Faster Adaptive Federated Learning

Xidong Wu[1], **Feihu Huang**[1,2], **Zhengmian Hu**[1], **Heng Huang**[1]

[1] Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA, United States
[2] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
xidong_wu@outlook.com, huangfeihu2018@gmail.com, huzhengmian@gmail.com, henghuanghh@gmail.com

Abstract

Federated learning has attracted increasing attention with the emergence of distributed data. While extensive federated learning algorithms have been proposed for the non-convex distributed problem, federated learning in practice still faces numerous challenges, such as the large training iterations to converge since the sizes of models and datasets keep increasing, and the lack of adaptivity by SGD-based model updates. Meanwhile, the study of adaptive methods in federated learning is scarce and existing works either lack a complete theoretical convergence guarantee or have slow sample complexity. In this paper, we propose an efficient adaptive algorithm (i.e., FAFED) based on the momentum-based variance-reduced technique in cross-silo FL. We first explore how to design the adaptive algorithm in the FL setting. By providing a counter-example, we prove that a simple combination of FL and adaptive methods could lead to divergence. More importantly, we provide a convergence analysis for our method and prove that our algorithm is the first adaptive FL algorithm to reach the best-known samples $O(\epsilon^{-3})$ and $O(\epsilon^{-2})$ communication rounds to find an ϵ-stationary point without large batches. The experimental results on the language modeling task and image classification task with heterogeneous data demonstrate the efficiency of our algorithms.

Introduction

Distributed training, which emerges to address the challenge of distributed data, has attracted wide attention (Bao et al. 2022). With the improvement of computing power, the bottleneck of training speed is gradually shifting from computing capacity to communication. Therefore, federated learning (FL) (McMahan et al. 2017) was proposed as an important distributed training paradigm in large-scale machine learning to reduce communication overhead. In the FL setting, a central server coordinates multiple worker nodes to learn a joint model together with periodic model averaging by leveraging the massive local data of each worker node. The worker nodes share the computational load, and FL also provides some level of data privacy because training data are not directly shared or aggregated.

More recently, an increasing number of FL works focus on addressing the cross-silo FL (i.e., FL between large institutions) problem (Xu and Huang 2022; Guo et al. 2022; Karimireddy et al. 2020b), where most clients participate in computation every round and can maintain state between rounds. The cross-silo FL involves many practical applications, such as collaborative learning on financial data across various corporations and stakeholders or health data across numerous medical centers (Xu et al. 2022; Guo et al. 2022).

In this paper, we consider solving a federated learning problem in the cross-silo setting, defined as

$$\min_{x\in\mathbb{R}^d} f(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x) \tag{1}$$

where $x \in \mathbb{R}^d$ denotes the model parameter and N indicates the number of worker nodes. $f_i(x) = \mathbb{E}_{\xi^{(i)}\sim\mathcal{D}_i}\left[f_i\left(x;\xi^{(i)}\right)\right]$ is the loss function of the i^{th} worker node, and $\xi^{(i)} \sim \mathcal{D}_i$ denotes the samples $\xi^{(i)}$ drawn from distribution $\mathcal{D}_i$ on the i^{th} worker node. When $\mathcal{D}_i$ and $\mathcal{D}_j$ are different ($i \neq j$), it is referred to as the heterogeneous data setting. In this paper, $\{\mathcal{D}_i\}_{i=1}^N$ are not identical. We restrict our focus to the non-convex problem, where the functions $f_i(x)$ and $f(x)$, therefore, are smooth and non-convex. On worker node i, we have access to a stochastic gradient $\nabla f_i(x;\xi^{(i)})$ as an unbiased estimation of the i^{th} worker node's true gradient $\nabla f_i(x)$. Worker nodes collaboratively learn a global model, but the raw data in each worker node is never shared with the server and other worker nodes.

Although, various FL methods have been proposed (Karimireddy et al. 2020b; Reddi et al. 2020; Hong et al. 2021; Xiong, Li, and Cai 2023), which substantially reduce communication cost by avoiding frequent transmission between local worker nodes and the central server, it suffers from unfavorable convergence behavior. It is caused by a variety of factors, such as (1) client drift (Karimireddy et al. 2020b), where local client models move towards local optima instead of global optima, (2) lack of adaptivity as SGD-based update (Reddi et al. 2020), and (3) large training iterations to converge as sizes of model parameters and training datasets keep increasing. Despite the recent advances, most of the existing work focuses on solving client drifts (Karimireddy et al. 2020b; Khanduri et al. 2021; Xu and Huang 2022). The current federated learning framework still cannot solve all challenges.

On the other hand, we know that adaptive methods have been widely developed and studied in non-federated settings since they generally require less parameter tuning and better convergence speed during the training phase. Meanwhile, like centralized method SGD, stochastic FL methods are not a good option for settings with heavy-tail stochastic gradient noise distributions. Such issues could be solved by adaptive learning rates (Zhang et al. 2019), which combines knowledge of past iterations. In addition, adaptive gradient methods also escape saddle points faster compared to SGD (Staib et al. 2019). Therefore, the introduction of adaptive tools is an important direction to improve the performance of FL algorithms in the practice.

However, the design of adaptive FL methods is nontrivial because the local worker node moves towards different directions and the global trackers cannot be updated frequently in the FL setting. The improper design of the adaptive FL method might lead to convergence issues (Chen, Li, and Li 2020). Reddi et al. (2020) firstly proposed a class of federated versions of adaptive optimizers, including FedAdagrad, FedYogi, and FedAdam. But its analysis only holds when the $\beta_1 = 0$ and it cannot use the advantage of the momentum. MimeAdam is proposed (Karimireddy et al. 2020a) and it applies server statistics locally to address this issue. Nevertheless, MimeAdam has to compute the full local gradient, which might be forbidden in practice. More recently, FedAMS is proposed (Wang, Lin, and Chen 2022) and it provides the completed proof, but it doesn't improve the convergence rate. Overall, the sample convergence rates of FedAdagrad, FedYogi, FedAdam and FedAMS are $O\left(\epsilon^{-4}\right)$ (Not better than FedAvg). At the same time, they also require an extra global learning rate to tune.

As the sizes of model parameters and training datasets keep increasing, the deep learning models require more training iterations to converge and more efficient optimization methods are welcomed. Consequently, a natural question is whether one can achieve a faster convergence rate in theory and practice with adaptive technology. In this paper, we give an affirmative answer to the above question by proposing a faster adaptive FL algorithm (i.e., FAFED).

Contributions The main contributions of this work are listed below:

- We study how to incorporate the adaptive gradient method into federated learning. We propose a faster stochastic adaptive FL method (i.e., FAFED) in heterogeneous data settings based on the momentum-based variance reduction technique with a general adaptive matrix.
- We provide a convergence analysis framework for our adaptive methods under some mild assumptions. Our algorithm is the first adaptive FL algorithm to reach the best-known samples complexity $O(\epsilon^{-3})$ and communication complexity $O(\epsilon^{-2})$ to find an ϵ-stationary point without large batches. The extensive experimental results on the language modeling task and image classification task confirm the effectiveness of our proposed algorithm.
- By establishing a counter example, we also show that a naive combination of adaptive gradient methods and periodic model averaging might result in divergence. Therefore, sharing adaptive learning should be considered in the FL setting.

Related Works

Federated Learning

FedAvg was proposed in (McMahan et al. 2017) as the first FL algorithm. With periodic model averaging, it can dramatically reduce communication overheads. Earlier works analyzed FL algorithms in the homogeneous data setting (Woodworth et al. 2020; Khaled, Mishchenko, and Richtárik 2020) and recent research extends federated learning to heterogeneous data settings (non-iid), as well as non-convex models, such as deep neural networks. When datasets on different worker nodes are homogeneous, FedAvg reduces to local SGD (Zinkevich et al. 2010).

Recent works (Yang, Fang, and Liu 2021; Karimireddy et al. 2020b) consider FedAvg with partial worker nodes participation with $O(1)$ local updates iterations and batch sizes. The sample and communication complexities are both $O(\epsilon^{-4})$. In (Yu, Jin, and Yang 2019; Yu, Yang, and Zhu 2019), authors propose Parallel Restarted SGD and Momentum SGD, and show that both of them require $O(\varepsilon^{-4})$ samples and $O(\varepsilon^{-3})$ rounds of communication to reach an ε-stationary solution. SCAFFOLD was proposed in (Karimireddy et al. 2020b), which uses control variates to correct for the 'client-drift' when the data is heterogeneous. It achieves the same sample and communication complexities as FedAvg. Li et al. (2020) proposed a penalty-based method called FedProx to reduce the communication complexity to $O(\varepsilon^{-2})$. The analysis of FedProx depends on a gradient similarity assumption to restrict the data heterogeneity, which essentially requires that all minimums of $f(x)$ are also minimums of $f_i(x)$. Later, FedPD was proposed in (Zhang et al. 2020) to relax this assumption.

Momentum-based optimizers are widely used in learning tasks (Sun et al. 2022). Subsequently, some momentum-based FL algorithms are proposed. For example, (Xu and Huang 2022) introduces a momentum fusion technique to coordinate the server and local momentum buffers, but they do not reduce the complexity. Based on variance reduction technology, Fed-GLOMO (Das et al. 2022) require $O(\varepsilon^{-3})$ sample complexity and $O(\varepsilon^{-3})$ communication complexity. Their sample complexity matches the optimal complexity of the centralized non-convex stochastic optimization algorithms (Fang et al. 2018; Cutkosky and Orabona 2019). More recently, STEM was proposed in (Khanduri et al. 2021) which utilizes a momentum-assisted stochastic gradient direction for both the worker nodes and central server updates. It further reduces the communication rounds to $O(\varepsilon^{-2})$ and keeps the same sample cost of $O(\varepsilon^{-3})$.

Adaptive Methods

Adaptive methods are a class of optimization algorithms as one of the most important variants of stochastic gradient descent in machine learning. For example, Adam (Kingma and Ba 2014) AdaGrad (Duchi, Hazan, and Singer 2011), AdaDelta (Zeiler 2012) are widely used as optimization tools in training deep neural networks (DNNs). Afterward, some variants (Reddi, Kale, and Kumar 2019) have been proposed to show a convergence guarantee in the non-convex setting. More recently, the works (Cutkosky

Algorithm	Reference	Sample	Communication	Adaptivity
FedAvg	(Yang, Fang, and Liu 2021) (Karimireddy et al. 2020b)	$O\left(\epsilon^{-4}\right)$	$O\left(\epsilon^{-4}\right)$	
FedAdagrad	(Reddi et al. 2020)	$O\left(\epsilon^{-4}\right)$	$O\left(\epsilon^{-4}\right)$	$\surd$
FedYogi	(Reddi et al. 2020)	$O\left(\epsilon^{-4}\right)$	$O\left(\epsilon^{-4}\right)$	$\surd$
FedAdam	(Reddi et al. 2020)	$O\left(\epsilon^{-4}\right)$	$O\left(\epsilon^{-4}\right)$	$\surd$
FedAMS	(Wang, Lin, and Chen 2022)	$O\left(\epsilon^{-4}\right)$	$O\left(\epsilon^{-4}\right)$	$\surd$
FAFED	Our work	$\tilde{O}\left(\epsilon^{-3}\right)$	$\tilde{O}\left(\epsilon^{-2}\right)$	$\surd$

Table 1: Complexity comparison of FedAvg and typical adaptive FL algorithms for finding an ϵ-stationary point. Sample complexity denotes the number of calls to the First-order Oracle (IFO) by all worker nodes to reach an ε-stationary point. Communication complexity is defined as the total number of back-and-forth communication rounds between each worker node and the central server required to reach an ε-stationary point.

and Orabona 2019; Huang, Li, and Huang 2021) presented some accelerated adaptive gradient methods based on the variance-reduced techniques.

In FL settings, Reddi et al. (2020) firstly propose federated versions of adaptive optimizers, including a class of adaptive FL methods, such as FedAdagrad, FedYogi, and FedAdam. These methods achieve the same sample cost and communication rounds as FedAvg when assuming the $\beta_1 = 0$. Chen, Li, and Li (2020) proposed Federated AMSGrad and achieves the same sample cost and communication rounds. MimeAdam is proposed in (Karimireddy et al. 2020a) but it requires the full local gradient in each communication round. More recently, FedAMS is proposed in (Wang, Lin, and Chen 2022) and it provides the completed proof and considers the gradient compression. But it doesn't improve the convergence rate. Table 1 summarizes the details of typical adaptive FL algorithms.

Preliminaries

Notations: For two vectors x and y in Euclidean space, $\langle x, y\rangle$ denote their inner product. $\|\cdot\|$ denotes the ℓ_2 norm for vectors and spectral norm for matrices, respectively. And $x_{t,i}$ denotes the local model parameters of the i^{th} worker node at the iteration t. $\nabla_x f(x)$ is the partial derivative w.r.t. variables x. I_d means d-dimension identity matrix. $a = O(b)$ denotes that $a \leq Cb$ for some constant $C > 0$, and the notation $\tilde{O}(\cdot)$ hides logarithmic terms. Given the mini-batch samples $\mathcal{B} = \{\xi_i\}_{i=1}^q$, we let $\nabla f_i(x;\mathcal{B}) = \frac{1}{q}\sum_{i=1}^q \nabla f_i(x;\xi_i)$.

Assumption 1. *(i) Unbiased Gradient. Each component function $f_i(x;\xi)$ computed at each worker node is unbiased $\forall \xi^{(i)} \sim \mathcal{D}_i$, $i \in [N]$ and $x \in \mathbb{R}^d$:*

$$\mathbb{E}[\nabla f_i(x;\xi)] = \nabla f_i(x),$$

(ii) Intra- and inter- node Variance Bound. The following holds for all $\xi^{(i)} \sim \mathcal{D}_i$, $i, j \in [N]$ and $x \in \mathbb{R}^d$:

$$\mathbb{E}\|\nabla f_i(x;\xi^{(i)}) - \nabla f_i(x)\|^2 \leq \sigma^2.$$
$$\|\nabla f_i(x) - \nabla f_j(x)\|^2 \leq \zeta^2$$

The assumption 1-(ii) is a typical assumption used in FL algorithms to constrain the data heterogeneity. ζ is the heterogeneity parameter and represents the level of data heterogeneity. If datasets across each worker node have the identical distributions, i.e., $D_i = D_j$ for all $i, j \in [N]$, then we have $\zeta = 0$, corresponds to the homogeneous data setting (I.I.D setting). In this paper, we consider the heterogeneous data setting and $\zeta > 0$.

Assumption 2. *Each component function $f_i(x;\xi)$ has a L-Lipschitz gradient, i.e., $\forall x_1, x_2$, we have*

$$\mathbb{E}\|\nabla_x f_i(x_1;\xi) - \nabla_x f(x_2;\xi)\| \leq L\|x_1 - x_2\|,$$

By using convexity of $\|\cdot\|$ and assumption 2, we have

$$\begin{aligned}&\|\nabla_x f(x_1) - \nabla_x f(x_2)\|\\ &= \|\mathbb{E}[\nabla_x f(x_1;\xi) - \nabla_x f(x_2;\xi)]\|\\ &\leq \mathbb{E}\|\nabla_x f(x_1;\xi) - \nabla_x f(x_2;\xi)\| \leq L\|x_1 - x_2\|\end{aligned}$$

Assumption 2 is Lipschitz smooth, it is still a widely used assumption in optimization analysis. Many typical centralized stochastic algorithms use this assumption, such as SPIDER (Fang et al. 2018), STORM (Cutkosky and Orabona 2019) . Similarly, it is used in FL algorithms such as MIME (Karimireddy et al. 2020a), Fed-GLOMO (Das et al. 2022) and STEM(Khanduri et al. 2021).

Assumption 3. *The function $F(x)$ is bounded below in $\mathcal{X}$,* i.e., $F^* = \inf_{x\in\mathcal{X}} F(x) > -\infty$.

Assumption 4. *In our algorithms, the adaptive matrices A_t for all $t \geq 1$ for updating the variables x is a diagonal matrix and satisfies $\lambda_{\min}(A_t) \geq \rho > 0$, where ρ is an appropriate positive number based on its definition.*

Assumption 4 ensures that the adaptive matrices A_t, $\forall t \geq 1$, are positive definite, as in (Huang, Li, and Huang 2021). The adaptive matrices A_t are diagonal matrices, and we do not need to inverse the matrix A_t.

Assumption 5. *(Bounded Gradients). The function $f_i(x)$ have G-bounded gradients, i.e., for any $i \in [N], x \in \mathbb{R}^d$, we have $\|\nabla f_i(x)\| \leq G$.*

Assumption 5 is used to provide the upper bound of the gradient in the adaptive methods, as in (Reddi et al. 2020; Chen, Li, and Li 2020; Wang, Lin, and Chen 2022). It is a typical assumption in the adaptive methods to constrain the upper bound of the adaptive learning rate. It is reasonable and often satisfied in practice, for example, it holds for the finite sum problem.

Definition 1. *A point x is called ϵ-stationary point if $\|\nabla f(x)\| \leq \epsilon$. Generally, a stochastic algorithm is defined to achieve an ϵ-stationary point in T iterations if $\mathbb{E}\|\nabla f(x_T)\| \leq \epsilon$.*

Faster Adaptive Federated Learning

In this section, we explore how to design the method to combine adaptive gradient method with federated learning. We propose two algorithms to show the idea behind the design of the adaptive FL methods and how to use the adaptive learning rate properly. We use a counter example to show that naive combination of local adaptive update might result in divergence, and then propose our fast adaptive federated learning method (i.e., FAFED).

Divergence of Local Adaptive Federated Learning

The FedAdam, FedYogi, FedAdagrad proposed in (Reddi et al. 2020) and FedAMS proposed in (Wang, Lin, and Chen 2022) adjust the adaptive learning rate on the server. These methods have a main drawback that adaptive term cannot adjust the performance of the model in the local update, and introduce an extra global learning rate to tune.

To improve the algorithm, the most straightforward way to design an adaptive federated learning method is to add an adaptive term on each worker node and run an existing adaptive method, such as Adam, SuperAdam locally, and then average the model periodically after the inner loop. For ease of understanding, we design the adaptive method in algorithm 1 based on FedAvg. Each work node runs local SGD with an adaptive learning rate independently. The model parameters $\{x_{t,i}\}_{i=1}^N$ are averaged after inner loop, as the FedAvg.

However, this design might suffer convergence issues and algorithm 1 can fail to converge to stationary points regardless of parameter selection (Chen, Li, and Li 2020). It is because of heterogeneous data settings and the fact that the adaptive learning rates on different nodes are different. As a result, the global model moves away from the global optima point. Following (Chen, Li, and Li 2020), theorem 1 uses an example to present the details of step update in the algorithm 1 and shows that in some cases, divergence is unavoidable no matter how we choose the tuning parameters.

Theorem 1. *Suppose the sequence $\{\bar{x}_t\}_{t=1}^T$ are generated from algorithm 1 using stochastic partial derivatives. $\{\bar{x}_t\}_{t=1}^T$ might fail to converge to non-stationary points regardless of tuning parameter selection.*

Proof. We utilize a counter example to prove theorem 1 and consider a simple 1-dimensional case with N = 3 worker nodes as:

$$f_1 = \begin{cases} 3x^2, & |x| \leq 1, \\ 6|x| - 2, & |x| > 1 \end{cases}$$

$$f_2 = f_3 = \begin{cases} -x^2, & |x| \leq 1 \\ -2|x| + 1, & |x| > 1 \end{cases}$$

$$f(x) = \frac{1}{3}\sum_{i=1}^{3} f_i(x) \begin{cases} \frac{1}{3}x^2, & |x| \leq 1, \\ \frac{2}{3}|x|, & |x| > 1 \end{cases}$$

It is clear that $x = 0$ is the unique stationary point. We begin from step $t = 0$. Assume $\eta = 0.1, \beta = 0.5$ and $v_{0,i} =$

Algorithm 1: Naive adaptive FedAvg Algorithm

```
1:  Input: $T$, tuning parameters $\{\beta, \eta\}$, $v_{0,i}$ and mini-batch
    size $b_0$;
2:  initialize: Initialize: $\mathbf{x}_i \in \mathbb{R}^d$ for $i \in [N]$,
3:  for $t = 1, 2, \ldots, T$ do
4:      Client i $\in [N]$:
5:      Draw mini-batch samples $\mathcal{B}_{t,i} = \{\xi_i^j\}_{j=1}^{b_0}$ with
        $|\mathcal{B}_t| = b_0$ from $D_i$ locally, and compute stochastic
        partial derivatives $\hat{g}_{t,i} = \nabla_x f_i(x_{t,i}; \mathcal{B}_{t,i})$
6:      $v_{t,i} = \beta v_{t-1,i} + (1-\beta)\,\hat{g}_{t,i}^2$
7:      if $\mod(t,q) = 0$ then
8:          Set $x_{t+1,i} = \bar{x}_{t+1,i} = \frac{1}{N}\sum_{j=1}^N \left(x_{t,j} - \eta\frac{\hat{g}_{t,j}}{\sqrt{v_{t,i}}}\right)$
9:      else
10:         Set $x_{t+1,i} = x_{t,i} - \eta\frac{\hat{g}_{t,i}}{\sqrt{v_{t,i}}}$
11:     end if
12: end for
13: Output: $\bar{x}$ chosen uniformly random from $\{\bar{x}_t\}_{t=1}^T$.
```

0 and the initial point is $x_{0,i}$ = 10 for $i = 1, 2, 3$. With the first update (t = 1), for the f_1, we have $g_{0,1}$ = 6, and $v_{0,1}$ = $0.5 \times 6^2 = 18$. For $i = 2, 3$, we have $g_{0,i}$ = -2 and $v_{0,1}$ = 2. Following the algorithm 1 each worker node has its adaptive learning rate. Thus, after the first update, we have $x_{1,1} = 10 - \frac{0.1}{3\sqrt{2}} \times 6 = 9.858$, and $x_{1,2} = x_{1,3} = 10 + \frac{0.1}{\sqrt{2}} \times 2$ = 10.14, and we have $\bar{x} = 10.05$.

The global model moves towards the opposite direction. We continue to show the following steps. We still have $g_{t,1}$ = 6 and $g_{t,2} = g_{t,3} = -2$. $v_{t,1}$ = $(1-\beta^t) \times 6^2$ and $v_{t,2} = v_{t,3} = (1-\beta^t) \times 2^2$. Therefore, $x_{t,1}$ always updates by $-6\eta/\sqrt{(1-\beta^t) \times 6^2}$ and $x_{t,2}, x_{t,3}$ always update by $2\eta/\sqrt{(1-\beta^t) \times 2^2}$. As a result, the averaged model parameter will update $\frac{\eta}{3\sqrt{(1-\beta^t)}}$. It is the opposite of the direction of convergence and is independent of the choice of parameters η and β.

Therefore, after the first inner loop on each worker node and averaging step on the central server, the global model moves away from the optima point. In the following steps, each worker node continues running the local SGD with an adaptive learning rate from the same point. The global model keeps moving away from the optima point after each inner loop training. Finally, the global model fails to converge to the optimal point.

From this example, we could see the model diverges no matter what tuning parameters we choose. The divergence is caused by the heterogeneous data setting and the non-consensus of adaptive learning rates on different worker nodes. This suggests that we should combine the gradient information across nodes when we design the adaptive method in the FL setting. Thus, we use the sharing adaptive learning rate in the Algorithm 2 to avoid divergence. □

Faster Adaptive Federated Learning Method

In the above subsection, we showed that SGD-based local adaptive learning method could diverge even in a very simple example regardless of tuning parameters selection. In this subsection, we propose a novel fast adaptive federated learning algorithm (FAFED) with shared adaptive learning rates for solving the problem under the heterogeneous data setting. Specifically, our FAFED algorithm is summarized in algorithm 2.

Algorithm 2: FAFED Algorithm

```
1: Input: $T$, Parameters: $\beta, \eta_t, \alpha_t$, the number of local updates $q$, and mini batch size $b$ and initial batch-size $B$;
2: initialize: Initialize: $x_{0,i} = \bar{x}_0 = \frac{1}{N}\sum_{i=1}^N x_{0,i}$. $m_{0,i} = \bar{m}_0 = \frac{1}{N}\sum_{i=1}^N \hat{m}_{0,i}$ with $\hat{m}_{0,i} = \nabla_x f(x_{0,i};\mathcal{B}_{0,i})$ and $v_{0,i} = \bar{v}_0 = \frac{1}{N}\sum_{i=1}^N \hat{v}_{0,i}$ with $\hat{v}_{0,i} = (\nabla_x f(x_{0,i};\mathcal{B}_{0,i}))^2$ where $|\mathcal{B}_{0,i}| = B$ from $D_i$ for $i \in [N]$. $A_0 = \mathrm{diag}(\sqrt{\bar{v}_0} + \rho)$
3: $x_{1,i} = x_{0,i} - \eta_0 m_{0,i}$, for all $i \in [N]$
4: for $t = 1, 2, \ldots, T$ do
5:    Client i $\in [N]$:
6:    Draw mini-batch samples $\mathcal{B}_{t,i} = \{\xi_i^j\}_{j=1}^b$ with $|\mathcal{B}_{t,i}| = b$ from $D_i$ locally, and compute stochastic partial derivatives $\hat{g}_{t,i} = \nabla_x f_i(x_{t,i};\mathcal{B}_{t,i})$ and $\hat{g}_{t-1,i} = \nabla_x f_i(x_{t-1,i};\mathcal{B}_{t,i})$
7:    $m_{t,i} = \hat{g}_{t,i} + (1-\alpha_t)(m_{t-1} - \hat{g}_{t-1,i})$
8:    $v_{t,i} = \beta v_{t-1,i} + (1-\beta)\,\hat{g}_{t,i}^2$
9:    if $\mod(t,q) = 0$ then
10:       $v_{t,i} = \bar{v}_t = \frac{1}{N}\sum_{i=1}^N v_{t,i}$ and $A_t = \mathrm{diag}(\sqrt{\bar{v}_t} + \rho)$
11:       $m_{t,i} = \bar{m}_t = \frac{1}{N}\sum_{i=1}^N m_{t,i}$
12:       $x_{t+1,i} = \bar{x}_{t+1} = \frac{1}{N}\sum_{i=1}^N (x_{t,i} - \eta_t A_t^{-1} m_{t,i})$
13:    else
14:       $A_t = A_{t-1}$
15:       $x_{t+1,i} = x_{t,i} - \eta_t A_t^{-1} m_{t,i}$
16:    end if
17: end for
18: Output: $\bar{x}$ chosen uniformly random from $\{\bar{x}_t\}_{t=1}^T$.
```

At the step 8 in algorithm 2, we use the coordinate-wise adaptive learning rate as in Adam (Kingma and Ba 2014), defined as:

$$v_{t,i} = \beta v_{t-1,i} + (1-\beta)\left(\nabla_x f_i(x_{t,i};\mathcal{B}_{t,i})\right)^2 \tag{2}$$

where $\beta \in (0,1)$. At the step 10 in algorithm 2, we add a periodic averaging step for local adaptive learning rate $v_{t,i}$ at the server side. Then we use $\bar{v}_{t,i}$ to generate an adaptive matrix $A_t = \mathrm{diag}(\sqrt{\bar{v}_t} + \rho)$, where $\rho > 0$. In fact, the adaptive vector v_t can be given with different adaptive learning rate methods, such as the global adaptive learning rate, AdaGrad-Norm (Ward, Wu, and Bottou 2019), and the A_t keeps the same form. The tuning parameter ρ is used to balance the adaptive information with noises.

In the local update, different from algorithm 1, algorithm 2 use the shared adaptive learning rates to avoid model divergence. At the step 15 in algorithm 2, the same A_t is used for local updates of different work nodes. The idea behind the design is that $v_{t,i}$ can be viewed as the second-moment estimation of the gradients, thus A_t established on the average of $v_{t,i}$ is also an estimation of the second moment of the global model. With the average of adaptive information, A_t could follow the global direction and avoid the divergence issue in the algorithm 1.

At step 7 in algorithm 2, we use the momentum-based variance reduced gradient estimator $m_{t,i}$, to track the gradient and update the model, defined as:

$$\begin{aligned} m_{t,i} =& \nabla_x f_i(x_{t,i};\mathcal{B}_{t,i}) \\ &+ (1-\alpha_t)(m_{t-1} - \nabla_x f_i(x_{t-1,i};\mathcal{B}_{t,i}) \end{aligned} \tag{3}$$

where $\alpha_t \in (0,1)$. At the step 11 in algorithm 2, the gradient estimator $m_{t,i}$ is also synchronized and averaged on the server.

Overall, the local servers run adaptive updates locally with the shared adaptive learning rates, and the global server aggregates the model parameters, gradient estimator $m_{t,i}$ and the second-moment estimator $v_{t,i}$ every q steps. In the next section, we will establish the theoretical convergence guarantee of the proposed algorithm.

Convergence Analysis of Our Algorithm

In this subsection, we study the convergence properties of our new algorithm under Assumptions 1, 2, 3, 4, and 5. The details about proofs are provided in the supplementary materials.

Given the sequence $\{\bar{x}\}_{t=1}^T$ generated from our algorithms, we first define a useful convergence metric as follows:

$$\mathcal{M}_t = \frac{1}{4\eta_t^2}\left\|\bar{x}_{t+1} - \bar{x}_t\right\|^2 + \frac{1}{4\rho^2}\left\|\nabla f\left(\bar{x}_t\right) - \bar{m}_t\right\|^2 \tag{4}$$

where these two terms of $\mathcal{M}_t$ measure the convergence of the iteration solution of $\{\bar{x}\}_{t=1}^T$. The new convergence measure is tighter than the standard gradient norm metric, $\|\nabla f(\bar{x}_t)\|$, and we complete the final convergence analysis based on it.

Theorem 2. *Suppose that sequence $\{x_t\}_{t=1}^T$ are generated from algorithm 2. Under the above Assumptions (1,2,3,4,5), given that $\forall t \geq 0$, $\alpha_{t+1} = c\eta_t^2$, $c = \frac{1}{12LI\bar{h}^3\rho^2} + \frac{60L^2}{bN\rho^2} \leq \frac{120L^2}{bN\rho}$, $w = max(\frac{3}{2}, w \leq 1728L^3I^3\bar{h}^3 - t)$ $\bar{h} = \frac{N^{2/3}}{L}$, and set*

$$\eta_t = \frac{\rho\bar{h}}{(w_t + t)^{1/3}} \tag{5}$$

then we have

$$\begin{aligned} &\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f(\bar{x}_t)\| \leq G'\sqrt{\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\mathcal{M}_t]} \\ &\leq G'\left[\left[\frac{12Lq}{\rho T} + \frac{L}{\rho(NT)^{2/3}}\right]\mathbb{E}\left[f\left(\bar{x}_0\right) - f^*\right] + \frac{6q\sigma^2}{T\rho^2}\right. \\ &+ [\frac{12^2 \times 150q}{b^2\rho^2 T} + \frac{1800}{b^2\rho^2(NT)^{2/3}}]\left[\frac{5\sigma^2}{3} + \frac{3\zeta^2}{2}\right](\ln T + 1) \\ &+ \left.\frac{\sigma^2}{2(NT)^{2/3}\rho^2}\right]^{1/2} \end{aligned}$$

where $G' = 4\sqrt{(\sigma^2 + G^2 + \rho^2)}$

Remark 1. *(Complexity) Without loss of generality, let $B = bq$ and $b = O(1)(b \geq 1)$, and choose $q = \big(T/N^2\big)^{1/3}$. Based on the definition of the ε-stationary point, namely, $\mathbb{E}\|\nabla f(x_T)\| \leq \epsilon$ and $\mathbb{E}\big[\mathcal{M}_T\big] \leq \epsilon^2$. we get $T = \tilde{O}(N^{-1}\varepsilon^{-3})$. And $\frac{T}{q} = (NT)^{2/3} = \tilde{O}(\varepsilon^{-2})$, Because the sample size b is a constant, the total sample cost is $\tilde{O}(N^{-1}\varepsilon^{-3})$ and the communication round is $\tilde{O}(\varepsilon^{-2})$ for finding an ε-stationary point that matches the state of the art of gradient complexity bound given in for solving the problem. And $\tilde{O}(N^{-1}\varepsilon^{-3})$ exhibits a linear speed-up compared with the aforementioned centralized optimal algorithms, such as SPIDER and STORM (Fang et al. 2018; Cutkosky and Orabona 2019).*

Remark 2. *(Data Heterogeneity) We use the ζ to present the data heterogeneity. From final results, it is shown that larger ζ (higher data heterogeneity) will slow down the training.*

Remark 3. *Due to Assumption 4 and the definition of A_t, the smallest eigenvalue of the adaptive matrix A_t has a lower bound $\rho > 0$. It balances the adaptive information in the adaptive learning rate. Generally, we choose $\rho = O(1)$ and we do not choose a very small or large parameter in practice.*

Experimental Results

In this section, we evaluate our algorithms with language modeling task and image classification tasks. We compare our algorithms with the existing state-of-the-art algorithms, including FedAvg, SCAFFOLD (Karimireddy et al. 2020b), STEM (Khanduri et al. 2021), FedAdam (Reddi et al. 2020) and FedAMS (Wang, Lin, and Chen 2022). Experiments are implemented using PyTorch, and we run all experiments on CPU machines with 2.3 GHz Intel Core i9 as well as NVIDIA Tesla P40 GPU.

Language Modeling Task

The WikiText2 dataset is used in the experiment and the data is partitioned by 16 worker nodes. Over the WikiText2 dataset, we train a 2-layer LSTM (Hochreiter and Schmidhuber 1997) with 650-dimensional word embeddings and 650 hidden units per layer. We used a batch size of 20 and an inner loop number q of 10 in the experiment, and set the dropout rate as 0.5. To avoid the case of an exploding gradient in LSTM, we also clip the gradients by norm 0.25 (Huang, Li, and Huang 2021).

Grid search is used to choose learning rates for each optimizer. In FedAvg, SCAFFOLD, FedAdam, and FedAMS, we set the learning rate as 10. The global learning rate in the SCAFFOLD is 1. Given that the large global learning rate in FedAdam and FedAms causes the divergence and big fluctuation, we tune it as $0.03 (\approx 10^{-1.5})$. In STEM algorithm, we set $\bar{\kappa}$ as 20, $w = \sigma = 1$, and the step-size is diminished in each epoch as in (Khanduri et al. 2021). In FAFED algorithm, we set $\rho \bar{h}$ as 1 and $w = 1$ and decrease the step size as (5). In the FedAdam, FedAMS, STEM and FAFED algorithm, the momentum parameters, such as α_t, β, β_1 and β_2 are chosen from the set $\{0.1, 0.9\}$. Their adaptive parameters τ or ρ are chosen as 0.01.

Figure 1 shows both training loss and training perplexities. Our FAFED algorithm outperforms all other baseline optimizers. Although FedAdam also has a good performance with an adaptive learning rate, it presents clear fluctuation at the beginning of the training phase because it utilizes the global adaptive method. FedAMS is worse because the adaptive term of FAFED and FedAdam is more flexible, while the adaptive term is monotonically increasing in FedAMS.

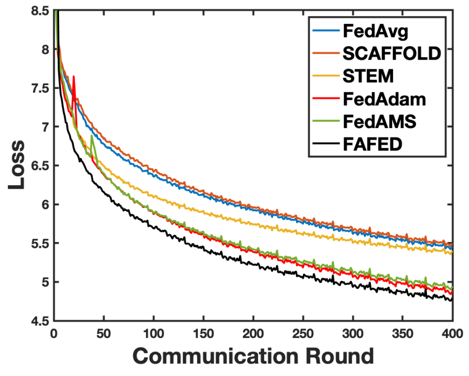
Image Classification Tasks

In the second task, we conduct image classification tasks on the Fashion-MNIST dataset as in (Nouiehed et al. 2019), MNIST dataset and CIFAR-10 dataset with 20 worker nodes in the network. The fashion-MNIST dataset and MNIST dataset includes $60,000$ training images and $10,000$ testing images classified into 10 classes. Each image in both datasets contains 28×28 arrays of the grayscale pixel. CIFAR-10 dataset includes $50,000$ training images and $10,000$ testing images. $60,000$ 32×32 color images are classified into 10 categories. Each worker node holds the same Convolutional Neural Network (CNN) model as the classifier. We use cross entropy as the loss function. The network structures are provided in the supplementary material.

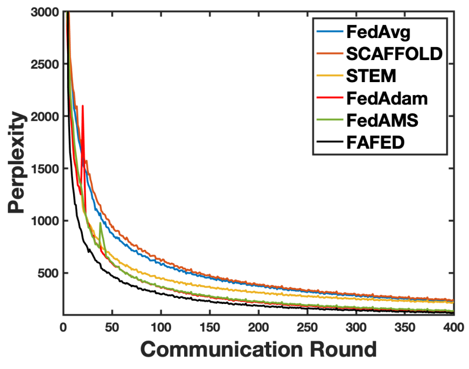
We consider three different heterogeneity settings: low heterogeneity, moderate heterogeneity and high heterogeneity. For the low heterogeneity setting, datasets in different worker nodes have 95% similarity (Karimireddy et al. 2020b). In the real-world application, the data on the different worker nodes are usually completely different, and they even have different categories. Then we consider two more challenging settings. For moderate heterogeneity settings and high heterogeneity settings, the datasets are divided into disjoint sets across all worker nodes. In the moderate heterogeneity setting, each worker node holds part of the data from all the classes, while for the high heterogeneity setting, each worker node only has a part of the total classes (5 out of 10 classes).

We carefully tune hyperparameters for all methods. We run grid search for step size, and choose the step size in the set $\{0.001, 0.01, 0.02, 0.05, 0.1\}$. We set the global learning rate as 1 for SCAFFOLD. For FedAdam and FedAMS, we set global learning from the set $\{10^{-1.5}, 10^{-2}, 10^{-2.5}, \}$, based on the Fig. 2 in (Reddi et al. 2020). The adaptive parameter τ is chosen as 0.01. Given that the datasets are divided by all worker nodes, the number of data points in each worker node is limited. Heterogeneity settings also slow down the training. Thus, the number of local steps required increases, and the methods perform badly with diminishing step size because it decreases rapidly. We choose the fixed step size for STEM and FAFED. We choose the momentum parameter in the set $\{0.1, 0.9\}$. The β_1 and β_2 in FedAdam and FedAMS, and β in FAFED are chosen from $\{0.1, 0.9\}$. The batch-size b is in $\{5, 50, 100\}$ and the inner loop number $q \in \{5, 10, 20\}$. With the increase of data heterogeneity from low heterogeneity to high heterogeneity, we increase the batch size and decrease the inner loop number.

Discussion: The goal of our experiments is two-fold: (1) To compare the performance of FAFED with other algo-

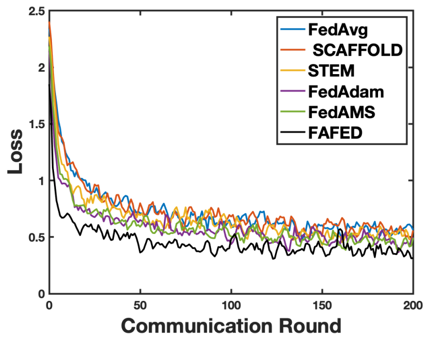


(a) Training Loss

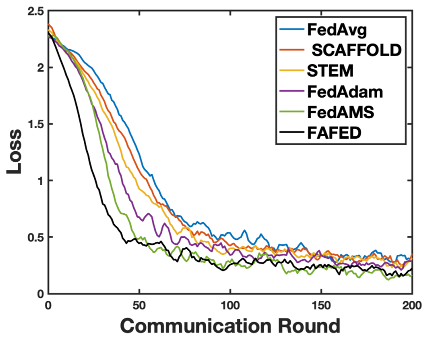


(b) Training perplexities

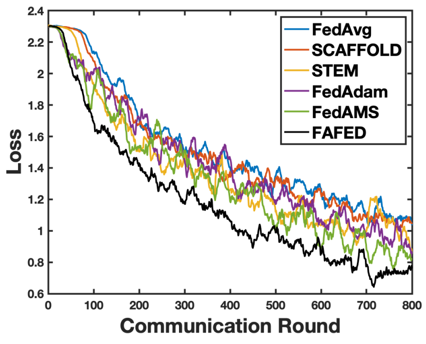
Figure 1: Experimental results of WikiText2 for language modeling task.



(a) Fashion-MNIST



(b) MNIST



(c) CIFAR-10

Figure 2: Training loss vs the number of communication rounds for low heterogeneity setting.

rithms in different heterogeneity settings during the training phase with training datasets; (2) To demonstrate the model performance on the test datasets.

In Figures 2, and figure (Seen in the supplementary materials), we show the performance of FAFED and other baseline methods against the number of communication rounds, namely back-and-forth communication rounds between each worker node and the central server on three datasets with different heterogeneity setting in the image classification task. From Figures, we can find that our algorithms consistently outperform the other baseline algorithms. Compared with FedAdam and FedAMS, our adaptive method has lower fluctuation (e.g., the beginning phase of FedAdam). That is because we use the adaptive learning rate locally, while FEDADAM just scales the model parameters in the central server after multistep training.

Finally, we focus on the performance on the testing datasets. In Tables (Seen in the supplementary mateirals), we show the testing accuracy of FAFED and that of other algorithms on the Fashion-MNIST dataset for different heterogeneity settings after training with the same epochs. Although, with the increasing data heterogeneity, model training becomes more difficult. FAFED performs well under all conditions. It shows the adaptive FL methods adapt well and our method (FAFED) has a good performance in different heterogeneity settings.

Conclusion

In this work, we proposed a novel adaptive algorithm (i.e., FAFED) based on the momentum-based variance reduced technique in the FL setting. We show that adaptive optimizers can be powerful tools and have a good performance in both theoretical analysis and numerical experiments. In the beginning, we explore how to design the adaptive algorithm in the FL setting. By providing a counter example, we present that a naive combination of the local adaptive method with the periodic model average can lead to divergence, and sharing adaptive learning should be considered. Moreover, we provide a solid convergence analysis for our methods, and prove that our algorithm is the first adaptive FL method to reach the best-known samples complexity $O(\epsilon^{-3})$ and communication complexity $O(\epsilon^{-2})$ to find an ϵ-stationary point without large batches. Finally, we conduct experiments on the language modeling task and image classification tasks with different levels of heterogeneous data.

Acknowledgements

This work was partially supported by NSF IIS 1838627, 1837956, 1956002, 2211492, CNS 2213701, CCF 2217003, DBI 2225775.

References

Bao, R.; Wu, X.; Xian, W.; and Huang, H. 2022. Doubly sparse asynchronous learning for stochastic composite optimization. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, 1916–1922.

Chen, X.; Li, X.; and Li, P. 2020. Toward communication efficient adaptive gradient method. In *Proceedings of the 2020 ACM-IMS on Foundations of Data Science Conference*, 119–128.

Cutkosky, A.; and Orabona, F. 2019. Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32.

Das, R.; Acharya, A.; Hashemi, A.; Sanghavi, S.; Dhillon, I. S.; and Topcu, U. 2022. Faster non-convex federated learning via global and local momentum. In *Uncertainty in Artificial Intelligence*, 496–506. PMLR.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

Fang, C.; Li, C. J.; Lin, Z.; and Zhang, T. 2018. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. *Advances in Neural Information Processing Systems*, 31.

Guo, P.; Yang, D.; Hatamizadeh, A.; Xu, A.; Xu, Z.; Li, W.; Zhao, C.; Xu, D.; Harmon, S.; Turkbey, E.; et al. 2022. Auto-FedRL: Federated Hyperparameter Optimization for Multi-institutional Medical Image Segmentation. *arXiv preprint arXiv:2203.06338*.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Hong, J.; Wang, H.; Wang, Z.; and Zhou, J. 2021. Federated robustness propagation: Sharing adversarial robustness in federated learning. *arXiv preprint arXiv:2106.10196*, 1.

Huang, F.; Li, J.; and Huang, H. 2021. Super-adam: faster and universal framework of adaptive gradients. *Advances in Neural Information Processing Systems*, 34.

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

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

Khaled, A.; Mishchenko, K.; and Richtárik, P. 2020. Tighter theory for local SGD on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, 4519–4529. PMLR.

Khanduri, P.; Sharma, P.; Yang, H.; Hong, M.; Liu, J.; Rajawat, K.; and Varshney, P. 2021. Stem: A stochastic two-sided momentum algorithm achieving near-optimal sample and communication complexities for federated learning. *Advances in Neural Information Processing Systems*, 34.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, T.; Sahu, A. K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; and Smith, V. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2: 429–450.

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

Nouiehed, M.; Sanjabi, M.; Huang, T.; Lee, J. D.; and Razaviyayn, M. 2019. Solving a class of non-convex min-max games using iterative first order methods. *Advances in Neural Information Processing Systems*, 32.

Reddi, S.; Charles, Z.; Zaheer, M.; Garrett, Z.; Rush, K.; Konečnỳ, J.; Kumar, S.; and McMahan, H. B. 2020. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*.

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

Staib, M.; Reddi, S.; Kale, S.; Kumar, S.; and Sra, S. 2019. Escaping saddle points with adaptive gradient methods. In *International Conference on Machine Learning*, 5956–5965. PMLR.

Sun, J.; Huai, M.; Jha, K.; and Zhang, A. 2022. Demystify Hyperparameters for Stochastic Optimization with Transferable Representations. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1706–1716.

Wang, Y.; Lin, L.; and Chen, J. 2022. Communication-Efficient Adaptive Federated Learning. *arXiv preprint arXiv:2205.02719*.

Ward, R.; Wu, X.; and Bottou, L. 2019. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, 6677–6686. PMLR.

Woodworth, B.; Patel, K. K.; Stich, S.; Dai, Z.; Bullins, B.; Mcmahan, B.; Shamir, O.; and Srebro, N. 2020. Is local SGD better than minibatch SGD? In *International Conference on Machine Learning*, 10334–10343. PMLR.

Xiong, Z.; Li, W.; and Cai, Z. 2023. Federated Generative Model on Multi-Source Heterogeneous Data in IoT. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Xu, A.; and Huang, H. 2022. Coordinating momenta for cross-silo federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 8735–8743.

Xu, A.; Li, W.; Guo, P.; Yang, D.; Roth, H. R.; Hatamizadeh, A.; Zhao, C.; Xu, D.; Huang, H.; and Xu, Z. 2022. Closing the Generalization Gap of Cross-silo Federated Medical Image Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20866–20875.

Yang, H.; Fang, M.; and Liu, J. 2021. Achieving linear speedup with partial worker participation in non-iid federated learning. *arXiv preprint arXiv:2101.11203*.

Yu, H.; Jin, R.; and Yang, S. 2019. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. In *International Conference on Machine Learning*, 7184–7193. PMLR.

Yu, H.; Yang, S.; and Zhu, S. 2019. 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, 5693–5700.

Zeiler, M. D. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Zhang, J.; Karimireddy, S. P.; Veit, A.; Kim, S.; Reddi, S. J.; Kumar, S.; and Sra, S. 2019. Why ADAM beats SGD for attention models. *arXiv preprint arXiv:1912.03194*.

Zhang, X.; Hong, M.; Dhople, S.; Yin, W.; and Liu, Y. 2020. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*.

Zinkevich, M.; Weimer, M.; Li, L.; and Smola, A. 2010. Parallelized stochastic gradient descent. *Advances in neural information processing systems*, 23.