Coordinating Momenta for Cross-Silo Federated Learning

An Xu, Heng Huang*

Electrical and Computer Engineering Department, University of Pittsburgh, PA, USA
an.xu@pitt.edu, heng.huang@pitt.edu

Abstract

Communication efficiency is crucial for federated learning (FL). Conducting local training steps in clients to reduce the communication frequency between clients and the server is a common method to address this issue. However, this strategy leads to the client drift problem due to *non-i.i.d.* data distributions in different clients which severely deteriorates the performance. In this work, we propose a new method to improve the training performance in cross-silo FL via maintaining double momentum buffers. In our algorithm, one momentum buffer is used to track the server model updating direction, and the other one is adopted to track the local model updating direction. More important, we introduce a novel momentum fusion technique to coordinate the server and local momentum buffers. We also derive the first theoretical convergence analysis involving both the server and local standard momentum SGD. Extensive deep FL experimental results verify that our new approach has a better training performance than the FedAvg and existing standard momentum SGD variants.

Introduction

With deep learning models becoming prevalent but data-hungry, data privacy emerges as an important issue. Federated learning (FL) (Konečnỳ et al. 2016) was thus introduced to leverage the massive data from different clients for training models without directly sharing nor aggregating data. More recently, an increasing number of FL techniques focus on addressing the cross-silo FL (Kairouz et al. 2019; Gu et al. 2021) problem, which has more and more real-world applications, such as the collaborative learning on health data across multiple medical centers (Liu et al. 2021; Guo et al. 2021) or financial data across different corporations and stakeholders. During the training, the server only communicates the model weights and updates with the participating clients.

However, the deep learning models require many training iterations to converge. Unlike workers in data-center distributed training with large network bandwidth and relatively low communication delay, the clients participating in the collaborative FL system are often faced with much more unstable conditions and slower network links due to the geo-distribution. Typically, (Bonawitz et al. 2019) showed that one communication round in FL could take about 2 to 3 minutes in practice. To address the communication inefficiency, the *de facto* standard method FedAvg was proposed in (McMahan et al. 2017). In FedAvg, the server sends clients the server model. Each client conducts many local training steps with its local data and sends back the updated model to the server. The server then averages the models received from clients and finishes one round of training. The increased local training steps can reduce the communication rounds and cost. Here we also refer to the idea of FedAvg as periodic averaging, which is closely related to local SGD (Stich 2018; Yu, Jin, and Yang 2019). Another parallel line of works is to compress the communication to reduce the volume of the message (Rothchild et al. 2020; Karimireddy et al. 2019; Gao, Xu, and Huang 2021; Xu, Huo, and Huang 2021, 2020a,b). In this paper, we do not focus on communication compression.

Although periodic averaging methods such as FedAvg greatly improve the training efficiency in FL, a new problem named client drift arises. The data distributions of different clients are *non-i.i.d.* because we cannot gather and randomly shuffle the client data as data-center distributed training does. Therefore, the stochastic gradient computed at different clients can be highly skewed. Given that we do many local training steps in each training round, skewed gradients will cause local model updating directions to gradually diverge and overfit local data at different clients. This client drift issue can deteriorate the performance of FedAvg drastically (Hsieh et al. 2020; Hsu, Qi, and Brown 2019), especially with a low similarity of the data distribution on different clients and a large number of local training steps.

As a method to reduce variance and smooth the model updating direction (Cutkosky and Orabona 2019) to accelerate optimization, momentum SGD has shown its power in training many deep learning models in various tasks (Sutskever et al. 2013). Local momentum SGD (Yu, Jin, and Yang 2019) (*i.e.*, FedAvgLM) maintains momentum for the stochastic gradient in each training step but requires averaging local momentum buffer at the end of each training round. Therefore, FedAvgLM requires $\times 2$ communication cost compared with FedAvg. One strategy to achieve the same communication cost as FedAvg is resetting the

*This work was supported by NSF IIS 1845666, 1852606, 1838627, 1837956, 1956002, IIA 2040588.

local momentum buffer to zero at the end of each training round (FedAvgLM-Z) (Seide and Agarwal 2016; Wang and Joshi 2018; Wang et al. 2020). Hsu, Qi, and Brown (2019) and Huo et al. (2020) proposed server momentum SGD (FedAvgSM) which maintains the momentum for the average local model update in a training round other than the stochastic gradient. The idea of FedAvgSM has been previously proposed in (Chen and Huo 2016) for speech models and in (Wang et al. 2019) for distributed training, but neither of them has applied it to FL. Hsu, Qi, and Brown (2019) and Huo et al. (2020) empirically showed the ability of FedAvgSM to tackle client drift in FL, and Wang et al. (2019) and Huo et al. (2020) provided the convergence analysis of FedAvgSM. Throughout this paper, we refer to the naive combination of FedAvgSM and FedAvgLM(-Z) as FedAvgSLM(-Z). However, FedAvgSLM(-Z) has **no convergence guarantee** to the best of our knowledge. Moreover, there is a lack of understanding in **the connection between the server and local momenta** in FL. Whether we can further **improve standard momentum-based methods** in FL remains another question.

To address the above challenging problems, in this paper, we propose a new double momentum SGD (DOMO) algorithm, which leverages double momentum buffers to track the server and local model updating directions separately. We introduce a novel momentum fusion technique to coordinate the server and local momentum buffers. More importantly, we provide the theoretical analysis for the convergence of our new method for non-convex problems. Our new algorithm focuses on addressing the cross-silo FL problem, considering the recently increasing needs on it as described at the beginning of this section. We also regard it as time-consuming to compute the full local gradient and focuses on stochastic methods. We summarize our major technical contributions as follows:

- Propose a new double momentum SGD (DOMO) method with a novel momentum fusion technique.
- Derive the first convergence analysis involving both server and local standard momentum SGD in non-convex settings and under mild assumptions, show incorporating server momentum's convergence benefits over local momentum SGD for the first time, and provide new insights into their connection.
- Conduct deep FL experiments to show that DOMO can improve the test accuracy by up to 5% compared with the state-of-the-art momentum-based method when training VGG-16 on CIFAR-10, while the naive combination FedAvgSLM(-Z) may sometimes hurt the performance compared with FedAvgSM.

Background and Related Work

FL can be formulated as an optimization problem of $\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{K}\sum_{k=0}^{K-1} f^{(k)}(\mathbf{x})$, where $f^{(k)}$ is the local loss function on client k, $\mathbf{x}$ is the model weights with d as its dimension, and K is the number of clients. Other basic notations are listed below. In FedAvg, the client trains the local model for P steps using stochastic gradient $\nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$ and sends local model update $\mathbf{x}_{r,P}^{(k)} - \mathbf{x}_{r,0}^{(k)}$ to server. Server then takes an average and updates the server model via $\mathbf{x}_{r+1} = \mathbf{x}_r - \frac{\alpha}{K}\sum_{k=0}^{K-1}(\mathbf{x}_{r,P}^{(k)} - \mathbf{x}_{r,0}^{(k)})$.

Basic notations:

- Training round (total): r (R); Local training steps (total): p (P); Client (total): k (K);
- Global training step (total): $t = rP + p$ ($T = RP$); Server, local learning rate: α, η;
- Momentum fusion constant β; Server, local momentum constant: μ_s, μ_l;
- Server, local momentum buffer: $\mathbf{m}_r$, $\mathbf{m}_{r,p}^{(k)}$; Server, (average) local model: $\mathbf{x}_r$, $\mathbf{x}_{r,p}^{(k)}$ ($\bar{\mathbf{x}}_{r,p}^{(k)}$);
- Local stochastic gradient: $\nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$, where $\xi_{r,p}^{(k)}$ is the sampling random variable;
- Local full gradient: $\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) = \mathbb{E}_{\xi_{r,p}^{(k)}}[\nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})]$ (unbiased sampling).

Momentum-based. State-of-the-art method server momentum SGD (FedAvgSM) maintains a server momentum buffer with the local model update $\frac{\alpha}{K}\sum_{k=0}^{K-1}(\mathbf{x}_{r,P}^{(k)} - \mathbf{x}_{r,0}^{(k)})$ to update the server model. While local momentum SGD maintains a local momentum buffer with $\nabla F^{(k)}(\mathbf{x}_{r,p}^{(K)}, \xi_{r,p}^{(k)})$ to update the local model. The communication costs compared with FedAvg are $\times 1$, $\times 2$ and $\times 1$ for FedAvgSM, FedAvgLM and FedAvgLM-Z respectively.

Adaptive Methods. (Reddi et al. 2020) applied the idea of using server statistics as in server momentum SGD to adaptive optimizers including Adam (Kingma and Ba 2014), AdaGrad (Duchi, Hazan, and Singer 2011), and Yogi (Zaheer et al. 2018). (Reddi et al. 2020) showed that server learning rate should be smaller than $\mathcal{O}(1)$ in terms of complexity, but the exact value was unknown.

Inter-client Variance Reduction. Variance reduction in FL (Karimireddy et al. 2020b; Acar et al. 2020) refers to correct client drift caused by *non-i.i.d.* data distribution on different clients following the variance reduction convention. In contrast, traditional stochastic variance reduced methods that are popular in convex optimization (Johnson and Zhang 2013; Defazio, Bach, and Lacoste-Julien 2014) can be seen as intra-client variance reduction. Scaffold (Karimireddy et al. 2020b) proposed to maintain a control variate c_k on each client k and add $\frac{1}{K}\sum_{k=0}^{K-1} c_k - c_k$ to gradient $\nabla F_{r,p}^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$ when conducting local training. A prior work VRL-SGD (Liang et al. 2019) was built on a similar idea with c_k equal to the average local gradients in the last training round. Both Scaffold and VRL-SGD have to maintain and communicate local statistics, which makes the clients stateful and requires $\times 2$ communication cost. Mime (Karimireddy et al. 2020a) proposed to apply server statistics locally to address this issue. However, Mime has to compute the full local gradient which can be prohibitive in cross-silo FL. It also needs $\times 2$ communication cost. Besides, Mime's theoretical results are based on Storm (Cutkosky and Orabona 2019) but their algorithm is based on Polyak's momentum. Though theoretically appealing, the variance reduction technique has shown to be ineffective in

Algorithm 1: FL with double momenta.

1: **Input:** local training steps $P \geq 1$, #rounds R, #clients K, server (local) learning rate α (η), server (local) momentum constant μ_s (μ_l), momentum fusion constant β.
2: **Initialize:** Server, local momentum buffer $\mathbf{m}_0, \mathbf{m}_{0,0}^{(k)} = \mathbf{0}$. Local model $\mathbf{x}_{0,0}^{(k)} = \mathbf{x}_0$.
3: **for** $r = 0, 1, \cdots, R-1$ **do**
4: **Client** k:
5: ($r \geq 1$) Receive $\mathbf{x}_{r,0}^{(k)} \leftarrow \mathbf{x}_r$. $\mathbf{m}_r = \frac{1}{\alpha\eta P}(\mathbf{x}_{r-1} - \mathbf{x}_r)$. $\mathbf{m}_{r,0}^{(k)} \leftarrow \mathbf{0}$. *// Reset local momentum.*
6: **for** $p = 0, 1, \cdots, P-1$ **do**
7: Option I: $\mathbf{x}_{r,p}^{(k)} \leftarrow \mathbf{x}_{r,p}^{(k)} - \eta\beta P \mathbf{m}_r \cdot \mathbf{1}_{p=0}$ *// Pre-momentum fusion (DOMO)*
8: $\mathbf{m}_{r,p+1}^{(k)} = \mu_l \mathbf{m}_{r,p}^{(k)} + \nabla F(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$
9: Option I: $\mathbf{x}_{r,p+1}^{(k)} = \mathbf{x}_{r,p}^{(k)} - \eta \mathbf{m}_{r,p+1}^{(k)}$
10: Option II: $\mathbf{x}_{r,p+1}^{(k)} = \mathbf{x}_{r,p}^{(k)} - \eta \mathbf{m}_{r,p+1}^{(k)} - \eta\beta \mathbf{m}_r$ *// Intra-momentum fusion (DOMO-S)*
11: **end for**
12: Send $\mathbf{d}_r^{(k)} = \frac{1}{P}\sum_{p=0}^{P-1} \mathbf{m}_{r,p+1}^{(k)}$ to the server.
13: **Server:**
14: Receive $\mathbf{d}_r^{(k)}$ from client $k \in [K]$. $\mathbf{m}_{r+1} = \mu_s \mathbf{m}_r + \frac{1}{K}\sum_{k=1}^{K} \mathbf{d}_r^{(k)}$.
15: $\mathbf{x}_{r+1} = \mathbf{x}_r - \alpha\eta P \mathbf{m}_{r+1}$. Send $\mathbf{x}_{r+1}$ to client $k \in [K]$.
16: **end for**
17: **Output:** $\mathbf{x}_R$

practical neural networks' optimization (Defazio and Bottou 2018). (Defazio and Bottou 2018) showed that common tricks such as data augmentation, batch normalization (Ioffe and Szegedy 2015), and dropout (Srivastava et al. 2014) broke the transformation locking and deviated practice from variance reduction's theory.

Other. There are some other settings of FL including heterogeneous optimization (Li et al. 2018; Wang et al. 2020), fairness (Mohri, Sivek, and Suresh 2019; Li et al. 2020, 2021), personalization (T Dinh, Tran, and Nguyen 2020; Fallah, Mokhtari, and Ozdaglar 2020; Jiang et al. 2019; Shamsian et al. 2021), etc. These different settings, variance reduction techniques, and server statistics can sometimes be combined. Here we focus on momentum-based FL methods.

New Double Momentum SGD (DOMO)

In this section, we address the connection and coordination of server and local momenta to improve momentum-based methods by introducing our new double momentum SGD (DOMO) algorithm. We maintain both the server and local statistics (momentum buffers). Nevertheless, the local momentum buffer does not make the clients stateful because the local momentum buffer will be reset to zero at the end of each training round for every client.

Motivation. We observe that FedAvgSM applies momentum SGD update after aggregating the local model update from clients in the training round r. Specifically, it updates the model at the server via $\mathbf{x}_{r+1} = \mathbf{x}_r - \alpha\eta P \mathbf{m}_{r+1} = \mathbf{x}_r - \alpha\eta\mu_s \mathbf{m}_r - \frac{\alpha\eta P}{K}\sum_{k=1}^{K} \mathbf{d}_r^{(k)}$ (Algorithm 1 lines 14 and 15). We can see that the server momentum buffer $\mathbf{m}_r$ is only applied at the *server* side after the clients finish the training round r. Therefore, FedAvgSM fails to take advantage of the server momentum buffer $\mathbf{m}_r$ during the local training at the *client* side in the training round r. The same issue exists for FedAvgSLM(-Z), where the local optimizer is also momentum SGD.

Recognizing this issue, we propose DOMO (summarized in Algorithm 1 where $\mathbf{1}$ denotes the indicator function) by utilizing server momentum statistics $\mathbf{m}_r$ to help local training in round r at the *client* side as it provides information on global updating direction. The whole framework can be briefly summarized in the following steps.

1. Receive the initial model from the server at the beginning of the training round.
2. Fuse the server momentum buffer in local training steps.
3. Remove the server momentum's effect from the local model update before sending it to the server.
4. Aggregate local model updates from clients to update model and statistics at the server.

To avoid incurring additional communication cost than FedAvg, we 1) infer the server momentum buffer $\mathbf{m}_r$ via the current and last initial model ($\mathbf{x}_r, \mathbf{x}_{r-1}$), and 2) reset the local momentum buffer $\mathbf{m}_{r,0}^{(k)}$ to zero instead of averaging. Note that for FedAvgSLM, the local momentum buffer has to be averaged.

Momentum Fusion in Local Training Steps. In each local training step, the local momentum buffer is updated following the standard momentum SGD method (Algorithm 1 line 8). We propose two options to fuse server momentum into local training steps. The default Option I is DOMO and the Option II is DOMO-S with "S" standing for "scatter". In DOMO, we apply server momentum buffer $\mathbf{m}_r$ with coefficient βP and learning rate η to the local model before the local training starts. β is the momentum fusion constant. DOMO-S is an heuristic extension of DOMO by evenly scattering this procedure to all the P local training steps. Therefore, the coefficient becomes β instead of βP. Intuitively, the local model updating direction should be adjusted by the direction of the server momentum buffer to alleviate the client drift issue. Furthermore, DOMO-S follows this motivation in a more fine-grained way by adjusting each local momentum SGD training step with the server momentum buffer.

Pre-Momentum, Intra-Momentum, Post-Momentum. To help understand the connection between the server and local momenta better, here we propose new concepts called pre-momentum, intra-momentum, and post-momentum. The naive combination FedAvgSLM(-Z) can be interpreted as post-momentum because the current server momentum buffer $\mathbf{m}_r$ is applied at the end of the training round and after all the local momentum SGD training steps are finished. Therefore, FedAvgSLM(-Z) has no momentum fusion to help local training. While our proposed DOMO can be regarded as pre-momentum because it applies the current server momentum buffer $\mathbf{m}_r$ at the beginning of the training round ($p = 0$) and before the local momentum SGD

training starts. Similarly, the proposed DOMO-S works as intra-momentum because it scatters the effect of the current server momentum buffer $\mathbf{m}_r$ to the whole local momentum SGD training steps. These new concepts shed new insights for the connection and coordination between server and local momenta by looking at the order of applying server momentum buffer and local momentum buffer. Considering that server and local momentum buffers can be regarded as the smoothed server and local model updating direction, the order of applying which one first should not make much difference when the similarity of data distribution across clients is high, *i.e.*, the client drift issue is not severe. However, when the data similarity is low, it becomes more critical to provide the information of server model updating direction during the local training as DOMO (pre-momentum) and DOMO-S (intra-momentum) do.

Aggregate Local Model Updates without Server Momentum. We propose to remove the effect of server momentum $\mathbf{m}_r$ in local model updates (Algorithm 1 line 12) when aggregating them to server. The equivalent server momentum constant would have been deviated to $\mu_s + \beta$ if we would not remove it.

Convergence Analysis

In this section, we will discuss our convergence analysis framework with double momenta and the potential difficulty for the naive combination FedAvgSLM(-Z). There has been little theoretical analysis in existing literature for FedAvgSLM(-Z) possibly due to this theoretical difficulty. After that, we will show how the motivation of DOMO addresses it. This is the first convergence analysis involving both server and local standard momentum SGD to the best of our knowledge. Both resetting and averaging local momentum buffer are considered, though we only reset it in Algorithm 1 for less communication. Please refer to the Supplementary Material for proof details.

We consider non-convex smooth objective function satisfying Assumption 1. We also assume that the local stochastic gradient is an unbiased estimation of the local full gradient and has a bounded variance in Assumption 2. Furthermore, we bound the *non-i.i.d.* data distribution across clients in Assumption 3, which is widely employed in existing works such as (Yu, Jin, and Yang 2019; Reddi et al. 2020; Wang et al. 2019; Karimireddy et al. 2020a). G measures the data similarity in different clients and $G = 0$ corresponds to *i.i.d.* data distribution as in data-center distributed training. A low data similarity will lead to a larger G^2. For simplicity, let f_* denote the optimal global objective value. Other basic notations have been summarized in Section "Background & Related Works".

Assumption 1. *(L-Lipschitz Smoothness) The global objective function $f(\cdot)$ and local objective function $f^{(k)}$ are L-smooth, i.e., $\|\nabla f^{(k)}(\boldsymbol{x}) - \nabla f^{(k)}(\boldsymbol{y})\|_2 \leq L\|\boldsymbol{x} - \boldsymbol{y}\|_2$ and $\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|_2 \leq L\|\boldsymbol{x} - \boldsymbol{y}\|_2, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d, k \in [K]$.*

Assumption 2. *(Unbiased Gradient and Bounded Variance) The stochastic gradient $\nabla F^{(k)}(\boldsymbol{x}, \xi)$ is an unbiased estimation of the full gradient $\nabla f^{(k)}(\boldsymbol{x})$, i.e., $\mathbb{E}_\xi \nabla F(\boldsymbol{x}, \xi) = \nabla f(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathbb{R}^d$. Its variance is also bounded, i.e., $\mathbb{E}_\xi \|\nabla F(\boldsymbol{x}, \xi) - \nabla f(\boldsymbol{x})\|_2^2 \leq \sigma^2, \forall \boldsymbol{x} \in \mathbb{R}^d$.*

Assumption 3. *(Bounded Non-i.i.d. Distribution (Yu, Jin, and Yang 2019; Reddi et al. 2020; Wang et al. 2019; Karimireddy et al. 2020a)) For any client $k \in [K]$ and $\boldsymbol{x} \in \mathbb{R}^d$, there exists $B \geq 0$ and $G \geq 0$, the variance of the local full gradient in each client is upper bounded so that $\frac{1}{K}\sum_{k=0}^{K-1} \|\nabla f^{(k)}(\boldsymbol{x}) - \nabla f(\boldsymbol{x})\|_2^2 \leq G^2$.*

Lemma 1. *(DOMO updating rule) Let $0 \leq r \leq R-1$ and $0 \leq p \leq P-1$. Let $\widehat{\boldsymbol{y}}_{r,p} = \boldsymbol{x}_0 - \frac{\alpha\eta}{(1-\mu_s)K}\sum_{k=0}^{K-1}\sum_{r'=0}^{r}\sum_{p'=0}^{p-1}\boldsymbol{m}_{r',p'+1}^{(k)}$ and $\boldsymbol{z}_{r,p} = \frac{1}{1-\mu_l}\widehat{\boldsymbol{y}}_{r,p} - \frac{\mu_l}{1-\mu_l}\widehat{\boldsymbol{y}}_{r,p-1}$ where $\widehat{\boldsymbol{y}}_{0,-1} = \widehat{\boldsymbol{y}}_{0,0} = \boldsymbol{x}_0$, then*

$$\boldsymbol{z}_{r,p+1} = \boldsymbol{z}_{r,p} - \frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\nabla F^{(k)}(\boldsymbol{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$$

The key of the proof is to find a novel auxiliary sequence $\{\mathbf{z}_{r,p}\}$ that not only has a concise update rule than the mixture of server and local momentum SGD, but also is close to the average local model $\{\overline{\mathbf{x}}_{r,p}\}$. One difficulty is to analyze the server model update at the end of the training round ($\overline{\mathbf{x}}_{r,P} \to \mathbf{x}_{r+1}$) due to server momentum. To tackle it, we design $\mathbf{z}_{r,P} = \mathbf{z}_{r+1,0}$ to facilitate the analysis at the end of the training round. Lemma 1 gives such an auxiliary sequence. Before to analyze the convergence of $\{\overline{\mathbf{x}}_{r,p}\}$ with the help of $\{\mathbf{z}_{r,p}\}$, we only have to bound $\|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2$ (**inconsistency bound**) and $\frac{1}{K}\sum_{k=0}^{K-1}\|\overline{\mathbf{x}}_{r,p} - \mathbf{x}_{r,p}^{(k)}\|_2^2$ (**divergence bound**). The divergence bound measures how the local models on different clients diverges and is more straightforward to analyze since it is only affected by local momentum SGD. The inconsistency bound measures the inconsistency between the auxiliary variable and the average local model as a trade-off for a more concise update rule.

Lemma 2. *(Inconsistency Bound) For DOMO, we have $(\boldsymbol{z}_{r,p} - \overline{\boldsymbol{x}}_{r,p})_{DOMO} = (1 - \frac{\alpha}{1-\mu_s})\frac{\eta}{K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\boldsymbol{m}_{r,p'+1}^{(k)} - \frac{\mu_l\eta}{(1-\mu_l)K}\sum_{k=0}^{K-1}\boldsymbol{m}_{r,p}^{(k)}$; while for FedAvgSLM(-Z), we have $(\boldsymbol{z}_{r,p} - \overline{\boldsymbol{x}}_{r,p})_{FedAvgSLM(-Z)} = (\boldsymbol{z}_{r,p} - \overline{\boldsymbol{x}}_{r,p})_{DOMO} + \frac{\mu_s}{1-\mu_s}\alpha\eta P\boldsymbol{m}_r$.*

Furthermore, assume that $\alpha \geq (1-\mu_s)(1-\mu_l)$, let $h = \frac{\alpha}{1-\mu_s}\frac{1+\mu_l-\mu_l^p}{1-\mu_l} - 1$ for DOMO and $h = \frac{\mu_l}{1-\mu_l}$ for FedAvgLM(-Z), and we have

$$\begin{aligned}\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|\boldsymbol{z}_{r,p} - \overline{\boldsymbol{x}}_{r,p}\|_2^2 \leq \frac{\eta^2}{1-\mu_l}\Big(\sum_{p=0}^{P-1}\frac{h^2\mu_l^p}{1-\mu_l^P}\Big)\cdot \\ \sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F^{(k)}(\boldsymbol{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})\|_2^2 .\end{aligned} \tag{1}$$

Theoretical Difficulty for FedAvgSLM(-Z) but Addressed by DOMO. From Lemma 2, we can see that without momentum fusion, the inconsistency bound for FedAvgSLM-Z has an additional term related to $P\mathbf{m}_r$ compared with DOMO. For the corresponding inconsistency bound, this term will lead to $\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|P\mathbf{m}_r\|_2^2 =$

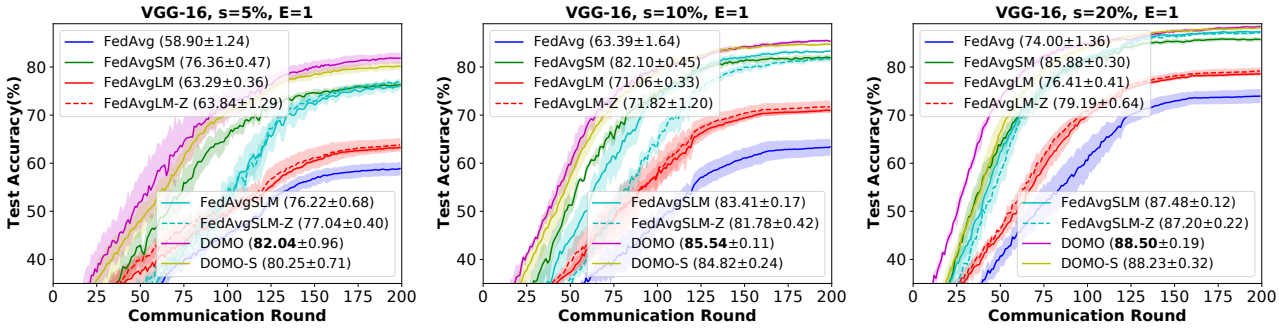

Figure 1: CIFAR-10 training curves using the VGG-16 model with various data similarity s.

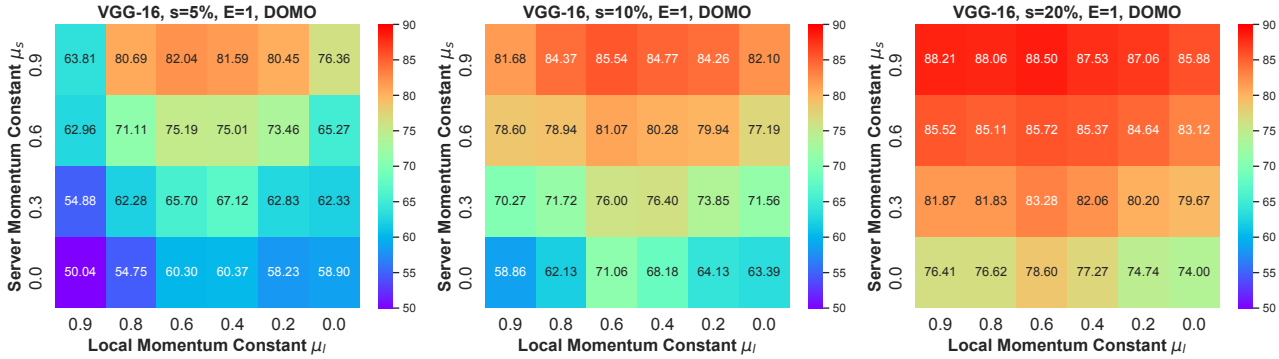

Figure 2: CIFAR-10 test accuracy (%) with various sever momentum constant μ_s and local momentum constant μ_l. $\mu_s = 0$ corresponds to FedAvgLM, $\mu_l = 0$ corresponds to FedAvgLM, $\mu_s = 0 \,\&\, \mu_l = 0$ corresponds to FedAvg, and $\mu_s \neq 0 \,\&\, \mu_l \neq 0$ corresponds to DOMO.

$P^2 \sum_{r=0}^{R-1} \sum_{p=0}^{P-1} \|\mathbf{m}_r\|_2^2$. Intuitively, if we ignore the constant and simply assume that $\|\mathbf{m}_r\|_2^2$ is of the same complexity as $\|\nabla F\|_2^2$, then it causes an additional term of complexity $\mathcal{O}(RP^3\|\nabla F\|_2^2)$ in the inconsistency bound, much larger than the complexity $\mathcal{O}(RP^2\|\nabla F\|_2^2)$ for DOMO in the R.H.S. of Eq. (1). This also means that FedAvgSLM(-Z) is more sensitive to P and may even hurt the performance when P is large as in FL.

Tighten Inconsistency Bound with DOMO. In Lemma 2, we also show that DOMO can tighten the inconsistency bound. Specifically, set $\alpha = (1-\mu_s)(1-\mu_l)$ and DOMO can scale the inconsistency bound down to about $(1-\mu_l)^2$ of that in FedAvgLM(-Z) ($\alpha = 1, \mu_s = 0$). Take the popular momentum constant $\mu_l = 0.9$ as an example, the inconsistency bound of DOMO is reduced to $(1-\mu_l)^2 = 1\%$ compared with FedAvgLM(-Z). Therefore, momentum fusion not only addresses the difficulty for FedAvgSLM(-Z), but also helps the local momentum SGD training. To the best of our knowledge, this is the first time to show that *incorporating server momentum leads to convergence benefits over local momentum*. It is also intuitively reasonable in that server momentum buffer carries historical local momentum information. But we note that this improvement analysis has not reached the optimal yet due to inequality scaling. Therefore, we fine-tune α and β for the best performance in practice.

When $\alpha = 1 - \mu_s$ and $\beta = \mu_s$, we can see that DOMO has the same inconsistency bound as FedAvgLM(-Z) ($\alpha = 1, \mu_s = 0$). Consider the momentum buffer as a smoothed updating direction. Suppose the update of server momentum buffer $\mathbf{m}_{r+1} = \mu_s \mathbf{m}_r + \Delta_r$ becomes steady with $\Delta_r \to \Delta$ (the sum of local momentum buffer in local training), then $\mathbf{m}_r$ will be approximately equal to $\frac{\Delta}{1-\mu_s}$. The coefficient $\frac{1}{1-\mu_s}$ leads to a different magnitude of the server and local momenta. Setting $\beta = \mu_s$ in Lemma 2 gives $\alpha = 1-\mu_s$, balancing the difference by a smaller server learning rate.

With the above lemmas, we have the following convergence analysis theorem for our new algorithm.

Theorem 1. *(Convergence of DOMO) Assume Assumptions 1, 2 and 3 exist. Let* $P \leq \frac{1-\mu_l}{6\eta L}$, $1 - 2\eta L - \frac{4\mu_l^2\eta^2L^2}{(1-\mu_l)^4} \geq 0$, $\alpha = 1-\mu_s$ *and* $\beta = \mu_s$. *For DOMO with either resetting or averaging local momentum buffer in Algorithm 1 line 5, we have*

$$\frac{1}{RP}\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_{r,p})\|_2^2 \leq \frac{2(1-\mu_l)(f(\boldsymbol{x}_0)-f_*)}{\eta RP} + \frac{9\eta^2L^2P^2G^2}{(1-\mu_l)^2} + \frac{\eta L\sigma^2}{(1-\mu_l)}\left(\frac{1}{K} + \frac{3\eta LP}{2(1-\mu_l)} + \frac{2\mu_l^2\eta L}{(1-\mu_l)^4K}\right)$$

Complexity. According to Theorem 1, let $\eta = \mathcal{O}(K^{\frac{1}{2}}R^{-\frac{1}{2}}P^{-\frac{1}{2}})$ and $P = \mathcal{O}(K^{-1}R^{\frac{1}{3}})$, then we have a convergence rate $\frac{1}{RP}\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_{r,p})\|_2^2 =$

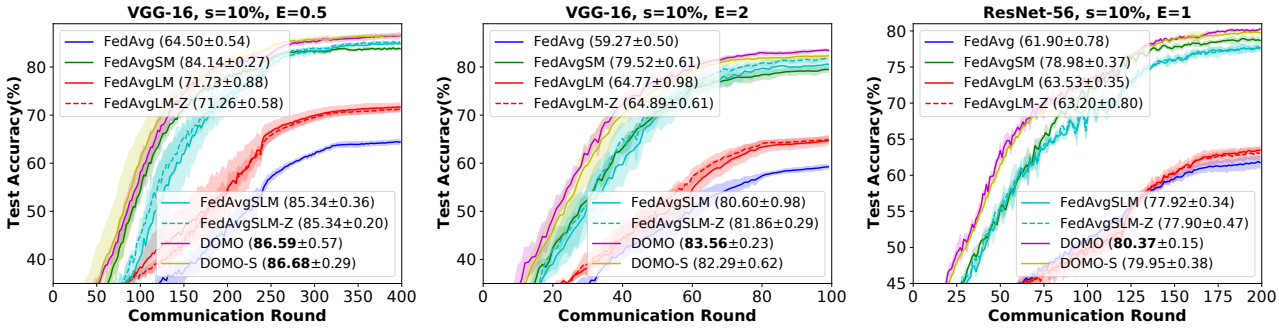

Figure 3: Left and Middle: CIFAR-10 training curves using the VGG-16 model with various local epoch E. $E = 1$ has been shown in the middle plot of Figure 1 and is not repeatedly shown here. Right: CIFAR-10 training curves using the ResNet-56 model.

β	$\alpha = 1.0$	β	$\alpha = 1.0$	α	$\beta = 0.9$	α	$\beta = 0.9$
1.0	81.89 ± 0.40	0.6	81.60 ± 0.40	1.0	$\mathbf{85.54} \pm 0.11$	0.6	83.76 ± 0.28
0.9	$\mathbf{85.54} \pm 0.11$	0.4	77.80 ± 0.88	0.9	84.63 ± 0.64	0.4	82.08 ± 0.50
0.8	83.84 ± 0.56	0.2	74.54 ± 0.49	0.8	84.83 ± 0.56	0.2	77.58 ± 0.62

Table 1: CIFAR-10 test accuracy (%) when training VGG-16 using DOMO with various hyper-parameters α and β. Data similarity $s = 10\%$ and local epoch $E = 1$. α is fixed at 1.0 with various β in the first column, while β is fixed at 0.9 with various α in the second column.

$\mathcal{O}(K^{-\frac{1}{2}}R^{-\frac{1}{2}}P^{-\frac{1}{2}})$ regarding iteration complexity, which achieves a linear speedup regarding the number of clients K. DOMO also has a communication complexity of $\frac{1}{P}$ when resetting local momentum buffer, which increases with a larger number of clients K but decreases with a larger communication rounds R. It becomes $\frac{2}{P}$ for averaging local momentum buffer, but 2 is a constant and does not affect the theoretical complexity. Note that there is no μ_s in Theorem 1 because it is eliminated in Lemma 2 by setting $\beta = \mu_s$.

Experimental Results

Settings

All experiments are implemented using PyTorch (Paszke et al. 2019) and run on a cluster where each node is equipped with 4 Tesla P40 GPUs and 64 Intel(R) Xeon(R) CPU E5-2683 v4 cores @ 2.10GHz. We compare the following momentum-based FL methods: 1) FedAvg, 2) FedAvgSM (*i.e.*, server momentum SGD), 3) FedAvgLM (*i.e.*, local momentum SGD), 4) FedAvgLM-Z (*i.e.*, local momentum SGD with resetting local momentum buffer), 5) FedAvgSLM (*i.e.*, FedAvgSM + FedAvgLM), 6) FedAvgSLM-Z (*i.e.*, FedAvgSM + FedAvgLM-Z, Algorithm 1 Option III), 7) DOMO (*i.e.*, Algorithm 1 Option I), and 8) DOMO-S (*i.e.*, Algorithm 1 Option II). In particular, FedAvg, FedAvgSM, FedAvgLM-Z, FedAvgSLM-Z, DOMO and DOMO-S have the same communication cost. FedAvgLM and FedAvgSLM need $\times 2$ communication cost.

We perform careful hyper-parameters tuning for all methods. The local momentum constant μ_l is selected from {0.9, 0.8, 0.6, 0.4, 0.2}. We select the server momentum constant μ_s from {0.9, 0.6, 0.3}. The base learning rate is selected from $\{..., 4\times10^{-1}, 2\times10^{-1}, 1\times10^{-1}, 5\times10^{-2}, 1\times10^{-2}, 5\times10^{-3}, ...\}$. The server learning rate α is selected from $\{0.2, 0.4, 0.6, 0.8, 0.9, 1.0\}$. The momentum fusion constant β is selected from $\{0.2, 0.4, 0.6, 0.8, 0.9, 1.0\}$. Following (Karimireddy et al. 2020b; Hsu, Qi, and Brown 2019; Wang et al. 2020), we use local epoch E instead of local training steps P in experiments. $E = 1$ is identical to one pass training of local data. We test local epoch $E \in \{0.5, 1, 2\}$ and $E = 1$ by default.

Data Similarity s. We follow (Karimireddy et al. 2020b) to simulate the *non-i.i.d.* data distribution. Specifically, fraction s of the data are randomly selected and allocated to clients, while the remaining fraction $1 - s$ are allocated by sorting according to the label. The data similarity is hence s. We run experiments with data similarity s in {5%, 10%, 20%}. By default, the data similarity is set to 10% and the number of clients (GPUs) $K = 16$ following (Wang et al. 2020). For all experiments, We report the mean and standard deviation metrics in the form of $(mean \pm std)$ over 3 runs with different random seeds for allocating data to clients.

Dataset. We train VGG-16 (Simonyan and Zisserman 2014) and ResNet-56 (He et al. 2016) models on CIFAR-10/100[1] (Krizhevsky 2009), and ResNet-20 on SVHN[2] image classification tasks. Please refer to the Supplementary Material for details.

Performance

We illustrate the experimental results in Figures 1, 2, and 3 with test accuracy $(mean \pm std)$ reported in the brackets

[1]https://www.cs.toronto.edu/ kriz/cifar.html
[2]http://ufldl.stanford.edu/housenumbers/

FedAvg	FedAvgSM	FedAvgLM(-Z)	FedAvgSLM(-Z)	DOMO-S	DOMO
87.79 ± 0.72	88.81 ± 0.49	88.86 ± 0.19 (87.93 ± 0.98)	88.67 ± 0.32 (88.89 ± 0.62)	**90.45** ± 0.56	90.34 ± 0.83

Table 2: SVHN test accuracy (%) when training ResNet-20.

FedAvg	FedAvgSM	FedAvgLM(-Z)	FedAvgSLM(-Z)	DOMO-S	DOMO
20.77 ± 1.31	35.14 ± 1.70	38.29 ± 1.01 (35.45 ± 2.04)	60.01 ± 0.28 (57.89 ± 2.79)	61.69 ± 0.41	**62.47** ± 0.73
39.61 ± 0.66	61.92 ± 0.43	46.65 ± 1.38 (45.09 ± 0.26)	62.95 ± 0.51 (63.45 ± 0.61)	64.34 ± 0.59	**65.84** ± 0.30

Table 3: CIFAR-100 test accuracy (%). Second row: VGG-16. Third row: ResNet-56.

of the legend, and Table **??** and **??**. Testing performance is the main metric for comparison in FL because local training metrics become less meaningful with clients tending to overfit their local data during local training. In overall, **DOMO(-S) > FedAvgSLM(-Z) > FedAvgSM > FedAvgLM(-Z) > FedAvg** regarding the test accuracy. DOMO and DOMO-S consistently achieve the fastest empirical convergence rate and best test accuracy in all experiments. On the contrary, the initial convergence rate of FedAvgSLM(-Z) can even be worse than FedAvgSM. In particular, FedAvgSLM(-Z) can hurt the performance compared with FedAvgSM as shown in the right plot of Figure 3, possibly due to the theoretical difficulties without momentum fusion discussed in Section "Convergence of DOMO". Besides, using server statistics is much better than without it (FedAvgSM $\gg$ FedAvg and FedAvgSLM(-Z) $\gg$ FedAvgLM(-Z)), in accordance with (Hsu, Qi, and Brown 2019; Huo et al. 2020).

Varying Data Similarity s. We plot the training curves under different data similarity settings in Figure 1. We can see that the improvement of DOMO and DOMO-S over other momentum-based methods increases with the data similarity s decreasing. This property makes our proposed method favorable in FL where the data heterogeneity can be complicated. In particular, DOMO improves FedAvgSLM-Z, FedAvgSM, and FedAvg by **5.00%, 5.68%, and 23.14%** respectively regarding the test accuracy when $s = 5\%$. When $s = 10\%$ and $s = 20\%$, DOMO improves over the best counterpart by **2.13%** and **1.02%**, while DOMO-S improves by **1.41%** and **0.85%** respectively.

Varying the Server and Local Momentum Constant μ_s, μ_l. We explore the various combinations of server and local momentum constant μ_s and μ_l of DOMO and report the test accuracy in Figure 2. $\mu_s = 0.9$ and $\mu_l = 0.6$ work best regardless of the data similarity s and the algorithm we use. Deviating from $\mu_s = 0.9$ and $\mu_l = 0.6$ leads to gradually lower test accuracy.

Varying the Local Epoch E. We plot the training curves of VGG-16 under different local epoch E settings in Figure 3 with data similarity $s = 10\%$. The number of local training steps $P = 49$ and 196 respectively when $E = 0.5$ and 2. We can see that DOMO improves the test accuracy over the best counterpart by **1.25%** and **1.70%** when $E = 0.5$ and 2 respectively.

Varying Hyper-parameters α and β. We explore the combinations of hyper-parameters α and β and report the corresponding test accuracy in Table **??**. $\alpha = 1.0$ and $\beta = 0.9$ work best.

Varying Model. We also plot the training curves of ResNet-56 in the right plot of Figure 3 which exhibit a similar pattern. DOMO improves the best counterpart by **1.35%** when data similarity $s = 10\%$ and local epoch $E = 1$. FedAvgSLM(-Z) is inferior to FedAvgSM, implying that a naive combination of FedAvgSM and FedAvgLM can hurt the performance. In contrast, DOMO and DOMO-S improve FedAvgSLM by **2.45%** and **2.03%** respectively.

Varying Dataset. The SVHN test accuracy is summarized in Table **??** and we can see that DOMO and DOMO-S improve the counterpart by **1.45%** and **1.56%** respectively. The CIFAR-100 test accuracy is summarized in Table **??** and we can see that DOMO and DOMO-S improve the best counterpart by **2.46%** and **1.68%** respectively when training VGG-16. They improve the counterpart by **2.39%** and **0.89%** respectively when training ResNet-56.

Conclusion

In this work, we proposed a new double momentum SGD (DOMO) method with a novel momentum fusion technique to improve the state-of-the-art momentum-based FL algorithm. We provided new insights for the connection between the server and local momentum with new concepts of pre-momentum, intra-momentum, and post-momentum. We also derived the first convergence analysis involving both the server and local Polyak's momentum SGD and discussed the difficulties of theoretical analysis in previous methods that are addressed by DOMO. From a theoretical perspective, we showed that momentum fusion in DOMO could lead to a tighter inconsistency bound. Future works may include incorporating the inter-client variance reduction technique to tighten the divergence bound as well. Deep FL experimental results on benchmark datasets verify the effectiveness of DOMO. DOMO can achieve an improvement of up to 5% regarding the test accuracy compared with the state-of-the-art momentum-based methods when training VGG-16 on CIFAR-10.

References

Acar, D. A. E.; Zhao, Y.; Matas, R.; Mattina, M.; Whatmough, P.; and Saligrama, V. 2020. Federated learning based on dynamic regularization. In *International Conference on Learning Representations*.

Bonawitz, K.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.; Konečnỳ, J.; Mazzocchi, S.; McMahan, H. B.; et al. 2019. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*.

Chen, K.; and Huo, Q. 2016. Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering. In *2016 ieee international conference on acoustics, speech and signal processing (icassp)*, 5880–5884. IEEE.

Cutkosky, A.; and Orabona, F. 2019. Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, 15236–15245.

Defazio, A.; Bach, F.; and Lacoste-Julien, S. 2014. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27: 1646–1654.

Defazio, A.; and Bottou, L. 2018. On the ineffectiveness of variance reduced optimization for deep learning. *arXiv preprint arXiv:1812.04529*.

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

Fallah, A.; Mokhtari, A.; and Ozdaglar, A. 2020. Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach. *Advances in Neural Information Processing Systems*, 33.

Gao, H.; Xu, A.; and Huang, H. 2021. On the Convergence of Communication-Efficient Local SGD for Federated Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35.

Gu, B.; Xu, A.; Huo, Z.; Deng, C.; and Huang, H. 2021. Privacy-Preserving Asynchronous Vertical Federated Learning Algorithms for Multiparty Collaborative Learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Guo, P.; Wang, P.; Zhou, J.; Jiang, S.; and Patel, V. M. 2021. Multi-institutional collaborations for improving deep learning-based magnetic resonance image reconstruction using federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2423–2432.

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

Hsieh, K.; Phanishayee, A.; Mutlu, O.; and Gibbons, P. 2020. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, 4387–4398. PMLR.

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

Huo, Z.; Yang, Q.; Gu, B.; Huang, L. C.; et al. 2020. Faster on-device training using new federated momentum algorithm. *arXiv preprint arXiv:2002.02090*.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. PMLR.

Jiang, Y.; Konečnỳ, J.; Rush, K.; and Kannan, S. 2019. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*.

Johnson, R.; and Zhang, T. 2013. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26: 315–323.

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

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.

Karimireddy, S. P.; Rebjock, Q.; Stich, S.; and Jaggi, M. 2019. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, 3252–3261. PMLR.

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

Konečnỳ, J.; McMahan, H. B.; Yu, F. X.; Richtárik, P.; Suresh, A. T.; and Bacon, D. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images. *https://www.cs.toronto.edu/ kriz/learning-features-2009-TR.pdf*.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2021. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, 6357–6368. PMLR.

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

Li, T.; Sanjabi, M.; Beirami, A.; and Smith, V. 2020. Fair Resource Allocation in Federated Learning. In *International Conference on Learning Representations*.

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

Liu, Q.; Chen, C.; Qin, J.; Dou, Q.; and Heng, P.-A. 2021. Feddg: Federated domain generalization on medical image segmentation via episodic learning in continuous frequency

space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1013–1023.

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.

Mohri, M.; Sivek, G.; and Suresh, A. T. 2019. Agnostic Federated Learning. In *International Conference on Machine Learning*, 4615–4625.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32: 8026–8037.

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*.

Rothchild, D.; Panda, A.; Ullah, E.; Ivkin, N.; Stoica, I.; Braverman, V.; Gonzalez, J.; and Arora, R. 2020. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, 8253–8265. PMLR.

Seide, F.; and Agarwal, A. 2016. CNTK: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2135–2135.

Shamsian, A.; Navon, A.; Fetaya, E.; and Chechik, G. 2021. Personalized Federated Learning using Hypernetworks. *arXiv preprint arXiv:2103.04628*.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958.

Stich, S. U. 2018. Local SGD Converges Fast and Communicates Little. In *International Conference on Learning Representations*.

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

T Dinh, C.; Tran, N.; and Nguyen, T. D. 2020. Personalized Federated Learning with Moreau Envelopes. *Advances in Neural Information Processing Systems*, 33.

Wang, J.; and Joshi, G. 2018. Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD. *arXiv preprint arXiv:1810.08313*.

Wang, J.; Liu, Q.; Liang, H.; Joshi, G.; and Poor, H. V. 2020. Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization. In *Advances in Neural Information Processing Systems*, volume 33, 7611–7623.

Wang, J.; Tantia, V.; Ballas, N.; and Rabbat, M. 2019. SlowMo: Improving Communication-Efficient Distributed SGD with Slow Momentum. In *International Conference on Learning Representations*.

Xu, A.; Huo, Z.; and Huang, H. 2020a. On the acceleration of deep learning model parallelism with staleness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2088–2097.

Xu, A.; Huo, Z.; and Huang, H. 2020b. Optimal gradient quantization condition for communication-efficient distributed training. *arXiv preprint arXiv:2002.11082*.

Xu, A.; Huo, Z.; and Huang, H. 2021. Step-Ahead Error Feedback for Distributed Training with Compressed Gradient.

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.

Zaheer, M.; Reddi, S.; Sachan, D.; Kale, S.; and Kumar, S. 2018. Adaptive methods for nonconvex optimization. In *Advances in neural information processing systems*, 9793–9803.