

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

Stich, S. U. Unified optimal analysis of the (stochastic) gradient method. *arXiv preprint arXiv:1907.04232*, 2019.

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.

Suresh, A. T., Yu, F. X., Kumar, S., and McMahan, H. B. Distributed mean estimation with limited communication. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3329–3337. JMLR. org, 2017.

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.

Vaswani, S., Bach, F., and Schmidt, M. Fast and faster convergence of sgd for over-parameterized models and an accelerated perceptron. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1195–1204, 2019.

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, 2019.

Woodworth, B. E., Wang, J., Smith, A., McMahan, H. B., and Srebro, N. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. In *Advances in neural information processing systems*, pp. 8496–8506, 2018.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., and Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *arXiv preprint arXiv:1812.02903*, 2018.

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, 2019.

Yuan, X.-T. and Li, P. On convergence of distributed approximate newton methods: Globalization, sharper bounds and beyond. *arXiv preprint arXiv:1908.02246*, 2019.

Zhang, L., Mahdavi, M., and Jin, R. Linear convergence with condition number independent access of full gradients. In *Advances in Neural Information Processing Systems*, pp. 980–988, 2013a.

Zhang, S., Choromanska, A. E., and LeCun, Y. Deep learning with elastic averaging sgd. In *Advances in neural information processing systems*, pp. 685–693, 2015.

Zhang, Y., Duchi, J. C., and Wainwright, M. J. Communication-efficient algorithms for statistical optimization. *The Journal of Machine Learning Research*, 14(1):3321–3363, 2013b.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

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