# New Efficient and Privacy-Preserving Methods for Distributed Training

by

**An Xu**

Bachelor of Engineering, Tsinghua University, 2017

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

**Doctor of Philosophy**

University of Pittsburgh

2023

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This dissertation was presented

by

An Xu

It was defended on

Jul 14, 2023

and approved by

Ahmed Dalla, PhD, Professor Department of Electrical and Computer Engineering

Masoud Barati, PhD, Professor, Department of Electrical and Computer Engineering

Zhi-Hong Mao, PhD, Professor, Department of Electrical and Computer Engineering

Mingui Sun, PhD, Professor, Department of Neurological Surgery, School of Medicine

Dissertation Director: Bo Zeng, PhD, Professor, Department of Electrical and Computer Engineering

**New Efficient and Privacy-Preserving Methods for Distributed Training**

An Xu, PhD

University of Pittsburgh, 2023

The distributed training of deep learning models faces two issues: efficiency and privacy. First of all, training models can be slow and inefficient, especially when it is large with data distributed across multiple devices. For model parallelism, the inefficiency is caused by the backpropagation algorithm's forward locking, backward locking, and update locking problems. Existing solutions for acceleration either can only handle one locking problem or lead to severe accuracy loss or memory inefficiency. Moreover, none of them consider the straggler problem among devices. We propose Layer-wise Staleness and a novel efficient training algorithm, Diversely Stale Parameters (DSP), to address these challenges.

For data parallelism, the communication bottleneck has been a critical problem in large-scale distributed deep learning. We study distributed SGD with random block-wise sparsification as the gradient compressor, which is ring-allreduce compatible and highly computation-efficient but leads to inferior performance. To tackle this important issue, we propose a new detached error feedback (DEF) algorithm, which shows better convergence bound than error feedback for non-convex problems.

Secondly, distributed training raises concerns of data privacy when user's data is gathered to a central server. To keep data privacy, cross-silo federated learning (FL) has attracted much attention. However, there can be a generalization gap between the model trained from FL and the one from centralized training. We propose a novel training framework FedSM to avoid the client drift issue and successfully close the generalization gap compared with the centralized training for medical image segmentation tasks for the first time.

Communication efficiency is also crucial for federated learning (FL). Conducting local training steps in clients to reduce the communication frequency 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. We propose a new method to improve the training performance via maintaining double momentum buffers.

## Table of Contents

## List of Tables

## List of Figures

## Preface

I would like to express my sincere gratitude to Dr. Bo Zeng (supervisor), Dr. Zhi-Hong Mao, Dr. Ahmed Dalla, Dr. Masoud Barati, and Dr. Minghui Sun for their invaluable contributions as members of my committee and their guidance throughout my research work. Additionally, I am deeply grateful to Dr. Heng Huang for providing unwavering support during my first four years of PhD study. I would also like to extend my thanks to my colleagues at school, as well as the collaborators from NVIDIA Research and Amazon Web Services, with whom I had the privilege to work during my internships.

The path of pursuing a PhD is filled with intellectual and mental challenges, and I firmly believe that I would not have been able to overcome them without the unwavering support of my beloved parents Honghua Xu and Xiaoping Xu, and my girlfriend Yang Bai. Their love and care provided me with the strength to navigate the stress of the pandemic and persevere.

The completion of my PhD degree is undeniably a significant milestone in my personal journey. In fact, it has already proven to be an immensely rewarding experience. I am profoundly grateful for the incredible individuals I have had the privilege of crossing paths with throughout my life.

# 1.0 Improve the Efficiency of Model Parallelism

## 1.1 Introduction

The deep convolutional neural network is an important method for solving computer vision problems such as classification, object detection, etc. However, as the neural networks get deeper and larger [47, 62, 52, 128, 144, 95], the required expensive training time has become the bottleneck. Data parallelism [134, 83, 19] and model parallelism [79, 75] are two standard parallelism techniques to utilize multiple devices for efficient training.

The data parallelism for efficient distributed training has been well studied and implemented in existing libraries [5, 22, 56, 148, 58, 60], but the model parallelism is still underexplored. In this paper, we focus on the model parallelism, where the deep neural network (DNN) benefits from being split onto multiple devices. But the resource utilization of standard model parallelism can be very low. The backpropagation algorithm [114, 78] typically requires two phases to update the model in each training step: the forward pass and backward pass. But the sequential propagation of activation and error gradient leads to *backward locking* and *forward locking* [64] respectively because of the computation dependencies between layers. The *update locking* [64] exists as the backward pass will not start until the forward pass has completed. This sequential execution keeps a device inefficiently waiting for the activation input and error gradient.

Several works have been proposed to address these locking issues (Figure 1). [64] uses Decoupled Neural Interfaces (DNI) to predict the error gradient via auxiliary networks, so that a layer uses the synthetic gradient and needs not to wait for the error gradient. [104] lets hidden layers receive error information directly from the output layer. However, these methods can not converge when dealing with very deep neural networks. [15] proposes layer-wise decoupled greedy learning (DGL), which introduces an auxiliary classifier for each block of layers so that a block updates its parameters according to its own classifier. But the objective function of DGL based on greedy local predictions can be very different from the original model. GPipe [54] proposes pipeline parallelism and divides each mini-batch

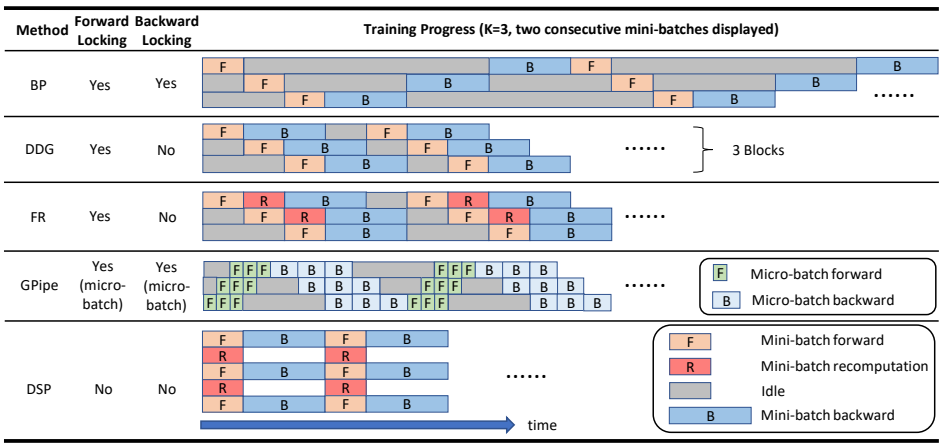


Figure 1: Sketches of different methods with three blocks. The forward and recomputation are overlapped in DSP.

into micro-batches, which can be regarded as a combination of model parallelism and data parallelism. However, the forward and backward lockings of the micro-batch still exist, and the update locking is not addressed because GPipe waits for the whole forward and backward pass to finish before updating the parameters. [59] proposes Decoupled Parallel Backpropagation (DDG), which divides the DNN into blocks and removes the backward locking by storing delayed error gradient and intermediate activations at each block. But DDG suffers from large memory consumption due to storing all the intermediate results. PipeDream [101] has to store multiple versions of weights in addition to intermediate activations as in DDG. Features Replay (FR) [57, 152] improves DDG via storing the history inputs and recomputing the intermediate results. Nevertheless, blocks in DDG and FR still need to wait for the backward error gradient. Besides, neither DDG nor FR addresses the forward locking problem.

To overcome the aforementioned drawbacks, we first propose *Layer-wise Staleness*, a fine-grained staleness within the model to allow different parts to be trained independently. Incorporating staleness is useful for efficient asynchronous execution without synchronization barrier [49], which can be interpreted as another form of locking/dependency. The introduction of preset Layer-wise Staleness enables each part of the convolutional neural network

(CNN) to run in a very flexible way with a certain degree of asynchrony. Based on the concept of Layer-wise Staleness, we propose a novel parallel CNN training algorithm named as Diversely Stale Parameters (DSP), where lower layers use more stale information to update parameters. DSP also utilizes the recomputation technique [23, 40] to reduce memory consumption, which is overlapped with the forward pass. Our contributions are summarized as follows:

- We propose Layer-wise Staleness and Diversely Stale Parameters which breaks the forward, backward and update lockings without memory issues.
- To ensure the theoretical guarantee, we provide convergence analysis for the proposed method. Even faced with parameters of different Layer-wise Staleness, we prove that DSP converges to critical points for non-convex problems with SGD and momentum SGD.
- We evaluate our method via training deep convolutional neural networks. Extensive empirical results show that DSP achieves significant training speedup and strong robustness against random stragglers.

### 1.2 Background

We divide a CNN into $K$ consecutive blocks so that the whole parameters

$$x = (x_0, x_1, ..., x_{K-1}) \in \mathbb{R}^d, \tag{1–1}$$

where $x_k \in \mathbb{R}^{d_k}$ denotes the partial parameters at block $k \in \{0, 1, ..., K-1\}$ and $d = \sum_{k=0}^{K-1} d_k$. Each block $k$ computes activation $h_{k+1} = f_k(h_k; x_k)$, where $h_k$ denotes the input of block $k$. In particular, $h_0$ is the input data. For simplicity, we define $F(h_0; x_0; x_1; ...; x_k) \coloneqq f_k(...f_1(f_0(h_0; x_0); x_1)...; x_k) = h_{k+1}$. The loss is $\mathcal{L}(h_K, l)$, where $l$ is the label. Minimizing the loss of a $K$-block neural network can be represented by the following problem:

$$\min_{x \in \mathbb{R}^d} \quad f(x) \coloneqq \mathcal{L}(F(h_0; x_0; x_1; ...; x_{K-1}), l). \tag{1–2}$$

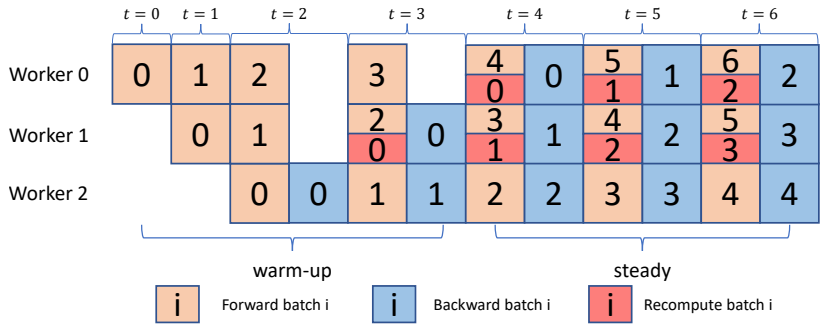


Figure 2: DSP(1,1,0;4,2,0) with Layer-wise Staleness of {4,2,0} (the index difference between the forward and backward batch). Worker $k \in \{0, 1, 2\}$ holds block $k$.

Backpropagation algorithm computes the gradient for block $k$ following chain rule via Eq. (1–3). The forward locking exists because the input of each block is dependent on the output from the lower block. The backward locking exists because each block cannot compute gradients until having received the error gradient $\mathcal{G}_h$ from the upper block. Besides, the backward process can not start until the whole forward process is completed, which is known as the update locking.

$$\begin{cases} \mathcal{G}_{h_k} = \frac{\partial f_k(h_k;x_k)}{\partial h_k}\mathcal{G}_{h_{k+1}}, \quad \mathcal{G}_{h_K} = \frac{\partial \mathcal{L}(h_K,l)}{\partial h_K} \\ \mathcal{G}_{x_k} = \frac{\partial f_k(h_k;x_k)}{\partial x_k}\mathcal{G}_{h_{k+1}}. \end{cases} \tag{1–3}$$

After computing the gradients, stochastic gradient descent (SGD) [111] and its variants such as stochastic unified momentum (SUM) [153], RMSPROP [133] and ADAM [72] are widely used for updating the model. SGD updates via $x^{n+1} = x^n - \alpha\mathcal{G}(x^n;\xi)$, where $x^n$ is the parameters when feeding the $n^{\text{th}}$ data (batch), $\alpha$ is the learning rate, and $\mathcal{G}(x^n;\xi)$ is the stochastic gradient. SUM updates the parameters via Eq. (1–4), where $\beta$ is the momentum constant and $y$ is the momentum term. When $s = 1$, SUM reduces to stochastic Nesterov's accelerated gradient (SNAG) [102].

$$\begin{cases} y^{n+1} = x^n - \alpha\mathcal{G}(x^n;\xi), \ y^{s,n+1} = x^n - s\alpha\mathcal{G}(x^n;\xi) \\ x^{n+1} = y^{n+1} + \beta(y^{s,n+1} - y^{s,n}). \end{cases} \tag{1–4}$$

## 1.3 Diversely Stale Parameters

In this section, we propose a novel training method named Diversely Stale Parameters (Figure 2). We first define layer-wise staleness and related notations in Section 1.3.1, then the motivation and formulation of DSP gradient in Section 1.3.2, finally the practical implementation using queues for pipelined batch input in Section 1.3.3.

### 1.3.1 Layer-Wise Staleness

Let the data be forwarded with parameters $x_0$ at timestamp $t_0$, $x_1$ at timestamp $t_1$, ..., and $x_{K-1}$ at timestamp $t_{K-1}$. For simplicity we denote the **Forward Parameters** as $\{x_k^{t_k}\}_{k=0,\dots,K-1}$. Similarly we denote the **Backward Parameters** as $\{x_k^{t_{2K-1-k}}\}_{k=0,\dots,K-1}$. Then we define **Layer-wise Staleness** as $\Delta t_k = t_{2K-k-1} - t_k \geq 0$. We preset each block's Layer-wise Staleness to a different value to break the synchronization barrier of backpropagation.

We also denote the maximum Layer-wise Staleness as $\Delta t = \max_{k=0,1,\dots,K-1} \Delta t_k$. It is worth noting that a) in standard backpropagation algorithm (Eq. (1–3)), Layer-wise Staleness $\Delta t_k = 0$; and b) Feeding data index is not identical to timestamp/training step.

### 1.3.2 DSP Gradient

We first set the constraints of DSP as $t_0 < t_1 < \ldots < t_{K-1} \leq t_K < t_{K+1} < \ldots < t_{2K-1}$ such that both the dependencies in the forward and backward pass no longer exist, because we do not need them to finish in the same timestamp anymore. The non-decreasing property corresponds to the fact that the data needs to go through the bottom layers before the top layers, and the error gradient needs to go through the top layers before the bottom layers.

Based on backpropagation algorithm and Eq. (1–3), we should compute the gradients according to the following formulas as we are updating the Backward Parameters, which is

defined as $\{x_k^{t_{2K-1-k}}\}_{k=0,...,K-1}$,

$$\begin{aligned}
\mathcal{G}_{x_k} &= \frac{\partial F(h_0; x_0^{t_{2K-1}}; ...; x_k^{t_{2K-1-k}})}{\partial x_k^{t_{2K-1-k}}} \mathcal{G}_{h_{k+1}} \\
\mathcal{G}_{h_k} &= \frac{\partial F(h_0; x_0^{t_{2K-1}}; ...; x_k^{t_{2K-1-k}})}{\partial F(h_0; x_0^{t_{2K-1}}; ...; x_{k-1}^{t_{2K-2-k}})} \mathcal{G}_{h_{k+1}} \\
\mathcal{G}_{h_K} &= \frac{\partial \mathcal{L}(F(h_0; x_0^{t_{2K-1}}; ...; x_{K-1}^{t_K}), l)}{F(h_0; x_0^{t_{2K-1}}; ...; x_{K-1}^{t_K})}.
\end{aligned} \tag{1–5}$$

However, during the forward pass the input of block $k$ is $F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}})$. Therefore we incorporate the recomputation technique and utilize both the Forward Parameters and Backward Parameters to compute DSP gradient as follows,

$$\begin{aligned}
\mathcal{G}_{x_k} &= \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; \boldsymbol{x_k^{t_{2K-1-k}}})}{\partial \boldsymbol{x_k^{t_{2K-1-k}}}} \mathcal{G}_{h_{k+1}} \\
\mathcal{G}_{h_k} &= \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; \boldsymbol{x_k^{t_{2K-1-k}}})}{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}})} \mathcal{G}_{h_{k+1}} \\
\mathcal{G}_{h_K} &= \frac{\partial \mathcal{L}(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l)}{F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})}.
\end{aligned} \tag{1–6}$$

The intuition behind the DSP gradient of Eq. (1–6) is that it is equivalent to Eq. (1–5) when the model converges to a local optimum where the gradient is zero ($x_k^{t_k} = x_k^{t_{2K-1-k}}$ afterwards).

### 1.3.3 Batch Pipeline Input

The computation of the DSP gradient breaks the forward and backward dependencies/lockings of the same data as it will not appear in different blocks at the same timestamp. The update locking is naturally broken.

For the parallel implementation of DSP as shown in Figure 2, we incorporate the data batch pipeline to keep all the blocks being fed with different data batches and running. The data source consecutively feeds data input. Different blocks transport and process different data via FIFO queues. As a result, the data travels each block at different timestamps. Specifically, each block $k$ maintains an input queue $\mathcal{M}_k$, output queue $\mathcal{P}_k$ and

Table 1: Best Top-1 Test Accuracy

| | | ResNet164 | | ResNet98 | |
|---|---|---|---|---|---|
| | | CIFAR-10 | CIFAR-100 | CIFAR-10 | CIFAR-100 |
| | BP | 94.41% | 75.66% | 93.38% | 72.66% |
| K=3 | FR | 94.55% | 76.25% | 93.60% | 73.27% |
| | DSP(1,1,0;4,2,0) | **94.68%** | 76.05% | 93.36% | 72.99% |
| | DSP(2,2,0;6,3,0) | 93.98% | 76.00% | **93.68%** | **73.70%** |
| | DSP(3,3,0;10,5,0) | 93.37% | **76.29%** | 93.27% | 73.38% |
| K=4 | FR | 94.44% | 75.84% | 93.26% | 72.41% |
| | DSP(1,1,1,0;6,4,2,0) | 94.32% | **76.22%** | 93.41% | **73.14%** |
| | DSP(2,2,2,0;9,6,3,0) | **94.87%** | 75.59% | 93.06% | 72.89% |
| | DSP(3,3,3,0;15,10,5,0) | 93.34% | 75.15% | **93.45%** | 72.96% |

gradient queue $\mathcal{Q}_k$ of length $1 + m_k$, $1 + p_k$ and $1 + q_k$ respectively. We denote it as $DSP(p_0, ..., p_{K-1}; m_0, ..., m_{K-1})$. $\{q_k\}$ is determined by $\{p_k\}$ and $\{m_k\}$ because the input should match the corresponding error gradient. We manually split the model to different workers to balance the workload at the steady stage.

Apart from adopting recomputation to reduce memory consumption, DSP overlaps recomputation with the forward pass to save time. Using queues also make DSP overlap the communication between blocks with computation. The FIFO queues allow for some asynchrony which is effective for dealing with random stragglers. The ideal time complexity of DSP is $\mathcal{O}(\frac{T_F+T_B}{K})$ and the space complexity is $\mathcal{O}(L+\sum_{k=0}^{K-1}(m_k+p_k+q_k))$, where $T_F$ and $T_B$ are serial forward and backward time, and $L$ is the number of layers. $m_k$ also represents the Layer-wise Staleness $\Delta t_k$ of block $k$. $K$ and the FIFO queues length $m_k+1, p_k+1, q_k+1 \ll L$ for deep models, so the extra space cost is trivial.

### 1.4 Convergence Analysis

The convergence of DSP with SGD is first analyzed, then DSP with Momentum SGD. For simplicity, we denote the Forward and Backward Parameters of data $n$ as $x^{n'}$ and $x^n$

Table 2: Robustness (ResNet164, CIFAR-10, K=3). Each GPU is randomly slowed down.

| | Slow down percentage | | | |
|---|---|---|---|---|
| GPU | 20% | 50% | 100% | 150% |
| FR | 8.977% | 28.52% | 97.06% | 359.2% |
| DSP(1,1,0;4,2,0) | **6.017%** | 16.14% | 37.44% | 70.99% |
| DSP(2,2,0;6,3,0) | 7.465% | **16.01%** | 36.57% | 54.57% |
| DSP(3,3,0;10,5,0) | 7.391% | 18.15% | **32.10%** | **53.42%** |

respectively.

**Assumption 1.4.1.** *(**Bounded variance**) Assume that the DSP stochastic gradient* $\mathcal{G}(x;\xi)$ *satisfies* $Var\left[\mathcal{G}(x;\xi)\right] \leq \sigma^2$. *Note* $\mathbb{E}\left[\mathcal{G}(x;\xi)\right] = \mathcal{G}(x) \neq \nabla f(x)$.

**Assumption 1.4.2.** *(**Lipschitz continuous gradient**) Assume that the loss and the output of the blocks have Lipschitz continuous gradient, that is,* $\forall k \in \{0,1,..,K-1\}$, *and* $\forall (x_{0,1},...,x_{k,1}), (x_{0,2},...,x_{k,2}) \in \mathbb{R}^{d_0+d_1+...+d_k}$, *we have*

$$\left\|\nabla F\left(h_0; x_{0,1}; ...; x_{k,1}\right) - \nabla F(h_0; x_{0,2}; ...; x_{k,2})\right\| \leq L_k \left\|(x_{0,1},...,x_{k,1}) - (x_{0,2},...,x_{k,2})\right\|, \quad (1\text{–}7)$$

*and* $\forall x_1,\ x_2 \in \mathbb{R}^d$,

$$\left\|\nabla f(x_1) - \nabla f(x_2)\right\| \leq L_K \left\|x_1 - x_2\right\|. \quad (1\text{–}8)$$

We define $L \coloneqq \max_{k\in\{0,1,...,K\}} L_k$. Note $\nabla F\left(h_0; x_{0,1}; ...; x_{k,1}\right)$ and $\nabla F(h_0; x_{0,2}; ...; x_{k,2})$ regarding parameters are Jacobian matrices. In fact, this is assuming that the partial model consisted of the blocks that the data has traveled, has Lipschitz continuous gradient.

**Assumption 1.4.3.** *(**Bounded error gradient**) Assume that the norm of the error gradient that a block receives is bounded, that is, for any* $x \in \mathbb{R}^d$, $\forall k \in \{0,1,...,K-2\}$, *we have* $\left\|\frac{\partial f_{k+1}(h_{k+1};x_{k+1})}{\partial h_{k+1}} ... \frac{\partial f_{K-1}(h_{K-1};x_{K-1})}{\partial h_{K-1}} \frac{\partial \mathcal{L}(h_K,l)}{\partial h_K}\right\| \leq M \quad and \quad \left\|\frac{\partial \mathcal{L}(h_K,l)}{\partial h_K}\right\| \leq M$.

This is assuming that the error gradient at each block does not explode. It is natural to make the above two block-wise assumptions as we are breaking the neural networks into blocks.

Table 3: Speedup Comparison Results.

| | CIFAR-10 | | | CIFAR-100 | ImageNet | |
|---|---|---|---|---|---|---|
| K, batch size | ResNet164 (4, 128) | ResNext-29 (4, 128) | VGG-19 (3, 128) | ResNet1001 (4, 128) | ResNet50 (3, 256) | ResNet101 (4, 128) |
| BP / BP-K | x1 / - | x1 / - | x1 / - | - / x1 | - / x1 | x1 / - |
| FR | x1.7 | x1.3 | x1.1 | x1.9 | x1.6 | x1.7 |
| GPipe | - | - | - | - | - | x2.2 |
| DSP | **x2.7** | **x2.4** | **x1.5** | **x4.8** | **x3.0** | **x2.7** |

**Lemma 1.4.1.** *If Assumptions 1.4.2 and 1.4.3 hold, the difference between DSP gradient and BP gradient regarding the parameters of block $k \in \{0, 1, ..., K-1\}$ satisfies*

$$\|\nabla_{x_k}\mathcal{L}(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), y) - \mathcal{G}_{x_k}(x_0^{t_{2K-1}}; ...; x_{K-1}^{t_K})\| \leq LM \sum_{i=k}^{K-1} \left\| x_i^{t_{2K-1-i}} - x_i^{t_i} \right\|. \quad (1\text{–}9)$$

### 1.4.1 DSP with SGD

**Theorem 1.4.1.** *Assume Assumptions 1.4.1, 1.4.2 and 1.4.3 hold. Let $c_0 = M^2K(K+1)^2$, and $c_1 = -(\Delta t^2 + 2) + \sqrt{(\Delta t^2+2)^2 + 2c_0\Delta t^2}$. If the learning rate $\alpha_n \leq \frac{c_1}{Lc_0\Delta t^2}$, then*

$$\frac{\sum_{n=0}^{N-1} \alpha_n \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2}{\sum_{n=0}^{N-1}\alpha_n} \leq \frac{2\left[f(x^0) - f^*\right]}{\sum_{n=0}^{N-1}\alpha_n} + \frac{L\sigma^2(2 + K\Delta t^2 + \frac{1}{4}Kc_1)\sum_{n=0}^{N-1}\alpha_n^2}{\sum_{n=0}^{N-1}\alpha_n}. \quad (1\text{–}10)$$

**Corollary 1.4.1.** *(Sublinear convergence rate) According to Theorem 1.4.1, by setting the learning rate $\alpha_n = \min\left\{\frac{1}{\sqrt{N}}, \frac{c_1}{Lc_0\Delta t^2}\right\}$, when $N$ is large enough we have $\alpha_n = \frac{1}{\sqrt{N}}$ and*

$$\min_{n=0,...,N-1} \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 \leq \frac{2(f(x^0) - f^*)}{\sqrt{N}} + \frac{L\sigma^2(2 + K\Delta t^2 + \frac{1}{4}Kc_1)}{\sqrt{N}}. \quad (1\text{–}11)$$

**Corollary 1.4.2.** *According to Theorem 1.4.1, if the learning rate $\alpha_n$ diminishes and satisfies the requirements in [111]: $\lim_{N\to\infty}\sum_{n=0}^{N-1}\alpha_n = \infty$ and $\lim_{N\to\infty}\sum_{n=0}^{N-1}\alpha_n^2 < \infty$, choose $x^n$ randomly from $\{x^n\}_{n=0}^{N-1}$ with probabilities proportional to $\{\alpha_n\}_{n=0}^{N-1}$. Then we can prove that it converges to critical points for the non-convex problem due to $\lim_{n\to\infty}\mathbb{E}\left\|\nabla f(x^n)\right\|^2 = 0$.*

Table 4: Best Top-1 Test Accuracy on ImageNet (K=3).

| Method | ResNet18 | ResNet50 |
|---|---|---|
| BP | 69.89% | 75.35% |
| FR | 68.94% | 74.47% |
| DSP(1,1,0;4,2,0) | 68.95% | 74.91% |

### 1.4.2 DSP with Momentum SGD

**Theorem 1.4.2.** *Assume Assumption 1.4.1, 1.4.2 and 1.4.3 hold. Let*

$$c_2 = \frac{((1-\beta)s-1)^2}{(1-\beta)^2}, \tag{1–12}$$

$$c_3 = M^2K(K+1)^2\Delta t^2(c_2+s^2), \tag{1–13}$$

$$c_4 = 3+\beta^2 c_2 + 2(1-\beta)^2\Delta t^2(c_2+s^2), \tag{1–14}$$

*and*

$$c_5 = \frac{2+\beta^2 c_2}{1-\beta} + 2(1-\beta)\Delta t^2(c_2+s^2) + \frac{-c_4+\sqrt{c_4^2+4(1-\beta)^2c_3}}{2(1-\beta)}. \tag{1–15}$$

*If the fixed learning rate $\alpha$ satisfies $\alpha \le \frac{-c_4+\sqrt{c_4^2+4(1-\beta)^2c_3}}{2(1-\beta)c_3L}$, then*

$$\frac{1}{N}\sum_{n=0}^{N-1}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 \le \frac{2(1-\beta)(f(x^0)-f^*)}{N\alpha} + c_5\sigma^2 L\alpha. \tag{1–16}$$

**Corollary 1.4.3.** *(Sublinear convergence rate) According to Theorem 1.4.2, by setting the learning rate $\alpha = \min\{\frac{1}{\sqrt{N}}, \frac{-c_4+\sqrt{c_4^2+4(1-\beta)^2c_3}}{2(1-\beta)c_3L}\}$, when $N$ is large enough we have $\alpha = \frac{1}{\sqrt{N}}$ and $\min_{n=0,\dots,N-1}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 \le \frac{2(1-\beta)(f(x^0)-f^*)}{\sqrt{N}} + \frac{c_5\sigma^2 L}{\sqrt{N}}$.*

**Remark 1.4.1.** *The convergence performance of DSP is affected by Layer-wise Staleness rather than the staleness between different blocks.*

## 1.5 Experiments

We implement DSP in TensorFlow [5] and run the experiments on Nvidia Tesla P40 GPUs. The model is divided into $K$ blocks and distributed onto $K$ GPUs. Data augmentation procedures include random cropping, random flipping, and standardization. We use SGD with the momentum constant of 0.9. In CIFAR experiments, the batch size is 128. We train ResNet98 and ResNet164 for 300 epochs. The weight decay is $5 \times 10^{-4}$ and the initial learning rate is 0.01 (test performance could be a little lower than 0.1 [96]) with a decay of 0.1 at epoch 150, 225; ResNet1001 is trained for 250 epochs. The weight decay is $2 \times 10^{-4}$ and the initial learning rate is 0.1 with a decay of 0.1 at epoch 100, 150, 200; VGG-19 and ResNext-29 are trained for 200 epochs. The weight decay is $5 \times 10^{-4}$ and the initial learning rate is 0.01 with a decay of 0.1 at epoch 100, 150. We also train ResNet on ImageNet for 90 epochs. The batch size is 256, the weight decay is $1 \times 10^{-4}$ and the initial learning rate is 0.1 with a decay of 0.1 at epoch 30, 60, 80. There are four compared methods:

- BP: The standard implementation in TensorFlow. BP (or BP-K) runs on one (or K) GPUs.
- DNI: The Decoupled Neural Interface algorithm in [64]. The auxiliary network consists of two hidden and one output convolution layers with $5 \times 5$ filters and padding size of 2. The hidden layers also use batch-normalization and ReLU.
- FR: The Features Replay algorithm proposed by [57].
- DSP: Our Diversely Stale Parameters.

### 1.5.1 Faster Training

The DSP convergence curves regarding training epochs are nearly the same as FR and BP, while DNI does not converge as shown in Figure 3. But the epoch time of DSP is much less. Due to the overlap of communication and computation, the overheads of DSP are much less than model parallel BP and the speedup can even exceed $K$. However, it is important that the model should be properly distributed onto different blocks such that the workload of each computing device is balanced. If not, the overall speed will be mostly

determined by the slowest device. To further demonstrate the scalability of DSP, we also run experiments on VGG-19 [120], ResNeXt-29 [144], ResNet1001 on the CIFAR dataset, and ResNet18 and ResNet50 on the ImageNet [28] dataset as shown in Figure 4 and Figure 5 respectively. The speedup is summarized in Table 3 (GPipe paper only reports speedup of ResNet101 and AmoebaNet-D (4,512)). Our proposed DSP improves the speedup compared with its counterparts from x0.5 to x3.1 based on different datasets, model and the value of K. Note that the implementation of DSP involves some inefficient copy operations due to limited supported features of the deep learning framework, which means that DSP could achieve a potentially even faster speedup.

### 1.5.2 Robustness

To show that DSP is more resilient to the straggle problem due to the FIFO queues introduced, we randomly slow down each GPU by a certain percentage with a probability of 1/3 and run the experiments on ResNet164 (Table 2). The performance of FR degrades a lot because it does not break the forward locking nor completely decouple the backward pass. In comparison, DSP is very robust with the best slow down percentage always less than 1/3 of the corresponding GPU slow down percentage. When the upper or lower block suddenly slows down, the current block's feeding data and gradient queues are less likely to be empty if the length of the queue is long. When the straggler effect is not serious, increasing the Layer-wise Staleness will not bring performance gain; when it is serious instead, DSP benefits a lot from increasing the Layer-wise Staleness. Generally speaking, longer queues improve DSP's resilience to random stragglers, which is shown in Table 2.

### 1.5.3 Generalization

Table 1 and Tabel 4 show the best top-1 test accuracy on the CIFAR and ImageNet dataset respectively. The test performance of DSP is better than BP and FR on the CIFAR dataset. From Lemma 1.4.1 we know that the DSP gradient deviates from the BP gradient due to the Layer-wise Staleness. This difference becomes small as the training proceeds but could impose small noise and help find a better local minimum on the comparatively less

complex CIFAR classification problem.

In comparison, on the ImageNet dataset, the Layer-wise Staleness can lead to performance degradation. By intuition, it is similar to asynchronous distributed training where the whole gradient is of the same staleness. But in DSP, the more fine-grained Layer-wise Staleness will impose different blocks with different staleness effects. Potential solutions could be using staleness-aware methods as proposed in asynchronous distributed training area, e.g. gradient compensation and staleness-aware learning rate, to alleviate the staleness effect. Another possible direction is to balance the staleness effect between all the blocks. Moreover, when compared with FR, DSP's test accuracy is slightly better. On ResNet18, the test accuracy of FR and DSP is very similar, but on ResNet50 there is a 0.44% gain using DSP. Besides, on the more complicated ResNet50 architecture, the performance degradation resulting from the staleness effect is smaller than that on ResNet18.

### 1.5.4 Gradient Difference

Here we attest our theoretical analysis of Lemma 1.4.1 via checking the difference between the DSP and the BP gradient on the CIFAR dataset with the ResNet164 model. From the top-left figure of Figure 4, we can see that the difference between the DSP and BP gradient drops very fast to the converged value as the training proceeds. This difference drops even faster for upper blocks where the Layer-wise Staleness effect is milder. It confirms the motivation behind the DSP algorithm that the DSP gradient will finally be similar to the BP gradient. Moreover, the lower blocks suffer from a larger difference. When the Layer-wise Staleness keeps increasing, the difference will also increase, which matches Lemma 1.4.1 well. Moreover, as the learning rate drops, the difference between the DSP gradient and the BP gradient will drop a lot. This implies that a smaller learning rate should be used when we need to deal with a larger number of blocks where the Layer-wise Staleness effect becomes non-trivial. This is also shown in Theorem 1.4.1 and 1.4.2 that the learning rate should be decreased to make sure it converges at the stated speed.

## 1.6 Conclusion

In this paper, we have proposed Layer-wise Staleness and DSP, a novel way to fast train neural networks. DSP is proved to converge to critical points for non-convex problems with SGD and Momentum SGD optimizer. We apply DSP to train CNNs in parallel and the experiment results confirm our theoretical analysis. Our proposed method achieves significant training speedup, strong resilience to random stragglers, better generalization on the CIFAR dataset and reasonable performance on the ImageNet dataset. The speedup can exceed $K$ compared with the model parallel BP. Potential future works include how to alleviate the staleness effect when we need to utilize a further larger number of blocks; how to automatically determine the proper model splitting strategy for load balance among devices; efficiently incorporating DSP with data parallelism to achieve even faster training speed.

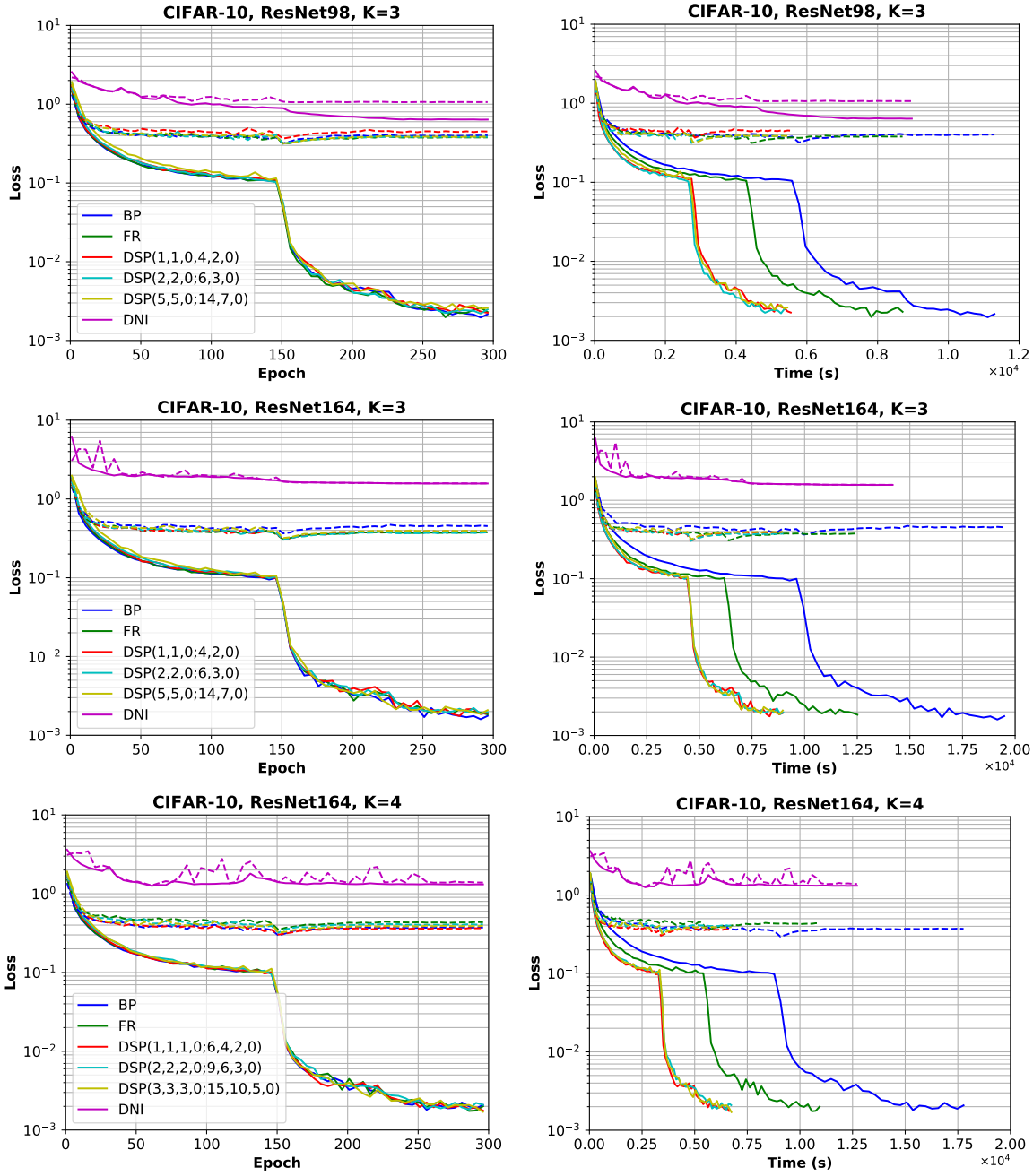


Figure 3: Training loss (solid line) and testing loss (dash line) for ResNet98, ResNet164 on CIFAR-10. The first row and second row plots the loss regarding the training epochs and time respectively.

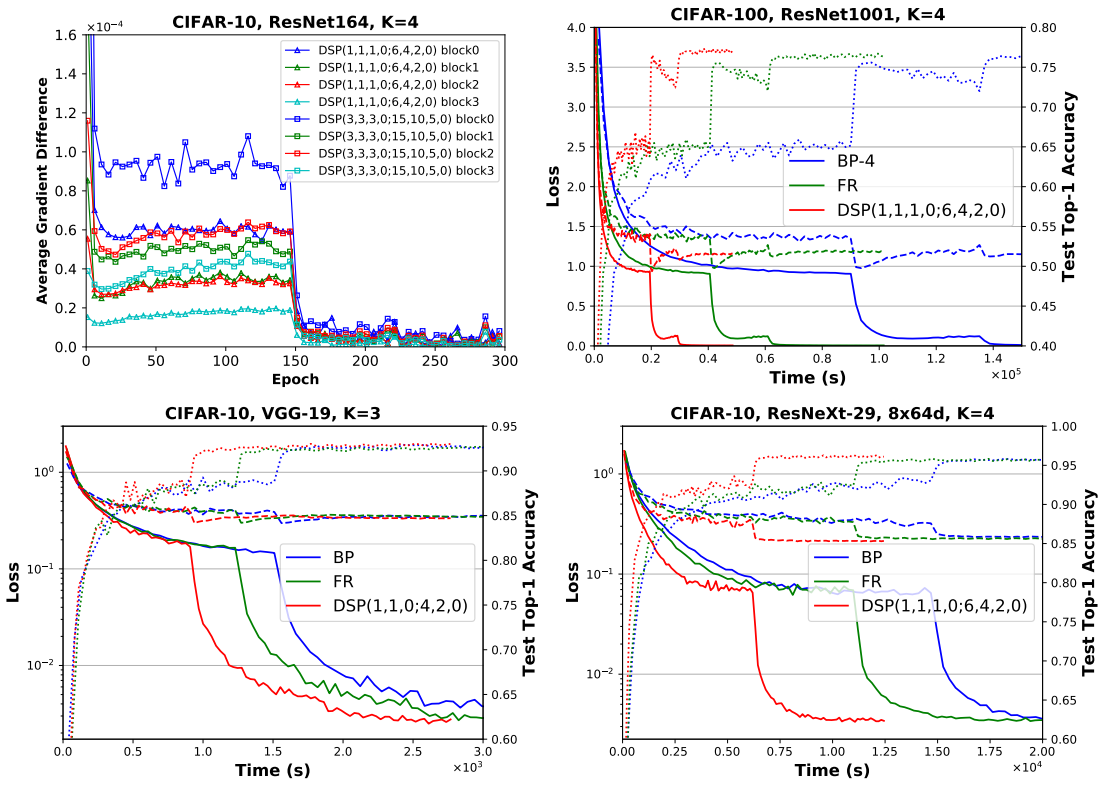


Figure 4: Top left: Average difference of DSP and BP gradient regarding the number of parameters. The rest: Training loss (solid line), testing loss (dash line) and test top-1 accuracy(dot line).

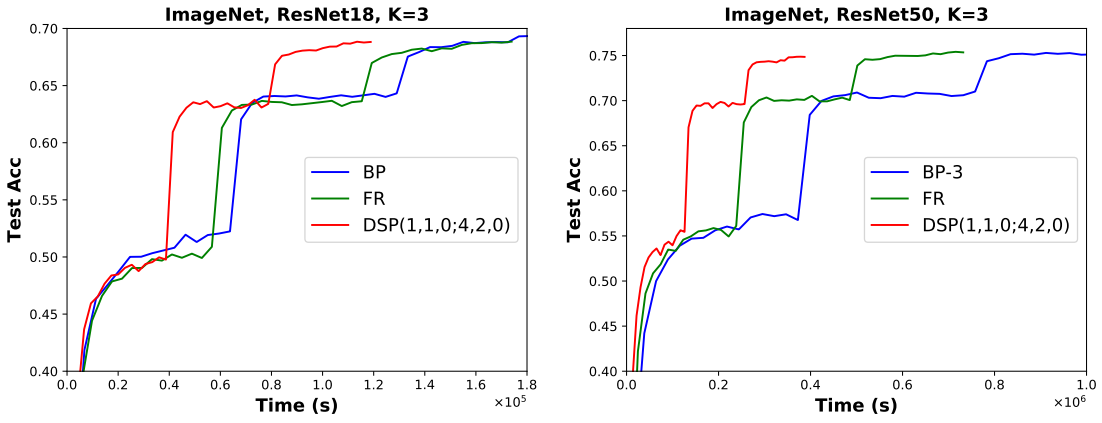


Figure 5: Test accuracy@1 on the ImageNet dataset.

# 2.0 Improve the Efficiency of Data Parallelism

## 2.1 Introduction

Deep learning models are hard to train due to the heavy computation complexity and long training iterations. Therefore, distributed deep learning with multiple workers (GPUs) has become a prevalent practice to parallelize and accelerate the training for large-scale tasks, where the model and dataset sizes continue to grow nowadays [120, 48, 28].

Nevertheless, synchronous distributed training have difficulty in scaling up the number of workers for large deep learning models, as the gradient in each worker to be communicated per iteration is of the same dimension as the model size. It is also known as the communication bottleneck. Besides, it incurs imbalanced communication traffic in the parameter-server [82, 84, 85] architecture, where the server suffers from much larger communication burden than workers. To address the communication bottleneck issue, there have been numerous lines of works including asynchronous execution [25], gradient compression [16, 17, 141, 8, 7, 9, 126, 93, 36], communication scheduling [38], infrequent communication [124], delayed gradient [89, 163], decentralized training [91, 73, 131, 12, 73], model parallelism [55, 147], etc.

In this work, we focus on synchronous distributed SGD with gradient compression, or more specifically, random block-wise gradient sparsification (RBGS) [135, 143]. The most popular gradient sparsifier is probably the Top-$\mathcal{K}$ gradient sparsification [9, 93], where each worker selects the largest $\mathcal{K}$ gradient components according to the absolute value as the sparsified gradient. However, Top-$\mathcal{K}$ has several drawbacks: 1) it requires extra communication overheads to communicate the gradient indices, 2) it is applied in parameter-server architecture but not ring-allreduce compatible, and most of all, 3) its computation overheads $\mathcal{O}(\mathcal{K} \log_2 d)$ for model $\theta \in \mathbb{R}^d$ may even outweigh its communication benefits [122, 143, 116] as it is efficient only for a small $\mathcal{K}$ for optimized implementations on GPU [119]. While in RBGS, we randomly sample a block of gradient as the sparsified gradient for communication among workers. To ensure the consistency of the sampling process, each worker will be

pre-assigned the same random seed. In comparison to Top-$\mathcal{K}$, RBGS is highly computation-efficient ($\mathcal{O}(1)$) as we only need to uniformly and randomly sample one starting index of the gradient block. RBGS is also ring-allreduce compatible. However, RBGS results in inferior model performance in that its sparsified gradient usually does not include as many significant gradient components as Top-$\mathcal{K}$, leading to large compression error.

To address this important problem, we propose a novel detached error feedback method (DEF), while the vanilla error feedback (EF) method [71, 161] fails to address it. We summarize our major contributions as follows.

- Our proposed DEF method is motivated by a novel insight that a trade-off between the gradient variance and second moment can improve the convergence bound related to compression error.
- We propose DEF-A to accelerate the generalization during the training with support from corresponding generalization analysis. It potentially demystifies why compression helps to improve the performance in some prior works [13, 160, 16, 17].
- We find that SGD with iterate averaging (SGD-IA) [107, 115, 103, 142] can be viewed as a special case of communication-efficient distributed SGD for the first time. Consequently, our generalization analysis of DEF-A extends to SGD-IA, providing potential theoretical explanations for some other applications incorporating SGD-IA [46, 63, 53].
- Extensive deep image classification experiments on CIFAR-10/100 and ImageNet show significant improvements of DEF(-A) over existing works with RBGS.

## 2.2 Related Works

To begin with, suppose the training dataset $\mathcal{S} = \{\xi_n\}_{n=1}^N$ and we have the training objective function

$$F_{\mathcal{S}}(\theta) = \frac{1}{N}\sum_{n=1}^{N} f(\theta;\xi_n) = \mathbb{E}_{\xi\in\mathcal{S}} f(\theta;\xi) \tag{2–1}$$

to minimize, where $\theta \in \mathbb{R}^d$ denotes the model and $f$ is the loss function. From now on, we will omit the subscript in $\mathbb{E}$ if the context is clear. For distributed SGD at iteration

$t$, each worker $k$ randomly selects one data sample $\xi_{k,t} \in \mathcal{S}$ and computes the stochastic gradient $g_{k,t} = \nabla f(\theta_t; \xi_{k,t})$. Then all the workers communication to get the average gradient $g_t = \frac{1}{K}\sum_{k=1}^{K} g_{k,t}$, where $K$ is the total number of workers, and update the model via

$$\theta_{t+1} = \theta_t - \eta g_t \,, \tag{2–2}$$

where $\eta$ is the learning rate.

**Compression.** Gradient compression includes quantization [16, 17, 141, 8], which reduces the 32-bit gradient component to as low as 1 bit (compression ratio $\leq$ 32), and sparsification [7, 9, 126], which reduces the number of gradient components for communication. Let the compression function be $\mathcal{C}$, then the workers will communicate $\mathcal{C}(g_{k,t})$ instead of $g_{k,t}$. In general, sparsification achieves flexible and higher compression ratio than quantization. Besides Top-$\mathcal{K}$, random-$\mathcal{K}$ [32, 125] randomly selects $\mathcal{K}$ gradient components as the sparsified gradient. [31] selects gradient components larger than a threshold and is a variable-dimension compressor. [140, 122] propose to select each gradient component with a probability to keep the sparsified gradient unbiased. In this work, we consider RBGS [135, 143], which is most easy to implement, highly computation-efficient, but challenging to retain the model performance. Moreover, it is ring-allreduce compatible for SOTA GPU communication backend library (e.g., NCCL), i.e.,

$$\mathcal{C}(\Delta_1) + \mathcal{C}(\Delta_2) = \mathcal{C}(\Delta_1 + \Delta_2) \,. \tag{2–3}$$

**Error Feedback.** Error feedback (EF) [71, 132] method maintains local compression error $e_{k,t}$ at worker $k$, adds it to the current gradient before compression, and communicates to average $\mathcal{C}(\eta g_{k,t} + e_{k,t})$. The error is updated via

$$e_{k,t+1} = \eta g_{k,t} + e_{k,t} - \mathcal{C}(g_{k,t} + e_{k,t}) \,. \tag{2–4}$$

[161] extends EF to momentum SGD [106]. EF works well for Top-$\mathcal{K}$ sparsifier but poorly for RBGS. [143] proposes PSync to immediately apply local error to each worker's model for RBGS. However, we will show that PSync works better for Wide ResNet [156] but has scalability issue for other common model architectures. SAEF [149] proposes to apply the local error before computing gradient in the next iteration to accelerate the generalization

during training. Other EF variants includes EF21 [110, 34] which compresses the gradient difference [99] but is evaluated only on logistic regression problems, acceleration for EF [108, 90], EF for variance reduction [130], etc.

**Generalization Analysis.** The generalization analysis of this work incorporates the uniform stability [20, 44] approach, focusing on the inherent stability property of the learning algorithm. [20] analyzes bagging methods. It is later used to analyze the generalization property of SGD [44] and its momentum variants [150]. [77] establishes a data-dependent notion of the stability to stress the distribution-dependent risk of the initialization point and make the generalization bounds more optimistic. [162] analyzes the generalization of the Lookahead optimizer [158] with uniform stability.

As there are numerous works combining various techniques [14], in this work, we focus on random block-wise gradient sparsification (RBGS).

## 2.3 Detached Error Feedback

In this section, we described our proposed DEF method (Algorithm 1) in detail. As RBGS is a very aggressive compressor, the algorithm is crucial for better performance.

### 2.3.1 Motivation

In EF variants [71, 161, 143] for practical large-scale distributed training of deep learning models, Assumptions 2.3.1 and 2.3.2 are needed to bound the norm of the stochastic gradient

$$\|\nabla f(\theta;\xi)\|_2 \leq G = \sqrt{\sigma^2 + M^2}\,. \tag{2–5}$$

Then $G$ bounds the compression error

$$\frac{1}{K}\sum_{k=1}^{K} \|e_{k,t}\|_2^2 = \mathcal{O}(\sigma^2 + M^2) \tag{2–6}$$

at iteration $t$. Though Assumption 2.3.2 often appears in related literature, it is usually regarded as a strong assumption [110] because $M^2$ could be much larger than $\sigma^2$. Hereby,

**Algorithm 1** Detached Error Feedback (DEF(-A)).

**Input:** training dataset $\mathcal{S}$, number of iterations $T$, number of workers $K$, learning rate $\eta$, ring-allreduce compressor $\mathcal{C}$, coefficient $\lambda \in [0, 1]$.
**Initialize:** model $x_0 = y_0$, local compression error $e_{k,0} = 0$, worker $k \in [K]$.
**for** $t = 0, 1, \cdots, T-1$ **do**
  **for** worker $k \in [k]$ in parallel **do**
    Randomly sample data $\xi_{k,t}$ from $\mathcal{S}$.
    Compute $g_{k,t} = \nabla f(x_t - \lambda e_{k,t}; \xi_{k,t})$. *// detach*
    $p_{k,t} = \eta g_{k,t} + e_{k,t}$. *// error feedback*
    $e_{k,t+1} = p_{k,t} - \mathcal{C}(p_{k,t})$.
    Ring-allreduce: $\mathcal{C}(p_t) = \mathcal{C}(\frac{1}{K}\sum_{k=1}^{K} p_{k,t}) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{C}(p_{k,t})$.
    Update $x_{t+1} = x_t - \mathcal{C}(p_t)$.
  **end for**
**end for**
**Output:** $y_T = x_T - e_T = x_T - \frac{1}{K}\sum_{k=1}^{K} e_{k,T}$ for DEF and $x_T$ for DEF-A.

we propose a novel insight that if some trade-off coefficient $\alpha$ can be introduced to transform the compression error bound to a similar interpolation form as

$$(\alpha\sigma)^2 + ((1-\alpha)M)^2 \overset{\alpha=\frac{M^2}{\sigma^2+M^2}}{\geq} \frac{\sigma^2 M^2}{\sigma^2 + M^2} \overset{M\gg\sigma}{=} \sigma^2\,, \tag{2–7}$$

then the bound $\mathcal{O}(\sigma^2 + M^2)$ can be reduced to $\mathcal{O}(\sigma^2)$ when $M \to \infty$, i.e., Assumption 2.3.2 does not hold.

**Assumption 2.3.1.** *(Bounded Variance)* $\forall \theta \in \mathbb{R}^d$, *the variance of the stochastic gradient satisfies* $\mathbb{E}_{\xi\in\mathcal{S}}\|\nabla f(\theta;\xi) - \nabla F_{\mathcal{S}}(\theta)\|_2^2 \leq \sigma^2$.

**Assumption 2.3.2.** *(Bounded Second Moment)* $\forall \theta \in \mathbb{R}^d$, *the second moment of the full gradient satisfies* $\|\nabla F_{\mathcal{S}}(\theta)\|_2^2 \leq M^2$.

### 2.3.2 Algorithm

**Assumption 2.3.3.** *(Ring-allreduce Compressor)* $\forall \Delta_1, \Delta_2 \in \mathbb{R}^d$, *the compressor* $\mathcal{C}$ *satisfies* $\mathcal{C}(\Delta_1) + \mathcal{C}(\Delta_2) = \mathcal{C}(\Delta_1 + \Delta_2)$.

Firstly, the ring-allreduce communication requires that the compressor should satisfy Assumption 2.3.3 such that Algorithm 1 line 9 holds. RBGS satisfies such a assumption.

Secondly, DEF returns $y_T = x_T - e_T$ by default because we have

$$y_{t+1} = y_t - \eta g_t = y_t - \frac{\eta}{K} \sum_{k=1}^{K} g_{k,t} \,. \tag{2–8}$$

In particular, when $K = 1$ (single worker) and $\lambda = 1$, $\{y_t\}$ is identical to the SGD solution path. We note that averaging $e_T = \frac{1}{K} \sum_{k=1}^{K} e_{k,T}$ only incurs a one-time communication cost after the training concludes.

Then, a major difference of DEF and EF is that we evaluate gradient at $x_t - \lambda e_{k,t}$, a point **detached** from the point $x_t$ to evaluate gradient as in EF. This step does not incur any communication cost. From Eq. (2–8), our goal is to make sure that the point to evaluate gradient $g_{k,t}$ is as close to $y_t = x_t - e_t$ as possible. For EF, the distance is $\|x_t - y_t\|_2^2 = \|e_t\|_2^2 \leq \frac{1}{K} \sum_{k=1}^{K} \|e_{k,t}\|_2^2$, while for DEF, the average distance to minimize regarding $\lambda$ becomes

$$\frac{1}{K} \sum_{k=1}^{K} \|x_t - \lambda e_{k,t} - y_t\|_2^2 = \frac{1}{K} \sum_{k=1}^{K} \|e_t - \lambda e_{k,t}\|_2^2 \,. \tag{2–9}$$

(1) When $\lambda = \lambda(k,t)$, it is obvious that $\lambda^*(k,t) = \frac{\langle e_t, e_{k,t} \rangle}{\|e_{k,t}\|_2^2}$, which is determined by the *projection* of $e_t$ onto $e_{k,t}$. However, it is impractical to decide $\lambda^*(k,t)$ for worker $k$ at iteration $t$ as $e_t$ is unknown ($e_t = \frac{1}{K} \sum_{k=1}^{K} e_{k,t}$ needs extra communication cost).

(2) When $\lambda = \lambda(t)$, we can derive $\lambda^*(t) = \frac{\|e_t\|_2^2}{\frac{1}{K} \sum_{k=1}^{K} \|e_{k,t}\|_2^2}$, which is still impractical due to unkown $e_t$.

(3) Therefore, we will regard $\lambda$ as a tuned hyper-parameter, invariant regarding $k$ and $t$. Then it becomes minimizing the sum of the errors $\frac{1}{KT} \sum_{t=0}^{T-1} \sum_{k=1}^{K} \|e_t - \lambda e_{k,t}\|_2^2$ which will appear in the convergence bound of DEF, similar to the suggestion in [116]. Previously when $\lambda$ is a function of $t$, it reduces to minimizing Eq. (2–9). In our CIFAR-10 VGG-16

experiments with $\lambda = 0.3$, we find that the new distance is $\times 1.7$ smaller than the distance in EF.

**Relation to Motivation.** Minimizing Eq. (2–9) is closely related to the motivation since

$$\begin{aligned}\|e_t - \lambda e_{k,t}\|_2^2 = \|(&\underbrace{\eta g_{t-1} - \lambda \eta g_{k,t-1}}_{} + e_{t-1} - \lambda e_{k,t-1}) \\ &- \mathcal{C}(\underbrace{\eta g_{t-1} - \lambda \eta g_{k,t-1}}_{} + e_{t-1} - \lambda e_{k,t-1})\|_2^2 \,,\end{aligned} \tag{2–10}$$

where $g_{t-1} - \lambda g_{k,t-1}$ is affected by the gradient variance and second moment trade-off via the choice of $\lambda$. For example, in extreme circumstances where $\sigma = 0$, in expectation, local errors on different workers are the same and $g_{t-1} - \lambda g_{k,t-1}$ is zero with $\lambda = 1$.

**Momentum Variant.** It is easy to extend DEF to momentum SGD variant. Let the momentum buffer on worker $k$ be $m_{k,0} = 0$ and the momentum constant be $\mu$. We only need to substitute Algorithm 1 line 7 with

$$m_{k,t+1} = \mu m_{k,t} + g_{k,t},\ p_{k,t} = \eta m_{k,t+1} + e_{k,t}\,. \tag{2–11}$$

**DEF-A.** Simply returning $x_T$ can accelerate the generalization performance of DEF during training in that when $K = 1$, $\lambda = 1$ and $\mathcal{C}(\Delta) = \delta\Delta$ $(0 < \delta < 1)$, $\{y_t\}$ reduces to SGD and $\{x_t\}$ reduces to a special case of SGD-IA (Iterate Averaging, a combination of models in each iteration) [142]:

$$x_t = \underbrace{(1-\delta)^t}_{P_0} y_0 + \sum_{t'=1}^{t} \underbrace{\delta(1-\delta)^{t-t'}}_{P_{t'}} y_{t'}\,, \tag{2–12}$$

where $P_0 + P_1 + \cdots + P_t = 1$. Note that for Polyak-Ruppert IA [107], $P_0 = P_1 = \cdots = P_t = \frac{1}{t+1}$. While for geomeric Polyak-Ruppert IA [103], $P_{t'} = \frac{\beta^{t'}}{1+\beta+\cdots+\beta^t}$ where $0 < \beta < 1$ is some constant and $0 \le t' \le t$. However, this part is based on generalization analysis instead of convergence analysis as for DEF. Hence we leave the details of the general case in the next section.

## 2.4 Theoretical Analysis

In this section, we consider non-convex objective functions as our target is the deep learning model. All detailed proof can be found in the Appendix. Suppose that each $\xi_n$ in the training dataset $\mathcal{S}$ is i.i.d drawn from an unknown data distribution $\mathcal{D}$ and $F_{\mathcal{D}}(\theta) = \mathbb{E}_{\xi \in \mathcal{D}} f(\theta; \xi)$. For generalization, we are interested in how the model $\theta_{\mathcal{A},\mathcal{S}}$, which is trained on $\mathcal{S}$ with a randomized algorithm $\mathcal{A}$, generalizes on $\mathcal{D}$ by measuring the well-known excess risk error $\epsilon$.

$$\begin{aligned} \epsilon &= \mathbb{E}_{\mathcal{A},\mathcal{S}}[F_{\mathcal{D}}(\theta_{\mathcal{A},\mathcal{S}})] - \mathbb{E}_{\mathcal{A},\mathcal{S}}[F_{\mathcal{S}}(\theta^*_{\mathcal{S}})] \\ &= \underbrace{\mathbb{E}_{\mathcal{A},\mathcal{S}}[F_{\mathcal{S}}(\theta_{\mathcal{A},\mathcal{S}}) - F_{\mathcal{S}}(\theta^*_{\mathcal{S}})]}_{\text{optimization error } \epsilon_{\text{opt}}} + \underbrace{\mathbb{E}_{\mathcal{A},\mathcal{S}}[F_{\mathcal{D}}(\theta_{\mathcal{A},\mathcal{S}}) - F_{\mathcal{S}}(\theta_{\mathcal{A},\mathcal{S}})]}_{\text{generalization error } \epsilon_{\text{gen}}} \end{aligned} \tag{2–13}$$

**Assumption 2.4.1.** *(L-Lipschitz Smooth)* $\forall \theta_1, \theta_2 \in \mathbb{R}^d$*, the loss function satisfies*

$$\|\nabla f(\theta_1; \xi) - \nabla f(\theta_2; \xi)\|_2 \leq L \|\theta_1 - \theta_2\|_2 \,. \tag{2–14}$$

*It also implies that*

$$\|\nabla F(\theta_1) - \nabla F(\theta_2)\|_2 \leq L \|\theta_1 - \theta_2\|_2 \,. \tag{2–15}$$

**Assumption 2.4.2.** *(*$\delta$*-approximate Compressor)* $\forall \Delta \in \mathbb{R}^d$*, the compressor* $\mathcal{C}$ *satisfies*

$$\|\mathcal{C}(\Delta) - \Delta\|_2^2 \leq (1 - \delta) \|\Delta\|_2^2 \,, \tag{2–16}$$

*where* $0 < \delta < 1$ *is related to the compression ratio.*

This assumption is widely used in communication-efficient distributed SGD [71, 161, 143]. For RBGS, we can take an expectation over the random compression and $\delta$ will be identical to the compression ratio.

### 2.4.1 Convergence Rate

In this section, we bound the gradient norm $\|\nabla F(\theta_{\mathcal{A},\mathcal{S}})\|_2^2$ for convergence rate analysis of the proposed DEF method.

**Theorem 2.4.1.** *(Convergence Rate of DEF, Appendix B.1) Let Assumptions 2.3.1, 2.3.2, 2.3.3, 2.4.1 and 2.4.2 hold. If $\eta \leq \frac{1}{4L}$, we have*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 \leq \frac{4\mathbb{E}[F_{\mathcal{S}}(y_0)-F_{\mathcal{S}}(y^*)]}{\eta T}+\frac{2\eta L\sigma^2}{K} + \frac{4\eta^2L^2[\frac{K-1}{K^2}\sigma^2+(\frac{1}{K}-\lambda)^2\sigma^2+2(1-\lambda)^2M^2]}{(\sqrt{(1-\delta/2)/(1-\delta)}-1)^2}\,. \tag{2–17}$$

**Remark 2.4.1.** *Suppose $\theta_{\mathcal{A},\mathcal{S}}$ is randomly chosen from the sequence $\{y_t\}_{t=0}^{T-1}$, $\eta=\mathcal{O}(\sqrt{\frac{K}{T}})\leq\frac{1}{4L}$, and $K=\mathcal{O}(T^{1/3})$ (i.e., $T$ is large enough), we have $\mathbb{E}\|\nabla F_{\mathcal{S}}(\theta_{\mathcal{A},\mathcal{S}})]\|_2^2=\mathcal{O}(\frac{1}{\sqrt{KT}}+\frac{K}{T})=\mathcal{O}(\frac{1}{\sqrt{KT}})$. It matches the rate of SGD with linear speedup regarding the number of workers $K$.*

**Remark 2.4.2.** *The last term in Eq. (2–17) is determined by the compression error. When $\delta$, $\sigma$, $M$ are of interest, we have $\mathbb{E}\|\nabla F_{\mathcal{S}}(\theta_{\mathcal{A},\mathcal{S}})]\|_2^2=$*

$$\mathcal{O}(\frac{\frac{K-1}{K^2}\sigma^2+(\frac{1}{K}-\lambda)^2\sigma^2+2(1-\lambda)^2M^2}{(\sqrt{(1-\delta/2)/(1-\delta)}-1)^2})\,. \tag{2–18}$$

*(1) When $K=1$ (single worker) and $\lambda=1$, it vanishes, which is better than EF [71, 161].*

*(2) When $\sigma$ and $M$ are of interest, following the motivation in the previous section and ignoring other constant factors, Eq. (2–18) becomes*

$$\mathcal{O}(\frac{K-1}{K^2}\sigma^2+\frac{2(1-\frac{1}{K})^2\sigma^2M^2}{\sigma^2+2M^2})\,. \tag{2–19}$$

*when $\lambda=\frac{\frac{1}{K}\sigma^2+2M^2}{\sigma^2+2M^2}$. It further reduces to $\mathcal{O}(\frac{K-1}{K}\sigma^2)$ when $M\to\infty$ (i.e. Assumption 2.3.2 does not hold). Therefore, DEF is the first EF variant compressing gradient without relying on the bound of the gradient second moment.*

*(3) When $K$ is large and $\sigma$ and $M$ are of interest, our bound improves $\mathcal{O}(\sigma^2+M^2)$ [71, 161, 143] to*

$$\mathcal{O}(\frac{2\sigma^2M^2}{\sigma^2+2M^2})\,. \tag{2–20}$$

*Our empirical deep learning experiments suggest that $\sigma^2\approx 0.3M^2$, which means that our bound is about $\times 5$ smaller ignoring other constant factors.*

### 2.4.2 Generalization Rate

In this section, we consider non-convex objective functions under PL condition, which establishes the relation between the gradient norm and the optimization error $\epsilon_{\text{opt}}$ [162]. We bound the excess risk error $\epsilon = \epsilon_{\text{opt}} + \epsilon_{gen}$ for the generalization analysis of the proposed DEF(-A) method.

**Polyak-Łojasiewicz (PL) Condition** [68]. Let $\theta^* \in \min_{\theta \in \mathbb{R}^d} F_{\mathcal{S}}(\theta)$. The objective function $F_{\mathcal{S}}(\theta)$ satisfies $\mu$-PL condition if $\forall \theta \in \mathbb{R}^d$, we have

$$2\mu[F_{\mathcal{S}}(\theta) - F_{\mathcal{S}}(\theta^*)] \leq \|\nabla F_{\mathcal{S}}(\theta)\|_2^2 . \tag{2–21}$$

**Theorem 2.4.2.** *(Excess Risk Error of DEF(-A), Appendix B.2) Let Assumptions 2.3.1, 2.3.2, 2.3.3, 2.4.1 and 2.4.2 hold. Suppose $\eta = \frac{c}{t+1}$, where $c > 0$ is some constant.*
*(1) The generalization error of DEF*

$$\epsilon_{gen} = \mathcal{O}(T^{(1-\frac{K}{N})Lc/((1-\frac{K}{N})Lc+1)}) . \tag{2–22}$$

*(2) Suppose $\eta \leq \frac{1}{4L}$. The optimization error of DEF*

$$\epsilon_{opt} = \widetilde{\mathcal{O}}(T^{-\frac{\mu c}{2}} + T^{-1}) . \tag{2–23}$$

*(3) For RBGS, the generalization error of DEF-A*

$$\epsilon_{gen} = \mathcal{O}(T^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc/((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc+1)}) . \tag{2–24}$$

*(4) Suppose $\eta \leq \frac{1}{8L}$. The optimization error of DEF-A*

$$\epsilon_{opt} = \widetilde{\mathcal{O}}(T^{-\frac{\mu\delta c}{2}} + T^{-1} + (1/\sqrt{1-\delta} - 1)^{-2}) . \tag{2–25}$$

**Remark 2.4.3.** *When $K = 1$, Eq. (2–22) matches the result of SGD in [44].*

**Remark 2.4.4.** *DEF-A has a better $\epsilon_{gen}$ but a worse $\epsilon_{opt}$ than DEF. Since $\epsilon = \epsilon_{gen} + \epsilon_{opt}$, DEF-A can achieve better generalization rate than DEF via a trade-off between $\epsilon_{gen}$ and $\epsilon_{opt}$ with a proper $\delta$.*

Table 5: The CIFAR-10 test accuracy (%) of DEF/DEF-A for various $\lambda$ with VGG-16. The compression ratio is 64.

| $\lambda$ | 0.1 | 0.2 | **0.3** |
|---|---|---|---|
| DEF | 92.83 ± 0.19 | 93.45 ± 0.06 | **93.75** ± 0.12 |
| DEF-A | 92.78 ± 0.13 | 93.20 ± 0.14 | <u>93.61</u> ± 0.07 |
| $\lambda$ | 0.4 | 0.6 | 0.8 |
| DEF | 93.41 ± 0.12 | 93.26 ± 0.10 | 92.60 ± 0.19 |
| DEF-A | 93.41 ± 0.20 | 93.11 ± 0.20 | 92.59 ± 0.15 |

**Remark 2.4.5.** *Theorem 2.4.2 provides a potential new theoretical insight for applications incorporating compression, though some of them were not related to communication-efficient distributed training. E.g., escaping saddle point with compressed gradient [13], feature quantization to improve GAN training [160], SignSGD that empirically accelerates training [16, 17], etc.*

### 2.4.3 Extension to Iterate Averaging (IA)

As SGD and SGD-IA is a special case of DEF and DEF-A respectively when $K = 1$, $\lambda = 1$, and $\mathcal{C}(\Delta) = \delta\Delta$, we immediately have the following Theorem 2.4.3.

**Theorem 2.4.3.** *(Excess Risk Error of SGD(-IA), Appendix B.3) Let Assumptions 2.3.1, 2.3.2, 2.3.3, 2.4.1 and 2.4.2 hold. Suppose $\eta = \frac{c}{t+1}$, where $c > 0$ is some constant.*
*(1) The generalization error of SGD*

$$\epsilon_{gen} = \mathcal{O}(T^{(1-\frac{1}{N})Lc/((1-\frac{1}{N})Lc+1)}) . \tag{2–26}$$

*(2) Suppose $\eta \leq \frac{1}{4L}$. The optimization error of SGD*

$$\epsilon_{opt} = \widetilde{\mathcal{O}}(T^{-\frac{\mu c}{2}} + T^{-1}) . \tag{2–27}$$

*(3) The generalization error of SGD-IA*

$$\epsilon_{gen} = \mathcal{O}(T^{(1-\frac{1}{N})\delta Lc/((1-\frac{1}{N})\delta Lc+1)}) . \tag{2–28}$$

*(4) Suppose $\eta \leq \frac{1}{8\delta L}$. The optimization error of SGD-IA*

$$\epsilon_{opt} = \widetilde{\mathcal{O}}(T^{-\frac{\mu\delta c}{2}} + T^{-1} + (1/\sqrt{1-\delta} - 1)^{-2}) \,. \tag{2–29}$$

**Remark 2.4.6.** *We have $\delta^{\frac{1}{2}}$ in Eq. (2–24) but $\delta$ in Eq. (2–28) because $\mathbb{E}_{\mathcal{C}}[\mathcal{C}(\Delta)] = \delta\Delta$ for RBGS but $\mathcal{C}(\Delta) = \delta\Delta$ for SGD-IA.*

**Remark 2.4.7.** *SGD-IA can achieve better generalization rate than SGD with a proper $\delta$. [103, 142] theoretically only show that SGD-IA achieves adjustable regularization for strongly-convex objective functions, while SGD-IA applications such as averaging weights [63] and ensemble of models during training with cyclic learning rate [53] only empirically show better generalization than SGD.*

**Remark 2.4.8.** *Compare with Theorem 2.4.2, we can see that DEF-A generalizes better than SGD with a proper $\delta$.*

**Remark 2.4.9.** *Theorem 2.4.3 provides a new theoretical explanation for an important line of works in unsupervised learning - momentum contrast [46]. In [46], two sets of weights are maintained with a contrastive loss. One is the "query" $y_t$ which is updated via SGD, and the other is the "key" $x_t$ ($x_0 = y_0$) which is updated via*

$$x_{t+1} = (1-\delta)x_t + \delta y_t \,. \tag{2–30}$$

*The success of momentum contrast is explained as a "slowly progressing" key $x_t$ [46] without theoretical guarantee. Interestingly, the above equation is identical to Eq. (2–12), i.e. SGD-IA. Therefore, our results suggests that the slowly progressing key $x_t$ may actually have stabler and better generalization than the query $y_t$ depending on $\delta$.*

## 2.5 Experiments

In this section, we conduct empirical experiments on benchmark deep learning tasks following settings in [71, 161, 143] to validate the performance of the proposed detached error feedback (DEF) method. We compare the following methods with RBGS as the gradient compressor: (1) SGD, which is the upper bound without gradient compression, (2) EF [71, 161], (3) SAEF [149], (4) PSync [143], and (5) the proposed DEF(-A), where $\lambda = 0.3$ by default. We have also tested EF21 [110, 34] on our deep learning tasks with RBGS, but it does not converge.

**Settings.** All experiments are implemented using PyTorch and conducted on a cluster of machines connected by. Each machine is equipped with 4 NVIDIA P40 GPUs and there are 16 workers (GPUs) in total. We use NCCL as the backend of the PyTorch distributed package. The task-specific settings are as follows.

**CIFAR.** We train VGG-16 [120], ResNet-110 [48] and Wide ResNet (WRN-28-10) [156] models CIFAR-10/100 [76] image classification task. We report the mean and standard deviation metrics over 3 runs. The base learning rate is tuned from $\{\cdots, 0.1, 0.05, 0.01, \cdots\}$ and the batch size is 128. The momentum constant is 0.9 and the weight decay is $5 \times 10^{-4}$. The model is trained for 200 epochs with a learning rate decay of 0.1 at epoch 100 and 150. Random cropping, random flipping, and standardization are applied as data augmentation techniques.

**ImageNet.** We train the ResNet-50 model on ImageNet [28] image classification tasks. The model is trained for 100 epochs with a learning rate decay of 0.1 at epoch 30, 60, and 90. The base learning rate is tuned from $\{\cdots, 0.1, 0.05, 0.01, \cdots\}$ and the batch size is 256. The momentum constant is 0.9 and the weight decay is $1 \times 10^{-4}$. Similar data augmentation techniques as in CIFAR experiments are applied.

### 2.5.1 General Results

We plot the CIFAR-10 training curves of VGG-16 in Figure 6 and summarize the test numbers under various compression ratio settings in Table 6. From the curves, DEF achieves

the best test acc and training loss among all the communication-efficient methods. Compared with SGD, DEF achieves $\times 3.6$ and $\times 4.0$ speedup when the compression ratio is 64 and 256 respectively. For the test numbers in the table, DEF and DEF-A achieve the best results, which can be comparable to SGD for compression up to 256. When the compression ratio is high, DEF(-A) can significantly improve over the best counterpart by **8%**. Overall, DEF and DEF-A have similar final test performances. A significant improvement of the training loss over the existing EF variants can be observed, validating our lower bound in the convergence analysis of DEF.

The ImageNet training curves of ResNet-50 is shown in Figure 7 and the test numbers under various compression ratio settings are summarized in Table 7. We can reach similar conclusions as in CIFAR-10 experiments. Specifically, DEF achieves $\times 2.5$ and $\times 2.9$ speedup compared with SGD when the compression ratio is 64 and 256 respectively. For the test numbers in the table, DEF and DEF-A can be comparable to SGD for the compression ratio of 1024. For some smaller compression ratios, we may even see a slight improvement over SGD. When the compression ratio is high, DEF(-A) can significantly improve the best counterpart by **4%**.

For the concern of scalability, we also summarize the test numbers of VGG-16, ResNet-110, and WRN-28-10 on CIFAR-10/100 in Table 8 with 64 as the compression ratio. DEF-A achieves lossless performance compared with SGD and largely improves all the counterparts. We find that for VGG-16 on CIFAR-100 and ResNet-110 on CIFAR-10/100, DEF-A has a noticeable improvement over DEF. In particular, we find that PSync achieves closer performance to SGD on WRN as reported in [143], but is much worse on VGG-16 and ResNet-110. Therefore, both the superior performance and scalability of DEF(-A) are validated.

### 2.5.2 Accelerate Generalization

Here we empirically validate the theoretical generalization analysis that DEF-A has a better generalization rate than DEF. We plot the training curves for VGG-16 on CIFAR-10 and ResNet-50 on ImageNet with compression ratio as 64 in Figure 8. We can see that DEF-A does have a much faster generalization rate than DEF. Specifically, the test accuracy

improvement is about **15%** on CIFAR-10 and **25%** on ImageNet before the first learning rate decay, which validates the theoretical benefits in our generalization analysis. DEF-A can be even faster than full-precision SGD.

A significant improvement can still be observed before the second learning rate decay, but it becomes smaller when the learning rate is smaller. This matches our generalization analysis well. Let $c$ be smaller such that the learning rate $\eta = \frac{c}{t+1}$ is smaller, then Eq. (2–22) is closer to Eq. (2–24), that is, the DEF-A's generalization error improvement over DEF becomes smaller. Then it is obvious that the excess risk error improvement will also become smaller.

### 2.5.3 Hyperparameter $\lambda$

Here we explore DEF(-A) with various choices of the hyper-parameter $\lambda$ with results summarized in Table 5. We can just set $\lambda = 0.3$ by default for the best performance. In comparison, an inappropriate choice of $\lambda$ (e.g., 0.1 and 0.8) can lead to the performance degradation of about 1%. We also observe that a wide range of $\lambda$ such as $0.2 \sim 0.6$ can result in fairly good performance compared with $\lambda = 0.3$, which means that the proposed DEF(-A) is not too sensitive to the hyper-parameter $\lambda$.

## 2.6 Conclusion

In this work, to address the performance loss issue for communication-efficient distributed SGD with the gradient sparsifier RBGS, we proposed a new DEF(-A) algorithm motivated by the trade-off between gradient variance and second moment. Our convergence analysis shows better bounds without relying on the bound of gradient second moment. We conduct the first generalization analysis for communication-efficient distributed training to show that DEF-A can generalize faster than DEF and SGD, which sheds light on other applications incorporating compression such as escaping saddle point, GAN training, and SignSGD training. We establish the connection to SGD-IA for the first time, thus our analysis provides

potential theoretical explanations for SGD-IA applications such as averaging weights, ensemble, and momentum contrast in unsupervised learning. Last but not least, deep learning experiments validate the significant improvement of DEF(-A) over existing EF variants.

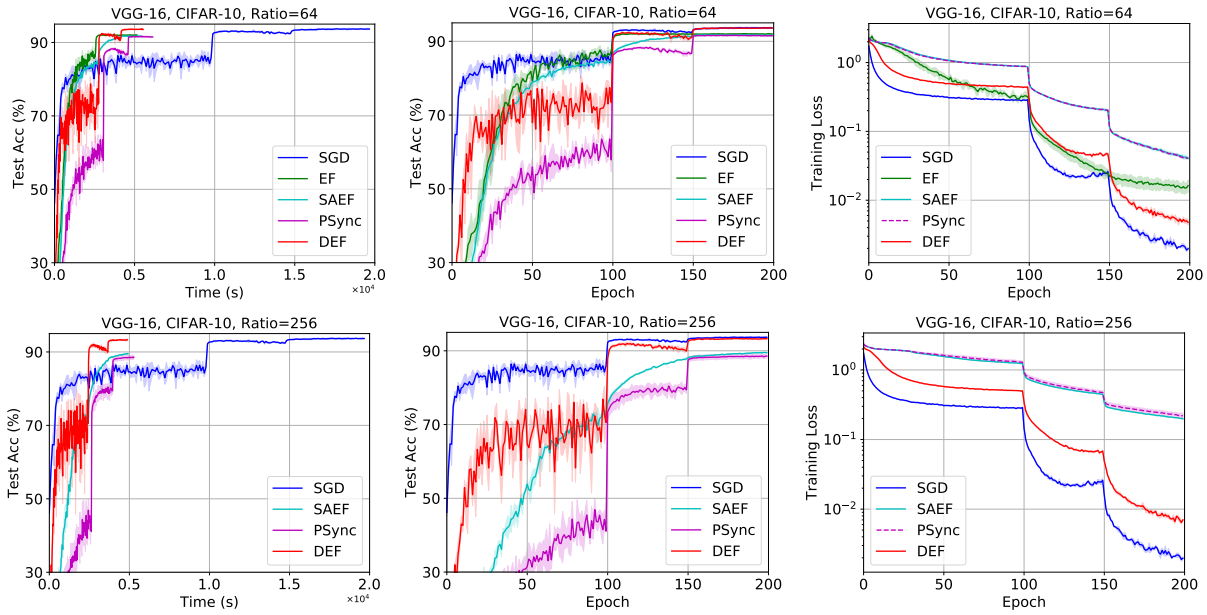


Figure 6: CIFAR-10 training curves of VGG-16. The compression ratio is 64 for the top row and 256 for the bottom row. EF is not plotted when the compression ratio is 256 due to divergence. From the left to right column, we plot the test accuracy (%) v.s. the wall-clock time, the test accuracy (%) v.s. training epochs, and the training loss v.s. training epochs respectively.

Table 6: The CIFAR-10 test accuracy (%) comparison of under various compression ratio settings with VGG-16.

| Ratio | SGD | EF | SAEF | PSync | DEF | DEF-A |
|---|---|---|---|---|---|---|
| 1 | 93.76 ± 0.14 | — | — | — | — | — |
| 16 | — | 93.04 ± 0.13 | 93.15 ± 0.04 | 93.31 ± 0.21 | <u>93.61</u> ± 0.04 | **93.66** ± 0.10 |
| 64 | — | 92.16 ± 0.06 | 91.88 ± 0.14 | 91.79 ± 0.17 | **93.75** ± 0.12 | <u>93.61</u> ± 0.07 |
| 256 | — | diverge | 89.59 ± 0.04 | 88.70 ± 0.61 | **93.45** ± 0.11 | <u>93.33</u> ± 0.26 |
| 512 | — | diverge | 87.83 ± 0.36 | 86.47 ± 0.14 | <u>93.24</u> ± 0.08 | **93.25** ± 0.18 |
| 1024 | — | diverge | 85.46 ± 0.80 | 84.27 ± 0.33 | <u>93.03</u> ± 0.15 | **93.06** ± 0.09 |

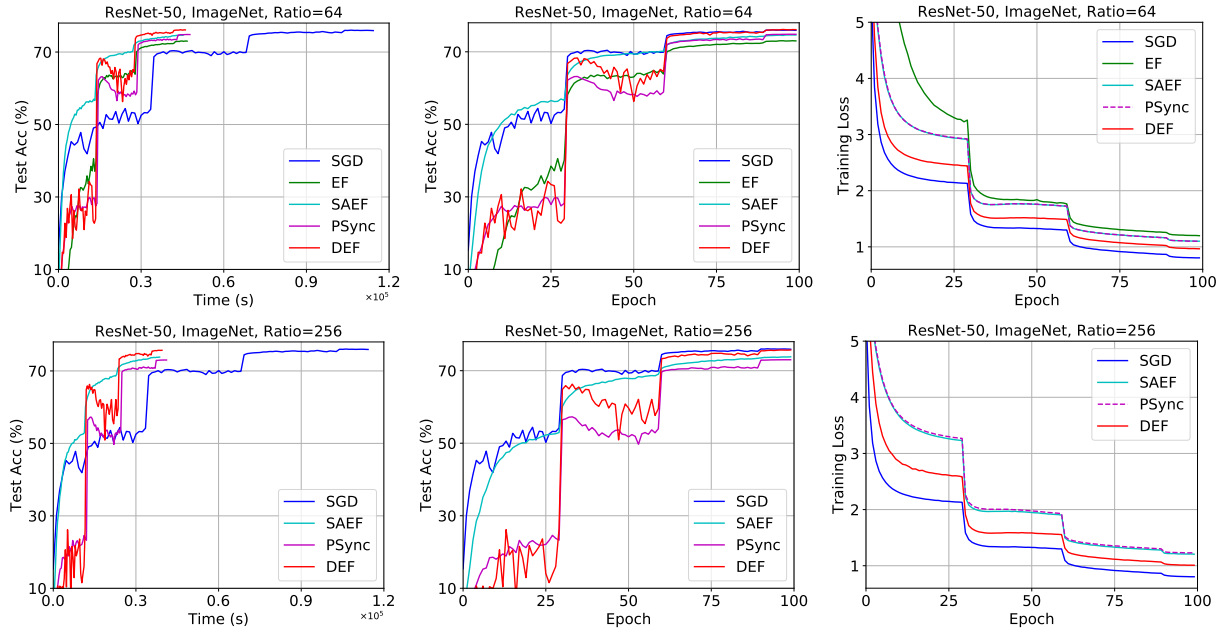


Figure 7: ImageNet training curves of ResNet-50. The compression ratio is 64 for the top row and 256 for the bottom row. From the left to right column, we plot the test accuracy (%) v.s. the wall-clock time, the test accuracy (%) v.s. training epochs, and the training loss v.s. training epochs respectively.

Table 7: The ImageNet test accuracy (%) comparison under various compression ratio settings with ResNet-50.

| Ratio | SGD | EF | SAEF | PSync | DEF | DEF-A |
|---|---|---|---|---|---|---|
| 1 | 76.04 | — | — | — | — | — |
| 16 | — | 75.29 (↓ 0.75) | 75.83 (↓ 0.21) | 75.63 (↓ 0.41) | <u>75.98</u> (↓ 0.06) | **76.10** (↑ 0.06) |
| 64 | — | 73.05 (↓ 2.99) | 74.65 (↓ 1.39) | 74.84 (↓ 1.20) | <u>76.16</u> (↑ 0.12) | **76.37** (↑ 0.33) |
| 128 | — | 63.80 (↓ 12.2) | 74.26 (↓ 1.78) | 74.12 (↓ 1.92) | **76.17** (↑ 0.13) | <u>76.14</u> (↑ 0.10) |
| 256 | — | diverge | 73.83 (↓ 2.21) | 73.02 (↓ 3.02) | <u>75.71</u> (↓ 0.33) | **76.00** (↓ 0.04) |
| 512 | — | diverge | 73.00 (↓ 3.04) | 72.60 (↓ 3.44) | <u>75.52</u> (↓ 0.52) | **75.77** (↓ 0.27) |
| 1024 | — | diverge | 71.89 (↓ 4.15) | 71.82 (↓ 4.22) | **75.64** (↓ 0.40) | <u>75.57</u> (↓ 0.47) |

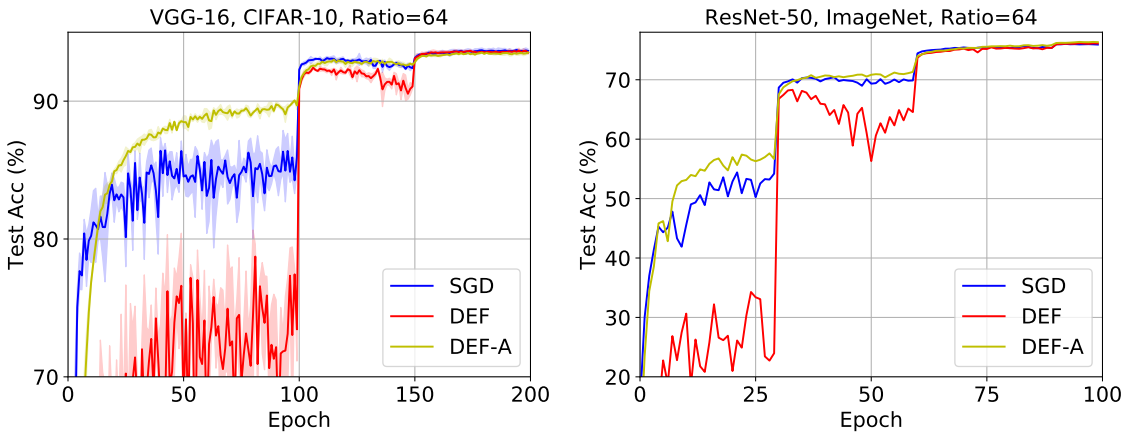


Figure 8: Accelerate the generalization with DEF-A. DEF-A significantly improves the test accuracy before the second learning rate decay compared with DEF.

Table 8: The CIFAR-10/100 test accuracy (%) comparison for various model architectures. The compression ratio is 1 for SGD and 64 for the other methods.

| Method | VGG-16 | ResNet-110 | WRN-28-10 |
|---|---|---|---|
| SGD | 93.76 ± 0.14 / 72.50 ± 0.33 | 94.73 ± 0.06 / 76.78 ± 0.29 | 96.21 ± 0.07 / 80.81 ± 0.12 |
| EF | 92.16 ± 0.06 / 68.87 ± 0.21 | diverge | diverge |
| SAEF | 91.88 ± 0.14 / 67.00 ± 0.07 | 92.95 ± 0.17 / 71.19 ± 0.31 | 95.33 ± 0.01 / 79.04 ± 0.01 |
| PSync | 91.79 ± 0.17 / 65.68 ± 0.16 | 92.26 ± 0.04 / 69.21 ± 0.04 | 95.44 ± 0.13 / 79.60 ± 0.12 |
| DEF | **93.75** ± 0.12 / 72.02 ± 0.10 | 94.34 ± 0.06 / 76.43 ± 0.12 | **96.26** ± 0.05 / 80.88 ± 0.16 |
| DEF-A | 93.61 ± 0.07 / **72.38** ± 0.07 | **94.66** ± 0.07 / **76.98** ± 0.21 | 96.24 ± 0.12 / **80.95** ± 0.16 |

# 3.0 Improve the Performance with Data Privacy

## 3.1 Introduction

Deep learning models have shown success in computer vision tasks in recent years [48, 120, 112]. However, training deep models that generalize well on unseen test data may require massive training data. Unfortunately, we are usually faced with **insufficient data** in a single medical institution for the medical image segmentation task due to the expensive procedure of collecting enough patients' data with experts' labeling.

A straightforward solution to address the insufficient data issue is gathering data from all the available medical institutions, while the amount of data owned by any single institution may be insufficient to train a well-performing deep model. However, this approach will raise the concern for **data privacy**. On one hand, collecting medical data is expensive as mentioned above, and those data have become a valuable asset at a medical institution. Institutions with more data may be more reluctant to contribute their data. In addition, medical institutions bear the obligation to keep the data collected from patients secure. Gathering data may expose patients to the risk of data leakage.

Of course, we can leverage the existing vanilla distributed training method [82, 149, 147] to keep the institution's data local and share only the gradient with a central server. But the training of deep model requires many iterations to converge, leading to unacceptable **communication complexity** for vanilla distributed training. It is not secure neither as recent works [164, 159, 37, 154] have shown that pixel-level images can be recovered from the leaked gradient.

Recently, federated learning (FL) [74, 36, 145, 41, 43] have been proposed to tackle all the above issues (insufficient data, data privacy, training efficiency). In medical applications, we are most interested in the cross-silo federated learning where we have a limited number of participating clients compared with cross-device federated learning (e.g., mobile devices) [67, 94, 42]. Specifically, in each training round of FedAvg [98], the *de facto* algorithm for FL, each client will perform local training with the global model received from a central

server for multiple iterations. Then the server gathers all the local models from each client and averages them as the new global model. Nevertheless, for FedAvg and its variants, a non-negligible issue called "client drift" arises due to non-iid data distribution on different clients. The local models on different clients will gradually diverge from each other during the local training. Client drift can drastically jeopardize the training performance of the global model when the data similarity decreases (more non-iid) [50, 51]. Theoretically, it leads to a convergence rate more sensitive to the number of local training steps [155].

Throughout this paper, we refer to centralized training as gathering data from clients and then training the model. Note that centralized training is impractical as it violates data privacy, but offers a performance upper bound for FL algorithms. Despite numerous efforts and previous works, there is still a **generalization gap** between FL and the centralized training. In this paper, unlike any previous works, we propose a novel training framework called Federated Super Model (FedSM) to avoid confronting the difficult client drift issue at all for FL medical image segmentation tasks. In FedSM, instead of finding one global model that fits all clients' data distribution, we propose to produce personalized models to fit different data distributions well and a novel model selector to decide the closest model/data distribution for any test data.

We summarize our contributions as follows.

- We propose a novel training framework FedSM to avoid the client drift issue and close the generalization gap between FL and centralized training for medical segmentation tasks for the first time to the best of our knowledge.
- We propose a novel formulation for personalized FL optimization, and a novel personalized method called SoftPull to solve it in our framework FedSM. A rigorous convergence analysis with common assumptions in FL is given for the proposed method.
- Experiments in real-world FL medical image segmentation tasks validate our motivation and the superiority of our methods over existing FL baselines.

## 3.2 Related Works

Here we introduce existing different approaches to improve the model performance in FL with representative methods. First, the FL optimization problem is usually defined as $\min_w \frac{1}{K}\sum_{k=1}^{K} p_k L_{\mathcal{D}_k}(w)$, where the coefficient $p_k = \frac{n_k}{n}$, $n_k$ is the number of client $k$'s data, and the total number of data $n = \sum_{k=1}^{K} n_k$. $L_{\mathcal{D}_k}$ is the objective at client $k$ with its local data $\mathcal{D}_k$, and $w$ is the model weights.

**FedAvg.** In FedAvg, clients will receive the starting model $w_r$ from the server at training round $r$. Each client $k$ performs $E$ epochs of local training to update the local model to $w_{r+1}^{(k)}$ with the popular momentum SGD or Adam [72] optimizer depending on the application needs. Then the server gathers and averages the local models to $w_{r+1} = \frac{1}{K}\sum_{k=1}^{K} p_k w_{r+1}^{(k)}$.

**Restrict Local Training.** To discourage the local models from diverging due to non-iid data distribution, FedProx [87] proposes to add a proximal loss term $\|w_{r+1}^{(k)} - w_r\|_2^2$ to the objective function for client $k$. It implies that the local training will encourage $w_{r+1}^{(k)}$ to stay close to the starting point $w_r$, such that $\{w_{r+1}^{(k)}\}_{k\in\{1,2,\cdots,K\}}$ will be close to each other to alleviate the client drift issue.

**Correct Client Drift.** Motivated by variance reduction techniques in optimization such as SVRG [66], SAGA [26], inter-client variance reduction techniques [6, 70, 92] are proposed for FL by correcting the local training with the predicted local and global updating direction. These methods are usually tested with convex or simple non-convex models/objectives. For the practical training of complicated deep models, [27] shows that variance reduction techniques fail to perform well in that correcting the stochastic gradient with variance reduction usually does not hold in deep learning due to common augmentation tricks such as batch normalization [62] and dropout [123], etc.

**Personalization.** Personalized models are usually a fine-tuned version of the global model to better fit the local data distribution of a specific client. We can fine-tune the global model [139] on a client's local data like the local training, or following MAML-based personalized methods [129, 33, 65]. However, an intrinsic drawback of the personalized models is that they generalize poorly on other sites' data and unseen data. In this work, we focus on finding a model that generalizes as well as centralized training for all clients.

**Other Topics.** There are also many other emerging and interesting topics in FL, such as heterogeneous optimization [137, 87], fairness and robustness [100, 88, 86], clustered federated learning [39], etc. These topics are not directly related to our work but can be valuable for potential future extension. A recent work FedDG [94] requires sharing partial information of the data, therefore it breaks the data privacy constraint to some extent. In this work, we share only the model update information for maximal data privacy.

## 3.3 Methodology

In this section, we present our motivation and the proposed method that can close the generalization gap for FL medical image segmentation tasks in detail.

**Motivation.** In traditional FL, the goal is to collaboratively train one global model that generalizes well on all clients' joint data distribution. The client drift issue comes from the fact that we only have access to clients' local data distribution during the local training. It is hard to train a global model generalizing as well as centralized training due to this issue despite numerous existing works. In this work, however, we show that it is possible to get rid of the client drift issue. Specifically, we propose that

- for the test data, we search for the closest (i.e., the most similar) local data distribution from all clients (Section 3.3.1).
- we find a model with the best generalization performance on this selected local data distribution, and use it for the inference of the test data (Section 3.3.2).

### 3.3.1 New Framework: FedSM

The first motivation above motivates us to design a new and general FL framework FedSM, where we train a Federated "Super Model" consisting of the global model, personalized models, and a model selector. These components are illustrated in Figure 9 and we elaborate them as follows.

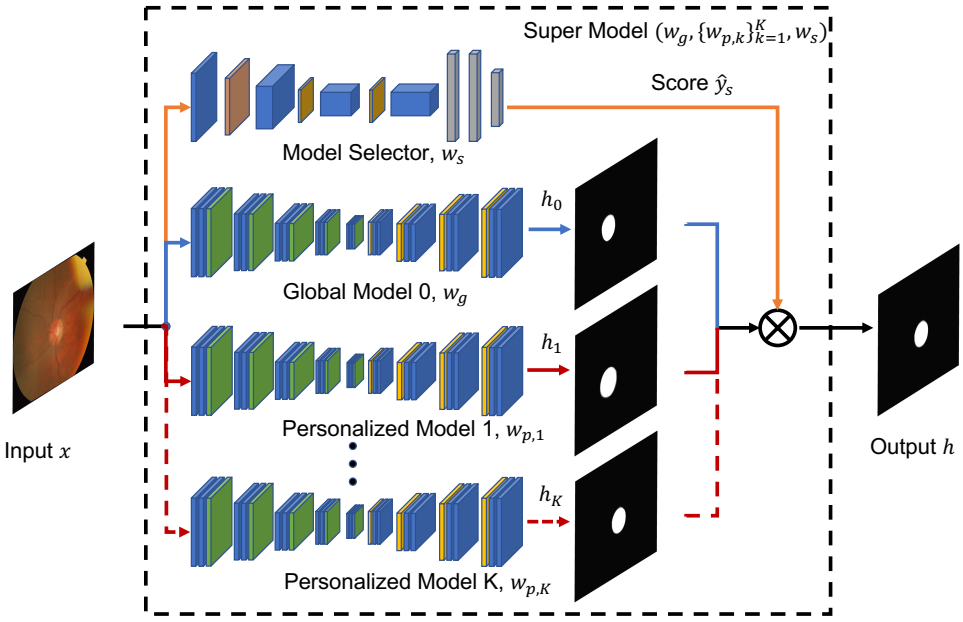


Figure 9: The proposed FedSM framework with "super model".

Global model $w_g$: the global model trained by FedAvg. It generalizes better than personalized models on the joint data distribution of all clients, but there is still a gap compared with centralized training. Suppose the model function is $f$ and we denote its output as $h_0 = f(w_g, x)$ for data $x$.

Personalized models $w_{p,k}$: the personalized models trained by any personalization FL training method. A personalized model usually generalizes better on local data than the global model. We denote its output as $h_k = f(w_{p,k}, x)$, where $k \in \{1, 2, \cdots, K\}$.

Model selector $w_s$: its goal is to determine the match between the unseen data input $x$ and each of the global/personalized models for inference. Specifically, it outputs a normalized prediction score vector $\widehat{y}_s$. The final output $h$ is determined by $\widehat{y}_s$ and $[h_0, h_1, \cdots, h_K]$. Suppose the candidate model set $\Omega \subseteq \{0, 1, 2, \cdots, K\}$, then $\sum_{k\in\Omega} \widehat{y}_{s,k} = 1$ and $h = \sum_{k\in\Omega} \widehat{y}_{s,k} h_k$. We discuss the potential training methods as follows.

#### 3.3.1.1 Ensemble

Suppose we already have the trained global model and personalized models. Given the FedSM framework as shown in Figure 9, a straightforward approach is to ensemble the outputs $[h_0, h_1, \cdots, h_K]$ from all models as the final output $h = \sum_{k=0}^{K} \widehat{y}_{s,k} h_k$. Let the ground truth of data $x$ be $y$ and the loss function be $L$. Then, we compute the loss $L(h, y)$ and update the model selector $w_s$ via FedAvg.

However, in practice we find it **hard to train** the model selector in this way in FL. The final performance can be even inferior to the global model. Let the desired value $y_s = \min_{\widehat{y}_s} L(\sum_{k=0}^{K} \widehat{y}_{s,k} h_k, y)$. We found that it was caused by the difficulty to train $\widehat{y}_s$ to the desired value $y_s$ by $\min_{w_s} L(\sum_{k=0}^{K} \widehat{y}_{s,k} h_k, y)$ as $w_s$ is the model weights to optimize. For each data input $x$, we may need many training steps to $\min_{w_s} L(\sum_{k=0}^{K} \widehat{y}_{s,k} h_k, y)$ such that $\widehat{y}_s$ will be close to $y_s$. However, it is unacceptable due to the large amount of computation cost.

Another issue of this approach is that we cannot start training the model selector until the training of the global model and personalized models finishes, which incurs **extra communication rounds** for FL.

#### 3.3.1.2 FedSM-extra

To tackle the **training difficulty** in ensemble, here we propose to compute

$$y_s = one\_hot(\arg\min_k \{L(h, h_k)\}_{k=0}^{K}), \tag{3–1}$$

where "one_hot" denotes one hot encoding. Then we compute the cross entropy loss $L_s(\widehat{y}_s, y_s)$ to update the model selector. In this way, the model selector is more clear about the desired value $y_s$. Thus it will be easier to train. We refer to this approach as FedSM-extra as it still needs extra communication rounds like the ensemble approach.

#### 3.3.1.3 FedSM

To address the issue of **extra training rounds**, the model selector needs to be trained together with the global model and personalized models. Nevertheless, from Eq. (3–1) we can see that the desired $y_s$ depends on the output of the trained global model and personalized models. Therefore, we need to decouple their dependency. As a further simplification, suppose the training data $x$ comes from the client $k \in \{1, 2, \cdots, K\}$, here we propose

$$y_s = one\_hot(k)\,. \tag{3–2}$$

Intuitively, the personalized model $k$ tends to generalize better on client $k$'s own local data. It is safe to set $y_s$ as the corresponding client index. Though theoretically, it may degrade the performance of Eq. (3–1), it is more practical due to no extra training rounds. We refer to this approach as FedSM which addresses all the issues raised by the ensemble.

### 3.3.2 New Personalization: SoftPull

In this section, we present a new personalized FL optimization formulation and a method, SoftPull, to solve it and produce personalized models for FedSM. We first present existing interpolation methods to tackle the insufficient local data issue.

Let the global dataset be $\mathcal{D}$. To tackle the insufficient local data issue, [97] proposes dataset interpolation for each client as $\min_{w_{p,k}} \lambda L_{\mathcal{D}_k}(w_{p,k}) + (1-\lambda) L_{\mathcal{D}}(w_{p,k})$, where coefficient $\lambda \in [0, 1]$. As client $k \in \{1, 2, \cdots, K\}$, it leads to $K$ optimization problems and is inefficient to solve. Besides, it is hard to acquire the information of the global dataset $D$ during the local training. [97] also proposes model interpolation $\min_{w_g, w_{p,k}, \lambda} \sum_{k=1}^{K} L_{\mathcal{D}_k}(\lambda w_{p,k} + (1-\lambda) w_g)$. To efficiently solve the model interpolation problem, APFL [29] proposes

$$w_g^* = \arg\min_{w_g} L_{\mathcal{D}}(w_g)\,, \tag{3–3}$$

$$w_{p,k}^* = \arg\min L_{\mathcal{D}_k}(\lambda w_{p,k} + (1-\lambda) w_g^*)\,, \tag{3–4}$$

$$w_{p.k} \leftarrow \lambda w_{p,k}^* + (1-\lambda) w_g^*\,. \tag{3–5}$$

**Motivation.** We observe that model interpolation tries to find an appropriate combination between the FL global and local models. When the local data distribution is not similar

to the global data distribution at all, we expect $\lambda \to 1$. When they are similar, we expect $\lambda \to \frac{1}{K}$ to leverage the global data information to improve the local generalization as the local dataset is small. Nevertheless, the formulation of APFL has two potential drawbacks:

- The involved global model $w_g^*$ may not generalize well on $\mathcal{D}$ and $\mathcal{D}_k$, but will affect the FL training.
- What objective function it is exactly optimizing is not clear.

In our problem formulation, we first suppose $w_k^*$ is the local optimum of client $k$:

$$w_k^* = \arg\min_w L_{\mathcal{D}_k}(w)\,. \tag{3–6}$$

However, local optimum $w_k^*$ may not generalizes well due to lack of local training data. Instead of interpolating the global and local optimum, we propose that the desired *personalized optimum $w_{p,k}^*$ is an interpolation between the local optimum of client $k$ and other clients' personalized optima*:

$$w_{p,k}^* = \lambda w_k^* + (1-\lambda)\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{p,k'}^*\,. \tag{3–7}$$

The new interpolation avoids the global model and guarantees that the interpolated model is the optimum to some explicit objective function, as opposed to APFL. In fact, the personalized optimum $w_{p,k}^*$ is also an interpolation between the local optimum of client $k$ and other clients' local optimum because Eq. (3–7) is identical to

$$w_{p,k}^* = \lambda w_k^* + (1-\lambda)\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{k'}^*\,. \tag{3–8}$$

However, Eq. (3–7) is better to help us to find what objective function we are optimizing as we can turn it to

$$w_k^* = \frac{1}{\lambda}w_{p,k}^* - \frac{1-\lambda}{\lambda}\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{p,k'}^*\,. \tag{3–9}$$

Compare it with Eq. (3–6) and we immediately have $\{w_{p,k}^*\}_{k=1}^K$ as the solution to the optimization problem

$$\min_{\{w_{p,k}\}}\sum_{k=1}^{K} L_{\mathcal{D}_k}(\frac{1}{\lambda}w_{p,k} - \frac{1-\lambda}{\lambda}\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{p,k'})\,. \tag{3–10}$$

To solve the proposed new personalized FL optimization problem Eq. (3–10), we propose a new method, SoftPull ($\lambda \in [\frac{1}{K}, 1]$), with the simplification of substituting $w_k^*$ with the locally trained model in Eq. (3–7), that is, after each training round at the server,

$$w_{p,k} \leftarrow \lambda w_{p,k} + (1-\lambda)\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{p,k'}\,. \tag{3–11}$$

The corresponding algorithm is summarized in Algorithm 2, line 16. When $\lambda = \frac{1}{K}$, it reduces to the "hard" averaging in FedAvg. To analyze the convergence, we start with common assumptions as follows.

**Assumption 3.3.1.** *(Lipschitz Smooth) The loss function $L_{\mathcal{D}_k}$ is L-smooth, i.r., $\forall w_1, w_2 \in \mathbb{R}^d$, we have*

$$\|\nabla L_{\mathcal{D}_k}(w_1) - \nabla L_{\mathcal{D}_k}(w_2)\|_2^2 \leq L\|w_1 - w_2\|_2^2\,. \tag{3–12}$$

**Assumption 3.3.2.** *(Bounded Variance) The stochastic gradient $\nabla L_{\mathcal{D}_k}(w, x)$ has bounded variance $\forall w \in \mathbb{R}^d$:*

$$\mathbb{E}\|\nabla L_{\mathcal{D}_k}(w, x) - \nabla L_{\mathcal{D}_k}(w)\|_2^2 \leq \sigma^2\,. \tag{3–13}$$

*where $\mathbb{E}$ is an expectation over $x \in \mathcal{D}_k$.*

**Assumption 3.3.3.** *[109] The gradient $\nabla L_{\mathcal{D}_k}(w)$ has bounded value $\forall w \in \mathbb{R}^d$:*

$$\|\nabla L_{\mathcal{D}_k}(w)\|_2^2 \leq G^2. \tag{3–14}$$

**Theorem 3.3.1.** *Suppose Assumptions 3.3.1, 3.3.2, and 3.3.3 exist. Let the proposed objective in Eq. (3–10) be F, superscript $(r, m)$ denote the global iteration, and $\overline{w}$ denote the average, then*

$$\begin{aligned}
&\frac{1}{KRM}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\sum_{k=1}^{K}\mathbb{E}\|\nabla_{w_{p,k}^{r,m}} F\|_2^2 \qquad (3–15)\\
&= \mathcal{O}\Big(\frac{1}{\eta RM\lambda^2} + \frac{(1-\lambda)^2}{KRM\eta^2\lambda^2}\sum_{K=1}^{K}\sum_{r=0}^{R-1}\mathbb{E}\|w_{p,k}^{r,M} - \overline{w}_{p,k}^{r,M}\|_2^2\\
&\quad + \frac{(1-\lambda)^2}{KRM\lambda^4}\sum_{k=1}^{K}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\mathbb{E}\|w_{p,k}^{r,m} - \overline{w}_{p,k}^{r,m}\|_2^2\Big)\\
&= \mathcal{O}\Big(\frac{1}{\eta RM\lambda^2} + \frac{M\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^2} + \frac{M^2\eta^2\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^4}\Big)\,.
\end{aligned}$$

*If* $\eta = \mathcal{O}(\frac{1}{\sqrt{RM}})$ *and* $M = \mathcal{O}(R^{\frac{1}{3}})$*, its convergence rate is* $\mathcal{O}(\frac{1}{\sqrt{RM}})$ *with a convergence error* $\mathcal{O}(\frac{M\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^2})$.

**Remark 3.3.1.** *When the data similarity is low among clients, we should set a larger* $\lambda$ *to reduce the effect of* $\|w_{p,k}^{r,m} - \overline{w}_{p,k}^{r,m}\|_2^2$ *and ensure the convergence rate. It is intuitively valid as the client has less to learn from other clients.*

**Remark 3.3.2.** $\lambda \downarrow$ *and the convergence error* $\uparrow$*, but it does not mean worse generalization because we do not want to overfit local data. We will empirically tune and validate it.*

The proof can be found in Appendix C.3.

### 3.3.3 All Together

We summarize the proposed SoftPull method to train personalized models and the FedSM framework consisting of the model selector, global model, and personalized models in Algorithm 2. Compared with FedAvg, the communication cost of each training round is $2w_g + w_s$ for FedSM. We note that some methods such as Scaffold [70] have a cost of $2w_g$. After the training, the server sends the super model ($w_g$, $\{w_{p,k}\}_{k=1}^K$, $w_s$) to each client for inference, which incurs only a one-time communication cost.

For the FedSM inference in Algorithm 3, we propose a heuristic technique that the model selector selects the global model when its confidence is low, because we do not have label 0 in Eq. (3–2) (the global model) during training. Intuitively, if the test data is not similar to any local data distribution, the global model should be a better choice for its inference, in that it covers the joint data distribution while the personalized model covers only one local data distribution. It also guarantees that FedSM is at least not worse than the global model from FedAvg with an appropriate threshold $\gamma$.

For FedSM-extra, both the training and inference algorithms are the same except for the determination of $y_s$, the extra training rounds, and no need for the threshold $\gamma$. More details are available in Appendix C.2.

Table 9: Retinal Dataset: number of data (2D image) in each client. The data sources from client 1 to 6 are Drishti-GS1 [121], RIGA [10] BinRushed, RIGA Magrabia, RIGA MESSIDOR, RIM-ONE [35], and REFUGE [4] respectively. Global refers to the data from all clients.

| Client | 1 | 2 | 3 | 4 | 5 | 6 | Global |
|---|---|---|---|---|---|---|---|
| Train | 50 | 98 | 47 | 230 | 80 | 400 | 905 |
| Val | 25 | 49 | 24 | 115 | 40 | 200 | 453 |
| Test | 26 | 48 | 23 | 115 | 39 | 200 | 451 |

### 3.4 Experiments

We validate our proposed method on three real-world FL medical image segmentation tasks: retinal disc & cup from 2D fundus images, and prostate segmentation from 3D MR images. The global and personalized model architecture is 2D U-Net [112], while the model selector architecture is VGG-11 [120]. We randomly split the data to train/validation/test with a ratio of 0.5/0.25/0.25. The image data are resized to $256 \times 256$. The local training epoch is 1 and the total training rounds is 150. Most methods converge in 100 rounds. But for FedSM-extra, we train the global and personalized models for 100 rounds and the model selector for an extra 50 rounds. The loss function is Dice loss and the test metric is Dice coefficient. The base optimizer is Adam with $\beta = (0.9, 0.999)$. We tune the best learning rate for all methods and the threshold $\gamma$ for FedSM. For prostate segmentation, in particular, the image data are 3D but we take the 2D slices and perform 2D segmentation. Each experiment repeatedly runs 3 times and we report the mean value.

The dataset information is summarized in Table 9 and 10. Overall, the retinal dataset features lower data similarity among clients (stronger non-iid). The images may differ in position, color, brightness, background ratio, etc. While the prostate dataset has a higher data similarity as the images mostly differ in brightness (see Appendix C.1).

We compare FedSM and FedSM-extra with baselines (1) Centralized: centralized training, which is the upper bound but prohibited in FL, (2) Local: local training on one client,

Table 10: Prostate Dataset: number of data (2D slices) in each client. The data sources from client 1 to 6 are I2CVB [80], MSD [11], NCI_ISBI_3T, NCI_ISBI_DX [1], Promise12 [2], and ProstateX [3] respectively. Global refers to the data from all clients.

| Client | 1 | 2 | 3 | 4 | 5 | 6 | Global |
|---|---|---|---|---|---|---|---|
| Train | 153 | 404 | 464 | 361 | 609 | 1179 | 3170 |
| Val | 77 | 215 | 219 | 162 | 289 | 582 | 1544 |
| Test | 61 | 245 | 198 | 150 | 329 | 532 | 1515 |

(3) FedAvg [98], the *de facto* FL method, (4) FedProx [87], and (5) Scaffold [70].

### 3.4.1 General Results

We compare the training curves of different methods in Figure 10. The centralized training upper bound is plotted as a horizontal dash line. We can see that the proposed FedSM is the only FL method to close the validation gap to centralized training. FedSM is even better than centralized training on the retinal cup segmentation task, due to the proposed SoftPull personalization method. Note that we can not show the training curve of FedSM-extra as its model selector has to be trained in the extra training rounds.

We summarize the testing numbers in Table 11 and 12. For retinal segmentation, FedSM slightly improves centralized training regarding the client average Dice and global Dice by 0.2% and 0.1% respectively, while FedAvg shows a decrease of 1.9% and 0.9%. The FedSM-extra shows the same performance as FedSM, validating the proposed simplification from Eq. (3–1) to Eq. (3–2). For prostate segmentation, similar patterns can be observed. But the gap becomes smaller due to higher data similarity among clients.

For retinal segmentation, FedSM not only outperforms centralized training for client 3 but also matches centralized training for the other clients. However, FedAvg is inferior to centralized training for clients 1, 2, 3, and 5 where the local dataset size is smaller. What's more, FedAvg shows similar test Dice performance to local training for clients 1 and 2, and is even inferior to local training for clients 3 and 5. Therefore, those clients do not benefit

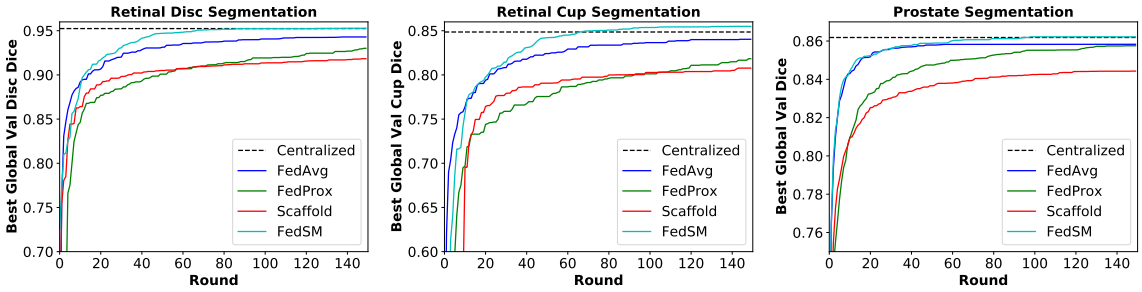


Figure 10: Training curves comparison. The curves are non-decreasing because we record the best result during training.

from FL via FedAvg, and may not be willing to join the FL system.

We also observe that local training does not generalize well on other clients' data, which is critical as it will perform poorly for patients from other clients (medical institutions). Centralized training improves the local training on the local dataset, especially for clients with insufficient data.

### 3.4.2 Validate Motivation

**Validate FedSM.** Recall that our first motivation is to find the closest local data distribution for the test data. In FedSM, we first plot the TSNE map of the features extracted from the model selector in Figure 11. To validate that the model selector can fulfill our motivation, we sequentially choose client $k \in \{1, 2, \cdots, 6\}$ as the unseen client to test and FL train the model with clients $\{1, 2, \cdots, 6\}/\{k\}$. We set the threshold $\gamma = 0$ to let the model selector select from the personalized models. We summarize the frequency in Table 13. We can see that the model selector tends to select the personalized models of clients 3 and 5 for client 6, which also matches Figure 11 and the local training results in Table 11 that clients 3 and 5 are more similar to client 6. Similar patterns can be observed for the other clients. Therefore, the model selector indeed fulfills our motivation. Note that to validate the model selector, we cannot let the unseen client $k$ join the FL system. Because in that case, the model selector tends to select its own personalized model.

Table 11: (low data similarity) Test Dice coefficient comparison of retinal segmentation. "Client $k$ Local" refers to local training on client $k$. The first row refers to the performance on client 1∼6's test data, their average, and the performance on all clients' test data. We report the average of disc and cup Dice coefficients here. We bold the best FL numbers. See Appendix C.4 for their separate numbers and the visual comparison of segmentation.

| Method | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Client Avg Dice | Global Dice |
|---|---|---|---|---|---|---|---|---|
| Centralized | 0.9161 | 0.8760 | 0.8758 | 0.9022 | 0.8510 | 0.9179 | 0.8898 | 0.9014 |
| Client 1 Local | 0.8835 | 0.3331 | 0.7345 | 0.4933 | 0.3408 | 0.7015 | 0.5811 | 0.5902 |
| Client 2 Local | 0.2346 | 0.8620 | 0.0886 | 0.7751 | 0.1791 | 0.4106 | 0.4250 | 0.5050 |
| Client 3 Local | 0.8337 | 0.3402 | 0.8766 | 0.6010 | 0.3644 | 0.7794 | 0.6326 | 0.6594 |
| Client 4 Local | 0.5108 | 0.8574 | 0.3457 | 0.9008 | 0.2361 | 0.6822 | 0.5888 | 0.6910 |
| Client 5 Local | 0.5241 | 0.1584 | 0.3953 | 0.2039 | 0.8223 | 0.6222 | 0.4544 | 0.4662 |
| Client 6 Local | 0.7908 | 0.6649 | 0.7325 | 0.7681 | 0.3742 | 0.9150 | 0.7076 | 0.7877 |
| FedAvg | 0.8847 | 0.8679 | 0.8667 | 0.9015 | 0.7877 | 0.9172 | 0.8710 | 0.8923 |
| FedProx | 0.8635 | 0.8522 | 0.8547 | 0.8952 | 0.6852 | 0.9095 | 0.8434 | 0.8749 |
| Scaffold | 0.8380 | 0.8513 | 0.8215 | 0.8935 | 0.5671 | 0.9130 | 0.8141 | 0.8625 |
| FedSM | **0.9132** | **0.8769** | **0.8865** | **0.9041** | **0.8483** | **0.9195** | **0.8914** | **0.9028** |
| FedSM-extra | **0.9134** | **0.8763** | **0.8841** | **0.9038** | **0.8483** | **0.9172** | **0.8905** | **0.9007** |

In Table 13, we also validate that the threshold $\gamma$ helps improve the performance of FedSM for the unseen data. For those unseen data with low confidence from the model selector, a larger $\gamma$ increases the chance of the global model to be selected because maybe none of the personalized models is suitable. By choosing a proper $\gamma$, we can further improve the Dice of unseen clients 5 and 6 by 3%.

**Validate SoftPull.** Recall that our second motivation is to find a model generalizing well on the local data distribution even with insufficient local data. To achieve it we propose a new personalized FL optimization formulation with SoftPull to solve it. The Remark 3.3.1 of the theoretical analysis can be empirically validated by the fact that the best $\lambda = 0.7$ (closer to 1) for the retinal segmentation task with lower data similarity, and that the best $\lambda = 0.3$ (closer to $\frac{1}{K} = \frac{1}{6} = 0.17$) for the prostate segmentation task with higher data similarity.

Next, we will validate Remark 3.3.2 that a proper $\lambda$ may lead to a convergence error, but in the meantime may improve the generalization by preventing overfitting the small local

Table 12: (high data similarity) Test Dice coefficient comparison of prostate segmentation. We bold the best FL numbers. See Appendix C.4 for the visual comparison.

| Method | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Client Avg Dice | Global Dice |
|---|---|---|---|---|---|---|---|---|
| Centralized | 0.9018 | 0.8583 | 0.8702 | 0.8844 | 0.8800 | 0.8474 | 0.8737 | 0.8651 |
| Client 1 Local | 0.8582 | 0.3886 | 0.4476 | 0.2849 | 0.3830 | 0.4697 | 0.4720 | 0.4336 |
| Client 2 Local | 0.7166 | 0.7669 | 0.8317 | 0.7341 | 0.6156 | 0.7754 | 0.7401 | 0.7403 |
| Client 3 Local | 0.6470 | 0.8541 | 0.8549 | 0.6735 | 0.6591 | 0.7519 | 0.7401 | 0.7496 |
| Client 4 Local | 0.4515 | 0.6566 | 0.6700 | 0.8518 | 0.4558 | 0.6267 | 0.6187 | 0.6148 |
| Client 5 Local | 0.8198 | 0.7751 | 0.8469 | 0.8029 | 0.8038 | 0.7928 | 0.8069 | 0.8016 |
| Client 6 Local | 0.8555 | 0.7965 | 0.8260 | 0.7206 | 0.6478 | 0.8466 | 0.7822 | 0.7809 |
| FedAvg | 0.8775 | 0.8575 | 0.8700 | 0.8802 | 0.8717 | 0.8532 | 0.8684 | 0.8638 |
| FedProx | **0.8948** | 0.8511 | 0.8722 | 0.8803 | 0.8668 | 0.8513 | 0.8694 | 0.8621 |
| Scaffold | 0.8500 | 0.8440 | 0.8570 | 0.8423 | 0.8431 | 0.8412 | 0.8463 | 0.8446 |
| FedSM | **0.8946** | **0.8596** | **0.8786** | **0.8898** | **0.8817** | **0.8535** | **0.8763** | **0.8692** |
| FedSM-extra | 0.8886 | **0.8584** | **0.8766** | **0.8880** | **0.8760** | **0.8542** | **0.8736** | **0.8673** |

dataset with the help of other clients. We plot the 1D loss surface near the trained model by computing the loss along 10 randomly sampled unit vector directions (Figure 12), following existing works [63, 45]. It is interesting to see that local training overfits the training data and leads to a sharp local training optimum, which is known to generalize worse [63, 151, 45]. On the contrary, we observe an "over-regularization" effect for FedAvg as it has an even flatter training optimum than centralized training and a large convergence error (worse training loss), which also leads to a worse generalization performance. Indeed, averaging model in FedAvg can be regarded as a sort of implicit regularization. In comparison, SoftPull achieves a tunable flatness by choosing a proper $\lambda$. Even if it leads to a convergence error, it achieves generalization performance better than local training and comparable to centralized training.

### 3.4.3 Ablation Study

**Personalization.** We compare personalization methods in FedSM in Table 14, including (1) FT (local fine-tuning) [139], (2) APFL [29], (3) Per-FedAvg [33], and (4) Per-FedMe [129]. All methods' hyper-parameters are tuned for best results. SoftPull is the better interpolation method among them, outperforming APFL by 0.62% regarding the global Dice coefficient.

Table 13: (retinal segmentation, Dice = average of disc and cup Dice coefficients) Model selection frequency from the model selector when FL train with clients $\{1, 2, \cdots, 6\}/\{k\}$ and test on the **unseen** client $k \in \{1, 2, \cdots, 6\}$. From left to right, GM denotes the global model and PM denotes the personalized model $\{1, 2, \cdots, 6\}/\{k\}$. The model selection frequency with the best $\gamma$, and the more detailed Dice results can be found in Appendix C.4. Note GM is never selected as the Threshold $\gamma$ is intentionally set to 0.

| Unseen Client $k$ | Threshold $\gamma$ | GM | PM1 | PM2 | PM3 | PM4 | PM5 | PM6 | Dice | Best $\gamma$, Dice |
|---|---|---|---|---|---|---|---|---|---|---|
| Client $k = 6$ | 0 | 0 | 0.02 | 0 | 0.35 | 0 | 0.63 | N/A | 0.8587 | 1, 0.8906 |
| Client $k = 5$ | 0 | 0 | 0.31 | 0.03 | 0 | 0.61 | N/A | 0.05 | 0.4015 | 0.9, 0.4304 |
| Client $k = 4$ | 0 | 0 | 0 | 1.00 | 0 | N/A | 0 | 0 | 0.8869 | <0.95, 0.8870 |
| Client $k = 3$ | 0 | 0 | 0 | 0.57 | N/A | 0 | 0 | 0.43 | 0.8441 | <0.9, 0.8446 |
| Client $k = 2$ | 0 | 0 | 0 | N/A | 0 | 0.92 | 0.08 | 0 | 0.8409 | <1, 0.8409 |
| Client $k = 1$ | 0 | 0 | N/A | 0 | 1.00 | 0 | 0 | 0 | 0.8839 | <0.99, 0.8839 |

It also outperforms the best counterpart by 0.44%.

**Interpolation Coefficient $\lambda$.** We explore different $\lambda$ values of FedSM in Table 15 and $\lambda = 0.7$ performs the best.

## 3.5 Conclusion

In this work, we propose FedSM to close the generalization gap between FL and centralized training for medical image segmentation for the first time. The empirical study on real-world medical FL tasks validates our theoretical analysis and motivation to avoid the client drift issue.

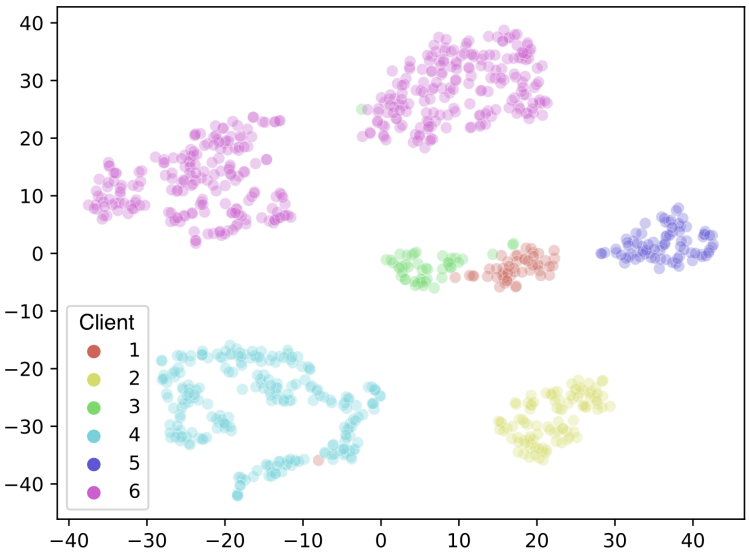


Figure 11: TSNE map of the features extracted form the model selector on retinal segmentation task.

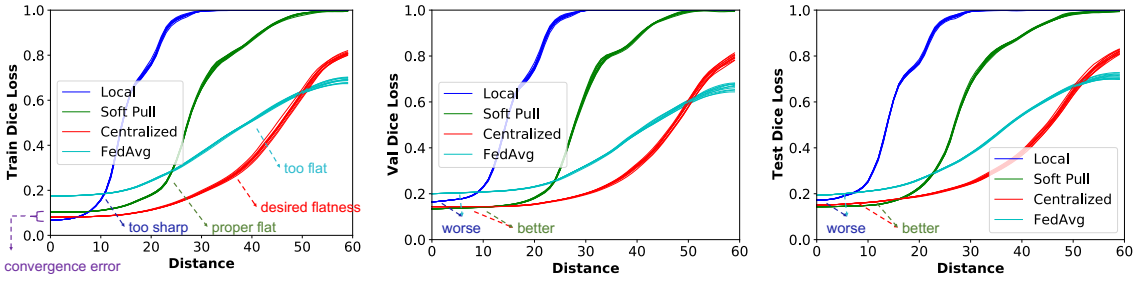


Figure 12: The 1D loss surface near the models trained by different methods on Client 5's data in retinal segmentation.

**Algorithm 2** FedSM training.

**Input:** local dataset $\mathcal{D}_k$, rounds $R$, number of sites $K$, learning rate $\eta$, $\eta_s$, coefficient $\lambda$, client weight $\frac{n_k}{n}$.

**Initialize:** global model $w_g$, personalized model $w_{p,k}$, model selector $w_s$, base optimizer OPT$(\cdot)$

**for** round $r = 1, 2, \cdots, R$ **do**

  SERVER: send models ($w_g$, $w_{p,k}$, $w_s$) to client $k$.

  **for** CLIENT $k \in \{1, 2, \cdots, K\}$ in parallel **do**

    initialize $w_{g,k} \leftarrow w_g$, $w_{s,k} \leftarrow w_s$

    **for** batch $(x, y) \in \mathcal{D}_k$ **do**

      $w_{g,k} \leftarrow \text{OPT}(w_{g,k}, \eta, \nabla_{w_{g,k}} L(f(w_{g,k}; x), y))$

      $w_{p,k} \leftarrow \text{OPT}(w_{p,k}, \eta, \nabla_{w_{p,k}} L(f(w_{p,k}; x), y))$

      *// $y_s$ from Eq. (3–2)*

      $w_{s,k} \leftarrow \text{OPT}(w_{s,k}, \eta_s, \nabla_{w_{s,k}} L_s(f_s(w_{s,k}; x), y_s))$

    **end for**

    send ($w_{g,k}$, $w_{p,k}$, $w_{s,k}$) to server

  **end for**

  SERVER: $w_g, w_s \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{g,k}, \sum_{k=1}^{K} \frac{n_k}{n} w_{s,k}$

  SERVER: $\forall k \in \{1, 2, \cdots, K\}$, $w_{p,k} \leftarrow \lambda w_{p,k} + (1-\lambda) \frac{1}{K-1} \sum_{k'=1, k' \neq k}^{K} w_{p,k'}$ *// SoftPull*

**end for**

**Output:** model ($w_g$, $\{w_{p,k}\}_{k=1}^{K}$, $w_s$)

Table 14: FedSM with different personalization method in retinal segmentation. Dice = average of disc and cup Dice coefficients.

| Method | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Client Avg Dice | Global Dice |
|---|---|---|---|---|---|---|---|---|
| FT [139] | 0.9087 | 0.8703 | 0.8877 | 0.9003 | 0.8409 | 0.9151 | 0.8875 | 0.8984 |
| APFL [29] | 0.9083 | 0.8640 | 0.8794 | 0.8969 | 0.8416 | 0.9152 | 0.8842 | 0.8966 |
| Per-FedAvg [33] | 0.9051 | 0.8559 | 0.8708 | 0.8954 | 0.8031 | 0.9119 | 0.8737 | 0.8900 |
| Per-FedMe [129] | 0.9084 | 0.8646 | 0.8822 | 0.8980 | 0.8211 | 0.9162 | 0.8818 | 0.8957 |
| SoftPull | **0.9132** | **0.8769** | **0.8865** | **0.9041** | **0.8483** | **0.9195** | **0.8914** | **0.9028** |

**Algorithm 3** FedSM inference.

**Input:** data $x$, model $(w_g, \{w_{p,k}\}_{k=1}^K, w_s)$, threshold $\gamma$

$\widehat{y}_s = f_s(w_s; x)$

**if** $\max(\widehat{y}_s) > \gamma$ **then**

  $k = \arg\max(\widehat{y}_s) \in \{1, 2, \cdots, K\}$ *// high confidence*

  $\widehat{y} = f(w_{p,k}; x)$

**else**

  $\widehat{y} = f(w_g; x)$ *// low confidence*

**end if**

**Output:** $\widehat{y}$

Table 15: FedSM with different coefficient $\lambda$ in retinal segmentation. Dice = average of disc and cup Dice coefficients.

| $\lambda$ | 0.1 | 0.3 | 0.5 | **0.7** | 0.9 |
|---|---|---|---|---|---|
| Client Avg | 0.8808 | 0.8859 | 0.8895 | **0.8914** | 0.8882 |
| Global | 0.8964 | 0.8896 | 0.9019 | **0.9028** | 0.9001 |

# 4.0 A New Optimizer with Data Privacy

## 4.1 Introduction

With deep learning models becoming prevalent but data-hungry, data privacy emerges as an important issue. Federated learning (FL) [74] 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 [67, 41] problem, which has more and more real-world applications, such as the collaborative learning on health data across multiple medical centers [94, 42] 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, [18] 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 [98]. 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 [124, 155]. Another parallel line of works is to compress the communication to reduce the volume of the message [113, 71, 36, 149, 147, 148]. 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 [50, 51], 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 [24] to accelerate optimization, momentum SGD has shown its power in training many deep learning models in various tasks [127, 146, 81]. Local momentum SGD [155] (*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 local momentum buffer to zero at the end of each training round (FedAvgLM-Z) [117, 136, 137]. [51] and [61] 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 [21] for speech models and in [138] for distributed training, but neither of them has applied it to FL. [51] and [61] empirically showed the ability of FedAvgSM to tackle client drift in FL, and [138] and [61] 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.

### 4.2 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: $\alpha$, $\eta$;
- Momentum fusion constant $\beta$; Server, local momentum constant: $\mu_s$, $\mu_l$;
- Server, local momentum buffer: $\mathbf{m}_r$, $\mathbf{m}_{r,p}^{(k)}$; Server, (average) local model: $\mathbf{x}_r$, $\mathbf{x}_{r,p}^{(k)}$ ($\overline{\mathbf{x}}_{r,p}^{(k)}$);
- 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.** SOTA 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.** [109] applied the idea of using server statistics as in server momentum SGD to adaptive optimizers including Adam [72], AdaGrad [30], and Yogi [157]. [109] 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 [70, 6] 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 [66, 26] can be seen as intra-client variance reduction. Scaffold [70] 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 [92] 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 [69] 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 [24] but their algorithm is based on Polyak's momentum. Though theoretically appealing, the variance reduction technique has shown to be ineffective in practical neural networks' optimization [27]. [27] showed that common tricks such as data augmentation, batch normalization [62], and dropout [123] broke the transformation locking and deviated practice from variance reduction's theory.

**Other.** There are some other settings of FL including heterogeneous optimization [87, 137], fairness [100, 88, 86], personalization [129, 33, 65, 118], etc. These different settings, variance reduction techniques, and server statistics can sometimes be combined. Here we focus on momentum-based FL methods.

**Algorithm 4** FL with double momenta.

1: **Input:** local training steps $P \geq 1$, #rounds $R$, #clients $K$, server (local) learning rate $\alpha$ ($\eta$), server (local) momentum constant $\mu_s$ ($\mu_l$), momentum fusion constant $\beta$.
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:** $\mathbf{x}_R$

### 4.3 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 4 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 4 where **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 4 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 $\beta P$ and learning rate $\eta$ to the local model before the local training starts. $\beta$ 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 $\beta$ instead of $\beta P$. 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. FedAvgSLM(-Z), the naive combination, 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 4 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.

### 4.4 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 4 for less communication. Please refer to the Supplementary

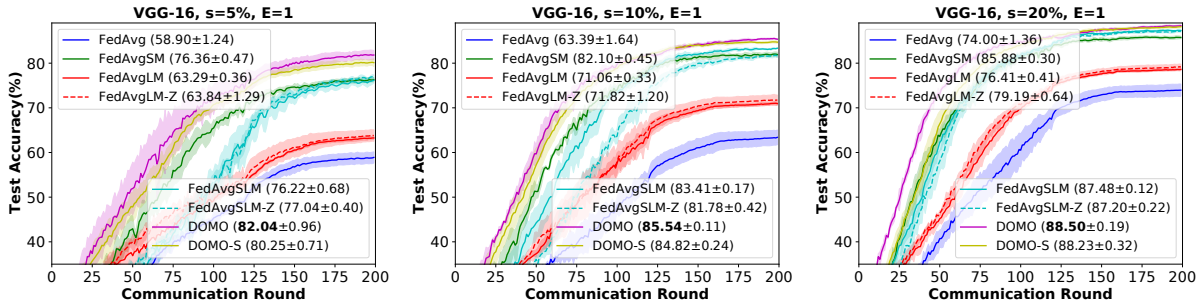


Figure 13: CIFAR-10 training curves using the VGG-16 model with various data similarity $s$.

Material for proof details.

We consider non-convex smooth objective function satisfying Assumption 4.4.1. We also assume that the local stochastic gradient is an unbiased estimation of the local full gradient and has a bounded variance in Assumption 4.4.2. Furthermore, we bound the *non-i.i.d.* data distribution across clients in Assumption 4.4.3, which is widely employed in existing works such as [155, 109, 138, 69]. $G$ 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 4.4.1.** *(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 4.4.2.** *(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 4.4.3.** *(Bounded Non-i.i.d. Distribution [155, 109, 138, 69]) 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*

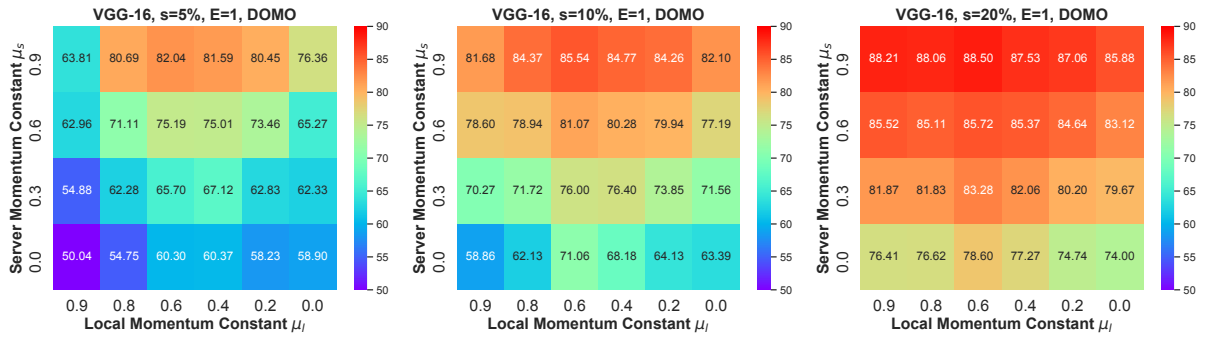


Figure 14: CIFAR-10 test accuracy (%) with various sever momentum constant $\mu_s$ and local momentum constant $\mu_l$. $\mu_s = 0$ corresponds to FedAvgLM, $\mu_l = 0$ corresponds to FedAvgLM, $\mu_s = 0 \,\&\, \mu_l = 0$ corresponds to FedAvg, and $\mu_s \neq 0 \,\&\, \mu_l \neq 0$ corresponds to DOMO.

*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 4.4.1.** *(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*

$$\boldsymbol{z}_{r,p+1} = \boldsymbol{z}_{r,p} - \frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1} \nabla F^{(k)}(\boldsymbol{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)}) \tag{4–1}$$

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 4.4.1 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

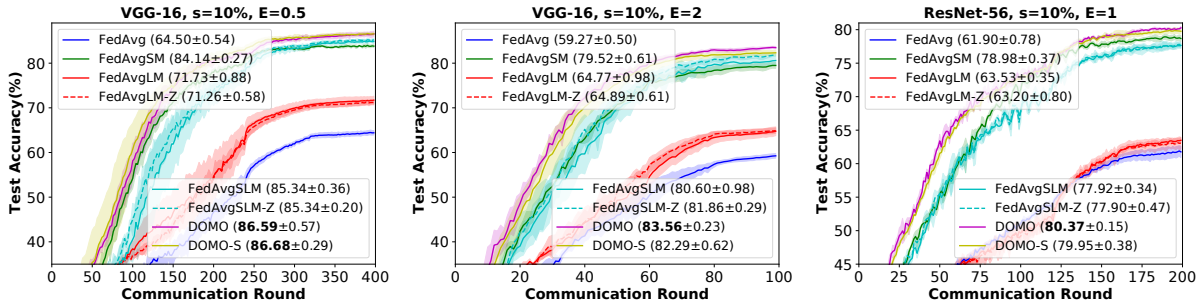


Figure 15: Left and Middle: CIFAR-10 training curves using the VGG-16 model with various local epoch $E$. $E = 1$ has been shown in the middle plot of Figure 13 and is not repeatedly shown here. Right: CIFAR-10 training curves using the ResNet-56 model.

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 4.4.2.** *(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*

$$\begin{aligned}\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|\boldsymbol{z}_{r,p} - \overline{\boldsymbol{x}}_{r,p}\|_2^2 \leq \frac{\eta^2}{1-\mu_l}\Big(\sum_{p=0}^{P-1}\frac{h^2\mu_l^p}{1-\mu_l^P}\Big)\cdot \\ \sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\Big\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F^{(k)}(\boldsymbol{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})\Big\|_2^2.\end{aligned} \tag{4–2}$$

**Theoretical Difficulty for FedAvgSLM(-Z) but Addressed by DOMO.** From Lemma 4.4.2, 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

$$\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|P\mathbf{m}_r\|_2^2 = P^2\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\|\mathbf{m}_r\|_2^2. \tag{4–3}$$

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

| $\beta$ | $\alpha = 1.0$ | $\beta$ | $\alpha = 1.0$ | $\alpha$ | $\beta = 0.9$ | $\alpha$ | $\beta = 0.9$ |
|---|---|---|---|---|---|---|---|
| 1.0 | 81.89 ± 0.40 | 0.6 | 81.60 ± 0.40 | 1.0 | **85.54** ± 0.11 | 0.6 | 83.76 ± 0.28 |
| 0.9 | **85.54** ± 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 |

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. (4–2). 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 4.4.2, 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 $\alpha$ and $\beta$ 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

Table 17: SVHN test accuracy (%) when training ResNet-20.

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

Table 18: CIFAR-100 test accuracy (%). Second row: VGG-16. Third row: ResNet-56.

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

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 4.4.2 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 4.4.1.** *(Convergence of DOMO) Assume Assumptions 4.4.1, 4.4.2 and 4.4.3 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 4 line 5, we have*

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

**Complexity.** According to Theorem 4.4.1, 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 $\frac{1}{RP}\sum_{r=0}^{R-1}\sum_{p=0}^{P-1}\mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 = \mathcal{O}(K^{-\frac{1}{2}}R^{-\frac{1}{2}}P^{-\frac{1}{2}})$ 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 $\mu_s$ in Theorem 4.4.1 because it is eliminated in Lemma 4.4.2 by setting $\beta = \mu_s$.

## 4.5 Experimental Results

### 4.5.1 Settings

All experiments are implemented using PyTorch [105] 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.*, Algorithm 4 Option III which is essentially FedAvgSM + FedAvgLM-Z), 7) DOMO (*i.e.*, Algorithm 4 Option I), and 8) DOMO-S (*i.e.*, Algorithm 4 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 $\mu_l$ is selected from {0.9, 0.8, 0.6, 0.4, 0.2}. We select the server momentum constant $\mu_s$ from {0.9, 0.6, 0.3}. The base learning rate is selected from {..., $4 \times 10^{-1}$, $2 \times 10^{-1}$, $1 \times 10^{-1}$, $5 \times 10^{-2}$, $1 \times 10^{-2}$, $5 \times 10^{-3}$, ...}. The server learning rate $\alpha$ is selected from $\{0.2, 0.4, 0.6, 0.8, 0.9, 1.0\}$. The momentum fusion constant $\beta$ is selected from $\{0.2, 0.4, 0.6, 0.8, 0.9, 1.0\}$. Following [70, 51, 137], 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 [70] to simulate the *non-i.i.d.* data distribution. Specifi-

cally, 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 [137]. 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 [120] and ResNet-56 [48] models on CIFAR-10/100[1] [76], and ResNet-20 on SVHN[2] image classification tasks. Please refer to the Supplementary Material for details.

### 4.5.2 Performance

We illustrate the experimental results in Figures 13, 14, and 15 with test accuracy $(mean \pm std)$ reported in the brackets of the legend, and Table 4.4 and 4.4. 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 15, 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 [51, 61].

**Varying Data Similarity $s$.** We plot the training curves under different data similarity settings in Figure 13. 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

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

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 $\mu_s$, $\mu_l$.** We explore the various combinations of server and local momentum constant $\mu_s$ and $\mu_l$ of DOMO and report the test accuracy in Figure 14. $\mu_s = 0.9$ 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 15 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 $\alpha$ and $\beta$.** We explore the combinations of $\alpha$ and $\beta$ and report the corresponding test accuracy in Table 4.4. $\alpha = 1.0$ and $\beta = 0.9$ work best.

**Varying Model.** We also plot the training curves of ResNet-56 in the right plot of Figure 15 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 4.4 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 4.4 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.

## 4.6 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.

# Appendix A "Improve the Efficiency of Model Parallelism"

## A.1 Queue Size

We mentioned that queue size "$\{q_k\}$ is determined by $\{p_k\}$ and $\{m_k\}$ because the input should match the corresponding error gradient". More specifically, it can be formulated as follows:

$$\begin{cases} q_k = m_{k-1} - p_{k-1} - m_k > 0 \quad \forall k \in \{1, \ldots, K-1\}, \\ q_0 = 0, \\ m_k > 0 \quad \forall k \in \{0, \ldots, K-1\}, \\ p_k > 0 \quad \forall k \in \{0, \ldots, K-2\}, \quad p_{K-1} = 0. \end{cases} \tag{A–1}$$

$q_0 = 0$ and $p_{K-1} = 0$ because usually there is no need for the corresponding queue in the first and last block. The first equation ensures that the input and backward error gradient in one block will come from the same data batch.

## A.2 Assumptions

**Assumption A.2.1.** *(**Bounded variance**) Assume that the variance of the DSP stochastic gradient $\mathcal{G}(x;\xi)$ is bounded, i.e.,*

$$Var[\mathcal{G}(x;\xi)] \leq \sigma^2. \tag{A–2}$$

**Assumption A.2.2.** *(**Lipschitz continuous gradient**) Assume that the loss and the output of the blocks have Lipschitz continuous gradient, that is, $\forall k \in \{0, 1, .., K-1\}$, and $\forall (x_{0,1}, ..., x_{k,1}), (x_{0,2}, ..., x_{k,2}) \in \mathbb{R}^{d_0+d_1+...+d_k}$,*

$$\|\nabla F(h_0; x_{0,1}; ...; x_{k,1}) - \nabla F(h_0; x_{0,2}; ...; x_{k,2})\| \leq L_k \|(x_{0,1}, ..., x_{k,1}) - (x_{0,2}, ..., x_{k,2})\|, \tag{A–3}$$

*and $\forall x_1,\ x_2 \in \mathbb{R}^d$,*

$$\|\nabla f(x_1) - \nabla f(x_2)\| \leq L_K \|x_1 - x_2\|. \tag{A–4}$$

**Assumption A.2.3.** *(**Bounded error gradient**) Assume that the norm of the error gradient that a block receives is bounded, that is, for any $x \in \mathbb{R}^d$, $\forall k \in \{0, 1, ..., K-2\}$,*

$$\left\| \frac{\partial f_{k+1}(h_{k+1}; x_{k+1})}{\partial h_{k+1}} ... \frac{\partial f_{K-1}(h_{K-1}; x_{K-1})}{\partial h_{K-1}} \frac{\partial \mathcal{L}(h_K, l)}{\partial h_K} \right\| \leq M, \quad \left\| \frac{\partial \mathcal{L}(h_K, l)}{\partial h_K} \right\| \leq M. \tag{A–5}$$

## A.3 Basic Lemmas

**Lemma A.3.1.** *If Assumptions A.2.2 and A.2.3 hold, the difference between DSP gradient and BP gradient regarding the parameters of block k satisfies:*

$$\left\| \nabla_{x_k} \mathcal{L}(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), y) - \mathcal{G}_{x_k}(x_0^{t_{2K-1}}; ...; x_{K-1}^{t_K}) \right\| \leq LM \sum_{i=k}^{K-1} \left\| x_i^{t_{2K-1-i}} - x_i^{t_i} \right\|. \tag{A–6}$$

*Proof.* We gradually move the DSP gradient of the block $k$ towards the BP gradient by replacing one block's backward parameters with its forward parameters at a time. $K-k$ steps in total are needed, and each step will introduce an error. After all the replacement is done, it becomes the BP gradient at the forward parameters. Firstly we replace $x_k^{t_{2K-1-k}}$ with $x_k^{t_k}$, and calculate the error introduced as follows,

$$\begin{aligned}
\|\Delta_k\| = & \left\| \left( \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; x_k^{t_{2K-1-k}})}{\partial x_k^{t_{2K-1-k}}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; x_k^{t_k})}{\partial x_k^{t_k}} \right) \cdot \right. \\
& \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{2K-2-k}})}{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k})} \cdots \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}})} \cdot \\
& \left. \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & \left\| \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; x_k^{t_{2K-1-k}})}{\partial x_k^{t_{2K-1-k}}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k-1}^{t_{k-1}}; x_k^{t_k})}{\partial x_k^{t_k}} \right\| \cdot \\
& \left\| \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{2K-2-k}})}{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k})} \cdots \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}})} \cdot \right. \\
& \left. \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & LM \left\| x_k^{t_{2K-1-k}} - x_k^{t_k} \right\|.
\end{aligned} \tag{A–7}$$

Secondly we replace $x_{k+1}^{t_{2K-2-k}}$ with $x_{k+1}^{t_{k+1}}$, and calculate the error introduced,

$$\begin{aligned}
\|\Delta_{k+1}\| = & \left\| \left( \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{2K-2-k}})}{\partial x_k^{t_k}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{k+1}})}{\partial x_k^{t_k}} \right) \cdot \right. \\
& \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k+1}^{t_{k+1}}; x_{k+2}^{t_{2K-3-k}})}{\partial F(h_0; x_0^{t_0}; ...; x_{k+1}^{t_{k+1}})} \cdots \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}})} \cdot \\
& \left. \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & \left\| \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{2K-2-k}})}{\partial x_k^{t_k}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_k^{t_k}; x_{k+1}^{t_{k+1}})}{\partial x_k^{t_k}} \right\| \cdot \\
& \left\| \frac{\partial F(h_0; x_0^{t_0}; ...; x_{k+1}^{t_{k+1}}; x_{k+2}^{t_{2K-3-k}})}{\partial F(h_0; x_0^{t_0}; ...; x_{k+1}^{t_{k+1}})} \cdots \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}})} \cdot \right. \\
& \left. \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & LM \left\| x_{k+1}^{t_{2K-2-k}} - x_{k+1}^{t_{k+1}} \right\|.
\end{aligned} \tag{A–8}$$

We repeatedly perform the above procedure, until we get the error in the last step,

$$\begin{aligned}
\|\Delta_{K-1}\| = & \left\| \left( \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial x_k^{t_k}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_{K-1}})}{\partial x_k^{t_k}} \right) \cdot \right. \\
& \left. \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & \left\| \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_K})}{\partial x_k^{t_k}} - \frac{\partial F(h_0; x_0^{t_0}; ...; x_{K-2}^{t_{K-2}}; x_{K-1}^{t_{K-1}})}{\partial x_k^{t_k}} \right\| \cdot \\
& \left\| \frac{\partial \mathcal{L}\left(F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}}), l\right)}{\partial F(h_0; x_0^{t_0}; ...; x_{K-1}^{t_{K-1}})} \right\| \\
\leq & LM \left\| x_{K-1}^{t_K} - x_{K-1}^{t_{K-1}} \right\|.
\end{aligned} \tag{A–9}$$

Add them together and we will have

$$\begin{aligned}
& \left\| \nabla_{x_k} \mathcal{L}(F(h_0; x_0^{t_0}; x_1^{t_1}; ...; x_{K-1}^{t_{K-1}}), l) - \mathcal{G}_{x_k}(x_0^{t_{2K-1}}; x_1^{t_{2K-2}}; ...; x_{K-1}^{t_K}) \right\| \\
= & \|\Delta_k + \Delta_{k+1} + ... + \Delta_{K-1}\| \leq \|\Delta_k\| + \|\Delta_{k+1}\| + ... + \|\Delta_{K-1}\| \\
\leq & LM \sum_{i=k}^{K-1} \left\| x_i^{t_{2K-1-i}} - x_i^{t_i} \right\|.
\end{aligned} \tag{A–10}$$

□

**Lemma A.3.2.** *Assume Assumption A.2.2 and A.2.3 exist. The second moment of the difference between DSP and BP gradient satisfies,*

$$\left\|\nabla f(x_0^{t_0};...;x_{K-1}^{t_{K-1}}) - \mathcal{G}(x_0^{t_{2K-1}};...;x_{K-1}^{t_K})\right\|^2 \leq \frac{1}{2}L^2 c_0 \sum_{k=0}^{K-1} \frac{k+1}{K+1} \left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|^2. \quad \text{(A–11)}$$

*Proof.* Via summation of Lemma A.3.1 we can get,

$$\left\|\nabla f(x_0^{t_0};x_1^{t_1};...;x_{K-1}^{t_{K-1}}) - \mathcal{G}(x_0^{t_{2K-1}};x_1^{t_{2K-2}};...;x_{K-1}^{t_K})\right\| \leq LM \sum_{k=0}^{K-1}(k+1)\left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|. \quad \text{(A–12)}$$

$$\begin{aligned}
&\left\|\nabla f(x_0^{t_0};x_1^{t_1};...;x_{K-1}^{t_{K-1}}) - \mathcal{G}(x_0^{t_{2K-1}};x_1^{t_{2K-2}};...;x_{K-1}^{t_K})\right\|^2 \\
&\leq L^2M^2\left(\sum_{k=0}^{K-1}(k+1)\left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|\right)^2 \\
&= L^2M^2\left(\sum_{k=0}^{K-1}(k+1)\right)^2\left(\sum_{k=0}^{K-1}\frac{k+1}{\sum_{k=0}^{K-1}(k+1)}\left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|\right)^2 \quad \text{(A–13)} \\
&\leq L^2M^2\left(\sum_{k=0}^{K-1}(k+1)\right)^2\sum_{k=0}^{K-1}\frac{k+1}{\sum_{k=0}^{K-1}(k+1)}\left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|^2 \\
&= \frac{1}{2}L^2M^2K(K+1)\sum_{k=0}^{K-1}(k+1)\left\|x_k^{t_{2K-1-k}} - x_k^{t_k}\right\|^2.
\end{aligned}$$

□

## A.4 DSP with SGD

**Theorem A.4.1.** *Assume Assumptions A.2.1, A.2.2 and A.2.3 hold. Let* $c_0 = M^2K(K+1)^2$*, and* $c_1 = -(\Delta t^2+2) + \sqrt{(\Delta t^2+2)^2 + 2c_0\Delta t^2}$*. If the learning rate* $\alpha_n \leq \frac{c_1}{Lc_0\Delta t^2}$*, then*

$$\frac{\sum_{n=0}^{N-1}\alpha_n\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2}{\sum_{n=0}^{N-1}\alpha_n} \leq \frac{2\left[f(x^0)-f^*\right]}{\sum_{n=0}^{N-1}\alpha_n} + \frac{L\sigma^2(2+K\Delta t^2+\frac{1}{4}Kc_1)\sum_{n=0}^{N-1}\alpha_n^2}{\sum_{n=0}^{N-1}\alpha_n}. \quad \text{(A–14)}$$

*Proof.* According to Lipschitz continuous, we have

$$
\begin{aligned}
&f(x^{n+1}) - f(x^n) \leq \left\langle \nabla f(x^n), x^{n+1} - x^n \right\rangle + \frac{L}{2} \left\| x^{n+1} - x^n \right\|^2 \\
&= -\alpha_n \left\langle \nabla f(x^n), \mathcal{G}(x^n;\xi) \right\rangle + \frac{L\alpha_n^2}{2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&= -\alpha_n \left\langle \nabla f(x^n) - \nabla f(x^{n'}), \mathcal{G}(x^n;\xi) \right\rangle - \alpha_n \left\langle \nabla f(x^{n'}), \mathcal{G}(x^n;\xi) \right\rangle + \frac{L\alpha_n^2}{2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&\leq \frac{1}{2L} \left\| \nabla f(x^n) - \nabla f(x^{n'}) \right\|^2 + \frac{L\alpha_n^2}{2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 - \alpha_n \left\langle \nabla f(x^{n'}), \mathcal{G}(x^n;\xi) \right\rangle \\
&\quad + \frac{L\alpha_n^2}{2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&\leq \frac{L}{2} \left\| x^n - x^{n'} \right\|^2 - \alpha_n \left\langle \nabla f(x^{n'}), \mathcal{G}(x^n;\xi) \right\rangle + L\alpha_n^2 \left\| \mathcal{G}(x^n;\xi) \right\|^2 .
\end{aligned}
\tag{A–15}
$$

Take expectation regarding $\xi$ on both sides,

$$
\begin{aligned}
&\mathbb{E}\left[ f(x^{n+1}) \right] - f(x^n) \leq \frac{L}{2} \left\| x^n - x^{n'} \right\|^2 - \alpha_n \left\langle \nabla f(x^{n'}), \mathcal{G}(x^n) \right\rangle + L\alpha_n^2 \mathbb{E} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&= \frac{L}{2} \left\| x^n - x^{n'} \right\|^2 + \frac{\alpha_n}{2} \left( \left\| \nabla f(x^{n'}) - \mathcal{G}(x^n) \right\|^2 - \left\| \nabla f(x^{n'}) \right\|^2 - \left\| \mathcal{G}(x^n) \right\|^2 \right) \\
&\quad + L\alpha_n^2 \left( \left\| \mathcal{G}(x^n) \right\|^2 + \mathrm{Var}\left[ \mathcal{G}(x^n;\xi) \right] \right) \\
&\leq \frac{L}{2} \left\| x^n - x^{n'} \right\|^2 + \frac{\alpha_n}{2} \left\| \nabla f(x^{n'}) - \mathcal{G}(x^n) \right\|^2 - \left( \frac{\alpha_n}{2} - L\alpha_n^2 \right) \left\| \mathcal{G}(x^n) \right\|^2 \\
&\quad - \frac{\alpha_n}{2} \left\| \nabla f(x^{n'}) \right\|^2 + L\alpha_n^2 \sigma^2 \\
&\leq \sum_{k=0}^{K-1} \left[ \frac{L}{2} + \frac{1}{4} \alpha_n L^2 M^2 K(K+1)(k+1) \right] \left\| x_k^n - x_k^{n'} \right\|^2 - \left( \frac{\alpha_n}{2} - L\alpha_n^2 \right) \left\| \mathcal{G}(x^n) \right\|^2 \\
&\quad - \frac{\alpha_n}{2} \left\| \nabla f(x^{n'}) \right\|^2 + L\alpha_n^2 \sigma^2 .
\end{aligned}
\tag{A–16}
$$

The last inequality utilizes Lemma A.3.2. Consider the first term and take expectation,

$$
\begin{aligned}
\mathbb{E} \left\| x_k^n - x_k^{n'} \right\|^2 &= \mathbb{E} \left\| \sum_{i=n-\Delta t_k}^{n-1} -\alpha_i \mathcal{G}_{x_k}(x^i;\xi) \right\|^2 \\
&\leq \Delta t_k \sum_{i=n-\Delta t_k}^{n-1} \alpha_i^2 \mathbb{E} \left\| \mathcal{G}_{x^k}(x^i;\xi) \right\|^2 \\
&\leq \Delta t \sum_{i=n-\Delta t}^{n-1} \alpha_i^2 \left( \left\| \mathcal{G}_{x_k}(x^i) \right\|^2 + \sigma^2 \right) .
\end{aligned}
\tag{A–17}
$$

Take the total expectation and perform summation for it,

$$\begin{aligned}
&\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\mathbb{E}\left\|x_k^n-x_k^{n'}\right\|^2\\
&\leq\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\Delta t\sum_{i=n-\Delta t}^{n-1}\alpha_i^2\left(\mathbb{E}\left\|\mathcal{G}_{x_k}(x^i)\right\|^2+\sigma^2\right) \qquad \text{(A–18)}\\
&\leq\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\Delta t\cdot\Delta t\cdot\alpha_n^2\left(\mathbb{E}\left\|\mathcal{G}_{x_k}(x^n)\right\|^2+\sigma^2\right).
\end{aligned}$$

Take the total expectation and perform summation for all the terms,

$$\begin{aligned}
&\mathbb{E}\left[f(x^N)\right]-f(x^0)\\
&\leq\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\Delta t^2\alpha_n^2\left(\mathbb{E}\left\|\mathcal{G}_{x_k}(x^n)\right\|^2+\sigma^2\right)\\
&\quad-\sum_{n=0}^{N-1}\left(\frac{\alpha_n}{2}-L\alpha_n^2\right)\mathbb{E}\sum_{k=0}^{K-1}\left\|\mathcal{G}_{x_k}(x^n)\right\|^2-\sum_{n=0}^{N-1}\frac{\alpha_n}{2}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2+L\sigma^2\sum_{n=0}^{N-1}\alpha_n^2\\
&=\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\Delta t^2\alpha_n^2-\frac{\alpha_n}{2}+L\alpha_n^2\right)\mathbb{E}\left\|\mathcal{G}_{x_k}(x^n)\right\|^2\\
&\quad+\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\left(\frac{L}{2}+\frac{1}{4}\alpha_n L^2M^2K(K+1)(k+1)\right)\Delta t^2\alpha_n^2\sigma^2-\sum_{n=0}^{N-1}\frac{\alpha_n}{2}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2\\
&\quad+L\sigma^2\sum_{n=0}^{N-1}\alpha_n^2
\end{aligned} \qquad \text{(A–19)}$$

$$\begin{aligned}
&\mathbb{E}\left[f(x^N)\right]-f(x^0)\\
&\leq\sum_{n=0}^{N-1}\sum_{k=0}^{K-1}\frac{1}{4}\alpha_n\left(L^2M^2K(K+1)^2\Delta t^2\alpha_n^2+\left(2\Delta t^2+4\right)L\alpha_n-2\right)\mathbb{E}\left\|\mathcal{G}_{x^k}(x^n)\right\|^2\\
&\quad+\sum_{n=0}^{N-1}\left(\frac{1}{2}LK+\frac{1}{8}\alpha_n L^2M^2K^2(K+1)^2\right)\Delta t^2\alpha_n^2\sigma^2-\sum_{n=0}^{N-1}\frac{\alpha_n}{2}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2+L\sigma^2\sum_{n=0}^{N-1}\alpha_n^2\\
&\leq\sum_{n=0}^{N-1}\left(\frac{1}{2}LK+\frac{1}{8}\alpha_n L^2M^2K^2(K+1)^2\right)\Delta t^2\alpha_n^2\sigma^2-\sum_{n=0}^{N-1}\frac{\alpha_n}{2}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2+L\sigma^2\sum_{n=0}^{N-1}\alpha_n^2.
\end{aligned} \qquad \text{(A–20)}$$

The last inequality utilizes the restriction on the learning rate. Then we have

$$\begin{aligned}&\frac{\sum_{n=0}^{N-1}\alpha_n\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2}{\sum_{n=0}^{N-1}\alpha_n}\\&\leq\frac{2\left[f(x^0)-f^*\right]}{\sum_{n=0}^{N-1}\alpha_n}+\frac{L\sigma^2\sum_{n=0}^{N-1}\alpha_n^2\left[2+K\Delta t^2+\frac{1}{4}\alpha_n LM^2K^2(K+1)^2\Delta t^2\right]}{\sum_{n=0}^{N-1}\alpha_n}.\end{aligned}\tag{A–21}$$

□

## A.5 DSP with Momentum SGD

The SUM method also implies the following recursions,

$$\begin{aligned}x^{n+1}+\frac{\beta}{1-\beta}v^{n+1}&=x^n+\frac{\beta}{1-\beta}v^n-\frac{\alpha}{1-\beta}\mathcal{G}(x^n;\xi),\quad n\geq 0\\v^{n+1}&=\beta v^n+((1-\beta)s-1)\alpha\mathcal{G}(x^n;\xi),\quad n\geq 0.\end{aligned}\tag{A–22}$$

$$v^n=\begin{cases}x^n-x^{n-1}+s\alpha\mathcal{G}(x^{n-1};\xi),\quad n\geq 1\\0,\quad n=0.\end{cases}\tag{A–23}$$

Let $z^n=x^n+\frac{\beta}{1-\beta}v^n$.

**Lemma A.5.1.** *Assume Assumption 1 exists. Let* $c_2=\frac{((1-\beta)s-1)^2}{(1-\beta)^2}$*, then*

$$\sum_{n=0}^{N-1}\mathbb{E}\left\|v^n\right\|^2\leq c_2\alpha^2\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2+c_2\sigma^2\alpha^2N.\tag{A–24}$$

*Proof.* Let $\hat{\alpha}=((1-\beta)s-1)\alpha$. From Eq. ( A–22),

$$v^{n+1}=\beta v^n+\hat{\alpha}\mathcal{G}(x^n;\xi).\tag{A–25}$$

Note that $v^0=0$. Then

$$v^n=\hat{\alpha}\sum_{i=0}^{n-1}\beta^{n-1-i}\mathcal{G}(x^i;\xi).\tag{A–26}$$

Then we have,

$$\begin{aligned}
\mathbb{E}\left\|v^n\right\|^2 &= \hat{\alpha}^2\mathbb{E}\left\|\sum_{i=0}^{n-1}\beta^{n-1-i}\mathcal{G}(x^i;\xi)\right\|^2 = \hat{\alpha}^2\left(\sum_{i=0}^{n-1}\beta^{n-1-i}\right)^2\mathbb{E}\left\|\sum_{i=0}^{n-1}\frac{\beta^{n-1-i}}{\sum_{i=0}^{n-1}\beta^{n-1-i}}\mathcal{G}(x^i;\xi)\right\|^2 \\
&\leq \hat{\alpha}^2\left(\sum_{i=0}^{n-1}\beta^{n-1-i}\right)^2\sum_{i=0}^{n-1}\frac{\beta^{n-1-i}}{\sum_{i=0}^{n-1}\beta^{n-1-i}}\mathbb{E}\left\|\mathcal{G}(x^i;\xi)\right\|^2 \\
&= \hat{\alpha}^2\sum_{i=0}^{n-1}\beta^{n-1-i}\sum_{i=0}^{n-1}\beta^{n-1-i}\left\|\mathcal{G}(x^i)\right\|^2 + \hat{\alpha}^2\sigma^2\left(\sum_{i=0}^{n-1}\beta^{n-1-i}\right)^2 \\
&\leq \frac{\hat{\alpha}^2}{1-\beta}\sum_{i=0}^{n-1}\beta^{n-1-i}\left\|\mathcal{G}(x^i)\right\|^2 + \frac{\hat{\alpha}^2\sigma^2}{(1-\beta)^2} \\
&= (1-\beta)c_2\alpha^2\sum_{i=0}^{n-1}\beta^{n-1-i}\left\|\mathcal{G}(x^i)\right\|^2 + c_2\alpha^2\sigma^2.
\end{aligned} \tag{A–27}$$

Take the total expectation and perform summation,

$$\begin{aligned}
\sum_{n=0}^{N-1}\mathbb{E}\left[\left\|v^n\right\|^2\right] &\leq (1-\beta)c_2\alpha^2\sum_{n=0}^{N-1}\sum_{i=0}^{n-1}\beta^{n-1-i}\mathbb{E}\left\|\mathcal{G}(x^i)\right\|^2 + c_2\alpha^2\sigma^2N \\
&= (1-\beta)c_2\alpha^2\sum_{i=0}^{N-2}\sum_{n=i+1}^{N-1}\beta^{n-1-i}\mathbb{E}\left\|\mathcal{G}(x^i)\right\|^2 + c_2\alpha^2\sigma^2N \\
&= (1-\beta)c_2\alpha^2\sum_{i=0}^{N-2}\frac{1-\beta^{N-1-i}}{1-\beta}\mathbb{E}\left\|\mathcal{G}(x^i)\right\|^2 + c_2\alpha^2\sigma^2N \\
&\leq c_2\alpha^2\sum_{n=0}^{N-2}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 + c_2\sigma^2\alpha^2N \leq c_2\alpha^2\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 + c_2\sigma^2\alpha^2N.
\end{aligned} \tag{A–28}$$

□

**Lemma A.5.2.** *Assume Assumption A.2.1 exists, then*

$$\sum_{n=0}^{N-1}\mathbb{E}\left\|x^n - x^{n'}\right\|^2 \leq 2\Delta t^2(c_2+s^2)\alpha^2\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 + 2\Delta t^2\sigma^2(c_2+s^2)\alpha^2N. \tag{A–29}$$

*Proof.* First take expectation regarding $\xi$,

$$\begin{aligned}
\mathbb{E}\left\|x^n - x^{n'}\right\|^2 &= \sum_{k=0}^{K-1} \mathbb{E}\left\|x_k^n - x_k^{n'}\right\|^2 = \sum_{k=0}^{K-1} \mathbb{E}\left\|\sum_{i=n-\Delta t_k}^{n-1} v_k^{i+1} - s\alpha \mathcal{G}_{x_k}(x^i;\xi)\right\|^2 \\
&\leq \sum_{k=0}^{K-1} \Delta t_k \sum_{i=n-\Delta t_k}^{n-1} \mathbb{E}\left\|v_k^{i+1} - s\alpha \mathcal{G}_{x_k}(x^i;\xi)\right\|^2 \\
&\leq \sum_{k=0}^{K-1} 2\Delta t_k \sum_{i=n-\Delta t_k}^{n-1} \left(\mathbb{E}\left\|v_k^{i+1}\right\|^2 + s^2\alpha^2 \mathbb{E}\left\|\mathcal{G}_{x_k}(x^i;\xi)\right\|^2\right) \\
&\leq \sum_{k=0}^{K-1} 2\Delta t \sum_{i=n-\Delta t}^{n-1} \left(\mathbb{E}\left\|v_k^{i+1}\right\|^2 + s^2\alpha^2 \mathbb{E}\left\|\mathcal{G}_{x_k}(x^i;\xi)\right\|^2\right) \\
&= 2\Delta t \sum_{i=n-\Delta t}^{n-1} \left(\mathbb{E}\left\|v^{i+1}\right\|^2 + s^2\alpha^2 \mathbb{E}\left\|\mathcal{G}(x^i;\xi)\right\|^2\right) \\
&\leq 2\Delta t \sum_{i=n-\Delta t}^{n-1} \left(\mathbb{E}\left\|v^{i+1}\right\|^2 + s^2\alpha^2 \left\|\mathcal{G}(x^i)\right\|^2 + s^2\alpha^2\sigma^2\right).
\end{aligned} \tag{A–30}$$

Take total expectation on both sides and perform summation,

$$\begin{aligned}
\sum_{n=0}^{N-1} \mathbb{E}\left\|x^n - x^{n'}\right\|^2 &\leq 2\Delta t \sum_{n=0}^{N-1} \sum_{i=n-\Delta t}^{n-1} \left(\mathbb{E}\left\|v^{i+1}\right\|^2 + s^2\alpha^2 \mathbb{E}\left\|\mathcal{G}(x^i)\right\|^2 + s^2\alpha^2\sigma^2\right) \\
&\leq 2\Delta t^2 \sum_{n=0}^{N-2} \left(\mathbb{E}\left\|v^{n+1}\right\|^2 + s^2\alpha^2 \mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 + s^2\alpha^2\sigma^2\right) \\
&\leq 2\Delta t^2 \sum_{n=0}^{N-1} \mathbb{E}\left\|v^n\right\|^2 + 2\Delta t^2 s^2\alpha^2 \sum_{n=0}^{N-1} \mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 + 2\Delta t^2 s^2\alpha^2\sigma^2 N \\
&\leq 2\Delta t^2 (c_2 + s^2)\alpha^2 \sum_{n=0}^{N-1} \mathbb{E}\left[\left\|\mathcal{G}(x^n)\right\|^2\right] + 2\Delta t^2\sigma^2(c_2+s^2)\alpha^2 N.
\end{aligned} \tag{A–31}$$

□

**Theorem A.5.1.** *Assume Assumption A.2.1, A.2.2 and A.2.3 hold. Let* $c_2 = \frac{((1-\beta)s-1)^2}{(1-\beta)^2}$, $c_3 = M^2K(K+1)^2\Delta t^2(c_2+s^2)$, $c_4 = 3 + \beta^2 c_2 + 2(1-\beta)^2\Delta t^2(c_2+s^2)$, *and* $c_5 = \frac{2+\beta^2 c_2}{1-\beta} + 2(1-\beta)\Delta t^2(c_2+s^2) + \frac{-c_4+\sqrt{c_4^2+4(1-\beta)^2 c_3}}{2(1-\beta)}$. *If the learning rate* $\alpha$ *is fixed and satisfies* $\alpha \leq \frac{-c_4+\sqrt{c_4^2+4(1-\beta)^2 c_3}}{2(1-\beta)c_3 L}$, *then*

$$\frac{1}{N}\sum_{n=0}^{N-1} \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 \leq \frac{2(1-\beta)(f(x^0)-f^*)}{N\alpha} + c_5\sigma^2 L\alpha. \tag{A–32}$$

*Proof.* According to Lipschitz continuous gradient,

$$
\begin{aligned}
&f(z^{n+1}) - f(z^n) \\
&\le \left\langle \nabla f(z^n), z^{n+1} - z^n \right\rangle + \frac{L}{2} \left\| z^{n+1} - z^n \right\|^2 \\
&= -\frac{\alpha}{1-\beta} \left\langle \nabla f(z^n), \mathcal{G}(x^n;\xi) \right\rangle + \frac{L\alpha^2}{2(1-\beta)^2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&= -\frac{\alpha}{1-\beta} \left\langle \nabla f(z^n) - \nabla f(x^n), \mathcal{G}(x^n;\xi) \right\rangle - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^n), \mathcal{G}(x^n;\xi) \right\rangle \\
&\quad + \frac{L\alpha^2}{2(1-\beta)^2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&\le \frac{1}{2} \left( \frac{1}{L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \right) \\
&\quad - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^n), \mathcal{G}(x^n;\xi) \right\rangle + \frac{L\alpha^2}{2(1-\beta)^2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 \\
&= \frac{1}{2L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^n), \mathcal{G}(x^n;\xi) \right\rangle + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n;\xi) \right\|^2 .
\end{aligned}
\tag{A–33}
$$

Take expectation regarding $\xi$ on both sides,

$$
\begin{aligned}
&\mathbb{E}\left[ f(z^{n+1}) \right] - f(z^n) \\
&\le \frac{1}{2L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^n), \mathcal{G}(x^n) \right\rangle + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \sigma^2 \\
&= \frac{1}{2L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^n) - \nabla f(x^{n'}), \mathcal{G}(x^n) \right\rangle \\
&\quad - \frac{\alpha}{1-\beta} \left\langle \nabla f(x^{n'}), \mathcal{G}(x^n) \right\rangle + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \sigma^2 \\
&\le \frac{1}{2L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 + \frac{1}{2} \left( \frac{1}{L} \left\| \nabla f(x^n) - \nabla f(x^{n'}) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n) \right\|^2 \right) \\
&\quad + \frac{\alpha}{2(1-\beta)} \left( \left\| \nabla f(x^{n'}) - \mathcal{G}(x^n) \right\|^2 - \left\| \nabla f(x^{n'}) \right\|^2 - \left\| \mathcal{G}(x^n) \right\|^2 \right) \\
&\quad + \frac{L\alpha^2}{(1-\beta)^2} \left\| \mathcal{G}(x^n) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \sigma^2 \\
&= -\frac{\alpha}{2(1-\beta)} \left\| \nabla f(x^{n'}) \right\|^2 + \frac{1}{2L} \left\| \nabla f(z^n) - \nabla f(x^n) \right\|^2 + \frac{1}{2L} \left\| \nabla f(x^n) - \nabla f(x^{n'}) \right\|^2 \\
&\quad + \frac{\alpha}{2(1-\beta)} \left\| \nabla f(x^{n'}) - \mathcal{G}(x^n) \right\|^2 - \left( \frac{\alpha}{2(1-\beta)} - \frac{3L\alpha^2}{2(1-\beta)^2} \right) \left\| \mathcal{G}(x^n) \right\|^2 + \frac{L\alpha^2}{(1-\beta)^2} \sigma^2 .
\end{aligned}
\tag{A–34}
$$

Take the total expectation and perform summation,

$$\sum_{n=0}^{N-1}\mathbb{E}\left[\frac{1}{2L}\left\|\nabla f(z^n)-\nabla f(x^n)\right\|^2\right]\leq\sum_{n=0}^{N-1}\frac{L}{2}\mathbb{E}\left\|z^n-x^n\right\|^2=\sum_{n=0}^{N-1}\frac{L\beta^2}{2(1-\beta)^2}\mathbb{E}\left\|v^n\right\|^2. \quad \text{(A–35)}$$

$$\begin{aligned}
&\sum_{n=0}^{N-1}\mathbb{E}\left[\frac{1}{2L}\left\|\nabla f(x^n)-\nabla f(x^{n'})\right\|^2+\frac{\alpha}{2(1-\beta)}\left\|\nabla f(x^{n'})-\mathcal{G}(x^n)\right\|^2\right]\\
&\leq\sum_{n=0}^{N-1}\frac{L}{2}\mathbb{E}\left\|x^n-x^{n'}\right\|^2+\frac{\alpha}{4(1-\beta)}L^2M^2K(K+1)\sum_{k=0}^{K-1}(k+1)\sum_{n=0}^{N-1}\mathbb{E}\left\|x_k^n-x_k^{n'}\right\|^2\\
&\leq\sum_{n=0}^{N-1}\frac{L}{2}\mathbb{E}\left\|x^n-x^{n'}\right\|^2+\frac{\alpha}{4(1-\beta)}L^2M^2K(K+1)^2\sum_{n=0}^{N-1}\mathbb{E}\left\|x^n-x^{n'}\right\|^2\\
&\leq\sum_{n=0}^{N-1}\frac{L}{2}\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\mathbb{E}\left\|x^n-x^{n'}\right\|^2.
\end{aligned} \quad \text{(A–36)}$$

Then we have,

$$\begin{aligned}
&\mathbb{E}\left[f(z^N)\right]-f(z^0)\\
&\leq-\frac{\alpha}{2(1-\beta)}\sum_{n=0}^{N-1}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2-\left(\frac{\alpha}{2(1-\beta)}-\frac{3L\alpha^2}{2(1-\beta)^2}\right)\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2+\frac{L\sigma^2\alpha^2}{(1-\beta)^2}N\\
&\quad+\sum_{n=0}^{N-1}\frac{L\beta^2}{2(1-\beta)^2}\mathbb{E}\left\|v^n\right\|^2+\sum_{n=0}^{N-1}\frac{L}{2}\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\mathbb{E}\left\|x^n-x^{n'}\right\|^2\\
&\leq-\frac{\alpha}{2(1-\beta)}\sum_{n=0}^{N-1}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2-\left(\frac{\alpha}{2(1-\beta)}-\frac{3L\alpha^2}{2(1-\beta)^2}\right)\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2+\frac{L\sigma^2\alpha^2}{(1-\beta)^2}N\\
&\quad+\frac{L\beta^2}{2(1-\beta)^2}\left(c_2\alpha^2\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2+c_2\sigma^2\alpha^2N\right)+\frac{L}{2}\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\cdot\\
&\qquad\left[2\Delta t^2(c_2+s^2)\alpha^2\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2+2\Delta t^2\sigma^2(c_2+s^2)\alpha^2N\right]\\
&=-\frac{\alpha}{2(1-\beta)}\sum_{n=0}^{N-1}\mathbb{E}\left\|\nabla f(x^{n'})\right\|^2-\left[\frac{\alpha}{2(1-\beta)}-\alpha^2\left(\frac{3L}{2(1-\beta)^2}+\frac{L\beta^2c_2}{2(1-\beta)^2}+\right.\right.\\
&\qquad\left.\left.L\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\Delta t^2(c_2+s^2)\right)\right]\cdot\sum_{n=0}^{N-1}\mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2\\
&\quad+\sigma^2\alpha^2N\left[\frac{L}{(1-\beta)^2}+\frac{L\beta^2c_2}{2(1-\beta)^2}+L\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\Delta t^2(c_2+s^2)\right]
\end{aligned} \quad \text{(A–37)}$$

$$
\begin{aligned}
&\mathbb{E}\left[f(z^N)\right] - f(z^0) \\
&\leq -\frac{\alpha}{2(1-\beta)} \sum_{n=0}^{N-1} \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 + \frac{\alpha}{2(1-\beta)^2}[(1-\beta)M^2K(K+1)^2\Delta t^2(c_2+s^2)L^2\alpha^2 + \\
&\quad \left(3+\beta^2c_2+2(1-\beta)^2\Delta t^2(c_2+s^2)\right)L\alpha - (1-\beta)] \cdot \sum_{n=0}^{N-1} \mathbb{E}\left\|\mathcal{G}(x^n)\right\|^2 \\
&\quad + \sigma^2\alpha^2 N\left[\frac{L}{(1-\beta)^2} + \frac{L\beta^2c_2}{2(1-\beta)^2} + L\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\Delta t^2(c_2+s^2)\right].
\end{aligned}
\tag{A–38}
$$

The second inequality utilizes Lemma A.5.1 and A.5.2. According to the restriction on the learning rate, we can remove the second term in the last equality,

$$
\begin{aligned}
f_* - f(x^0) \leq -\frac{\alpha}{2(1-\beta)} \sum_{n=0}^{N-1} \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 + \sigma^2 L\alpha^2 N\left[\frac{1}{(1-\beta)^2} + \frac{\beta^2c_2}{2(1-\beta)^2} + \right. \\
\left.\left(1+\frac{\alpha}{2(1-\beta)}LM^2K(K+1)^2\right)\Delta t^2(c_2+s^2)\right].
\end{aligned}
\tag{A–39}
$$

Therefore we have,

$$
\begin{aligned}
\frac{1}{N}\sum_{n=0}^{N-1} \mathbb{E}\left\|\nabla f(x^{n'})\right\|^2 \leq \frac{2(1-\beta)(f^* - f(x^0))}{N\alpha} \\
+ \sigma^2 L\alpha\left[\frac{2+\beta^2c_2}{1-\beta} + \left(2(1-\beta)+\alpha LM^2K(K+1)^2\right)\Delta t^2(c_2+s^2)\right].
\end{aligned}
\tag{A–40}
$$

□

# Appendix B "Improve the Efficiency of Data Parallelism"

## B.1 Proof of Convergence of DEF (Theorem 2.4.1)

In this section, we consider general non-convex objective functions with $\delta$-approximate and ring-allreduce compatible compressor. We want $\frac{1}{K}\sum_{k=1}^{K}\|\frac{1}{K}\sum_{k'=1}^{K}e_{k',t}-\lambda_t e_{k,t}\|_2^2$ to be as close to zero as possible.

For ease of notation, let $e_t \coloneqq \frac{1}{K}\sum_{k=1}^{K}e_{k,t}$, $\mathrm{Var}(e_t) \coloneqq \frac{1}{K}\sum_{k=1}^{K}\|e_{k,t}-e_t\|_2^2$. We have $\frac{1}{K}\sum_{k=1}^{K}\|e_{k,t}\|_2^2 = \|e_t\|_2^2 + \mathrm{Var}(e_t)$. Then

$$\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\|\frac{1}{K}\sum_{k'=1}^{K}e_{k,t}-\lambda_t e_{k,t}\|_2^2 = \frac{1}{K}\sum_{k=1}^{K}\|e_t-\lambda_t e_{k,t}\|^2 \\
&= \frac{1}{K}\sum_{k=1}^{K}(\|e_t\|_2^2 - 2\lambda_t\langle e_t, e_{k,t}\rangle + \lambda_t^2\|e_{k,t}\|_2^2) = \frac{1}{K}\sum_{k=1}^{K}\|e_{k,t}\|_2^2\lambda_t^2 - 2\|e_t\|_2^2\lambda_t + \|e_t\|_2^2 \qquad \text{(B–1)} \\
&= (\|e_t\|_2^2 + \mathrm{Var}(e_t))\lambda_t^2 - 2\|e_t\|_2^2\lambda_t + \|e_t\|_2^2\,.
\end{aligned}$$

When $\lambda_t^* = \frac{\|e_t\|_2^2}{\|e_t\|_2^2+\mathrm{Var}(e_t)}$, it has the minimum $\frac{\|e_t\|_2^2\cdot\mathrm{Var}(e_t)}{\|e_t\|_2^2+\mathrm{Var}(e_t)}$.

If we allow individual coefficient $\lambda_{k,t}$ for each worker $k \in [K]$, then $\lambda_{k,t}^* = \frac{\langle e_t, e_{k,t}\rangle}{\|e_{k,t}\|_2^2}$ by minimizing $\|e_t-\lambda_{k,t}e_{k,t}\|_2^2$. However, it is impractical due to the additional $K$ hyper-parameters to tune when $K$ is large.

For simplicity, let $\lambda_t \to \lambda$ because it is hard to manually tune $\lambda_t$ during the training.

### B.1.1 Lemmas

For ease of notation, let $g_{k,t}$ denotes the stochastic gradient computed at iteration $t$ for worker $k$ and $g_t \coloneqq \frac{1}{K}\sum_{k=1}^{K}g_{k,t}$.

**Lemma B.1.1.** *Let Assumptions 2.3.1, 2.3.2, and 2.4.1 hold. Let* $B_1 = (\frac{1}{K}-\lambda)^2 + \frac{K-1}{K^2}$ *and* $B_2 = |\frac{1}{K}-\lambda| + \frac{K-1}{K}$. *We have*

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|g_t-\lambda g_{k,t}\|_2^2 \le B_1\sigma^2 + 2(1-\lambda)^2M^2 + 2B_2^2L^2\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t-\lambda e_{k,t}\|_2^2\,, \qquad \text{(B–2)}$$

*Proof.* We know that $g_{k,t} = \nabla f(x_t - \lambda e_{k,t}; \xi_{k,t})$. Then

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|g_t - \lambda g_{k,t}\|_2^2 \\
&= \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}[\nabla f(x_t - \lambda e_{k',t}; \xi_{k',t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t})] \\
&\qquad\qquad - \lambda[\nabla f(x_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t})] \\
&\qquad\qquad + \frac{1}{K}\sum_{k'=1}^{K}[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t}) - \nabla F_{\mathcal{S}}(y_t)] \\
&\qquad\qquad - \lambda[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)] + (1-\lambda)\nabla F_{\mathcal{S}}(y_t)\|_2^2 \\
&\overset{(a)}{=} \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}[\nabla f(x_t - \lambda e_{k',t}; \xi_{k',t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t})] \\
&\qquad\qquad - \lambda[\nabla f(x_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t})]\|_2^2 \\
&\quad + \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t}) - \nabla F_{\mathcal{S}}(y_t)] - \lambda[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)] \\
&\qquad\qquad + (1-\lambda)\nabla F_{\mathcal{S}}(y_t)\|_2^2 \\
&\overset{(b)}{\leq} ① + \underbrace{\frac{2}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t}) - \nabla F_{\mathcal{S}}(y_t)] - \lambda[\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)]\|_2^2}_{②} \\
&\quad + 2(1-\lambda)^2 M^2\,,
\end{aligned}
\tag{B–3}
$$

where we define

$$
\begin{aligned}
① = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}&[\nabla f(x_t - \lambda e_{k',t}; \xi_{k',t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k',t})] \\
&- \lambda[\nabla f(x_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t})]\|_2^2,
\end{aligned}
\tag{B–4}
$$

and (a) is due to $\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) = \mathbb{E}g_{k,t} = \mathbb{E}\nabla f(x_t - \lambda e_{k,t}; \xi_{k,t})$, (b) follows Assumption 2.3.2. Now we consider term ①. For simplicity, let $a_k = \nabla f(x_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t})$

and we will have $\mathbb{E}\langle a_k, a_{k'}\rangle = 0$ when $k \neq k'$. Then

$$① = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|(\frac{1}{K}-\lambda)a_k + \frac{1}{K}\sum_{k'=1,k'\neq k}^{K} a_{k'}\|_2^2 = \frac{1}{K}\sum_{k=1}^{K}\underbrace{[(\frac{1}{K}-\lambda)^2 + \frac{K-1}{K^2}]}_{B_1}\|a_k\|_2^2 \overset{(a)}{\leq} B_1\sigma^2\,, \tag{B–5}$$

where (a) follows Assumption 2.3.1. $B_1 = 0$ if and only if $K = 1$ and $\lambda = 1$. Now we consider term ②. For simplicity, let $b_k = \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)$ and $B_2 = |\frac{1}{K} - \lambda| + \frac{K-1}{K}$. $B_2 = 0$ if and only if $K = 1$ and $\lambda = 1$. Then

$$\begin{aligned}
② &= \frac{2}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K} b_{k'} - \lambda b_k\|_2^2 = \frac{2}{K}\sum_{k=1}^{K}\mathbb{E}\|(\frac{1}{K}-\lambda)b_k + \frac{1}{K}\sum_{k'=1,k'\neq k}^{K} b_{k'}\|_2^2 \\
&= \frac{2}{K}\sum_{k=1}^{K} B_2^2\mathbb{E}\|\frac{\frac{1}{K}-\lambda}{B_2}b_k + \frac{1}{KB_2}\sum_{k'=1,k'\neq k}^{K} b_{k'}\|_2^2 \\
&\leq \frac{2}{K}\sum_{k=1}^{K} B_2^2\mathbb{E}(\frac{|\frac{1}{K}-\lambda|}{B_2}\|b_k\|_2^2 + \frac{1}{KB_2}\sum_{k'=1,k'\neq k}^{K}\|b_{k'}\|_2^2) \\
&= \frac{2}{K}\sum_{k=1}^{K} B_2^2(\frac{|\frac{1}{K}-\lambda|}{B_2} + \frac{K-1}{KB_2})\mathbb{E}\|b_k\|_2^2 = \frac{2B_2^2}{K}\sum_{k=1}^{K}\mathbb{E}\|b_k\|_2^2 \\
&\overset{(a)}{\leq} 2B_2^2L^2\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2\,,
\end{aligned} \tag{B–6}$$

where (a) follows Assumption 2.4.1. Substitute terms ① and ② with their bounds and we can complete the proof.

□

**Lemma B.1.2.** *Let Assumptions 2.3.1, 2.3.2, 2.4.1, 2.4.2 and 2.3.3 hold. Let* $B_1 = (\frac{1}{K}-\lambda)^2 + \frac{K-1}{K^2}$ *and* $\eta < \frac{1}{2L}$*, we have*

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K} e_{k',t} - \lambda e_{k,t}\|_2^2 \leq \frac{1}{(\sqrt{\frac{1-\delta/2}{1-\delta}}-1)^2}\eta^2[B_1\sigma^2 + 2(1-\lambda)^2M^2]\,. \tag{B–7}$$

*Proof.* We have

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{K}\sum_{k'=1}^{K}e_{k',t}-\lambda e_{k,t}\|_2^2=\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t-\lambda e_{k,t}\|_2^2\\
&\overset{(a)}{=}\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|(\eta g_{t-1}+e_{t-1})-\mathcal{C}(\eta g_{t-1}+e_{t-1})-\lambda(\eta g_{k,t-1}+e_{k,t-1})+\lambda\mathcal{C}(\eta g_{k,t-1}+e_{k,t-1})\|_2^2\\
&\overset{(a)}{=}\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|(\eta g_{t-1}-\lambda\eta g_{k,t-1}+e_{t-1}-\lambda e_{k,t-1})-\mathcal{C}(\eta g_{t-1}-\lambda\eta g_{k,t-1}+e_{t-1}-\lambda e_{k,t-1})\|_2^2\\
&\overset{(b)}{=}(1-\delta)\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\eta g_{t-1}-\lambda\eta g_{k,t-1}+e_{t-1}-\lambda e_{k,t-1}\|_2^2\\
&\leq(1-\delta)(1+\beta)\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{t-1}-\lambda e_{k,t-1}\|_2^2+(1-\delta)(1+\frac{1}{\beta})\eta^2\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|g_{t-1}-\lambda g_{k,t-1}\|_2^2\\
&\overset{(c)}{=}(1-\delta)(1+\beta)\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{t-1}-\lambda e_{k,t-1}\|_2^2+(1-\delta)(1+\frac{1}{\beta})\eta^2[B_1\sigma^2+2(1-\lambda)^2M^2]\\
&\quad+(1-\delta)(1+\frac{1}{\beta})2B_2^2\eta^2L^2\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{t-1}-\lambda e_{k,t-1}\|_2^2\,,
\end{aligned}
\tag{B–8}
$$

where (a) follows Assumption 2.3.3, (b) follows Assumption 2.4.2, and (c) follows Lemma B.1.1. $\beta$ is a constant such that $0<\beta<\frac{\delta}{1-\delta}$, i.e., $(1-\delta)(1+\beta)<1$. Let $B_3=(1-\delta)(1+\frac{1}{\beta})2B_2^2\eta^2L^2<1-(1-\delta)(1+\beta)$, i.e., $B_3+(1-\delta)(1+\beta)<1$, then

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t-\lambda e_{k,t}\|_2^2\\
&\leq[B_3+(1-\delta)(1+\beta)]\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{t-1}-\lambda e_{k,t-1}\|_2^2+(1-\delta)(1+\frac{1}{\beta})\eta^2[B_1\sigma^2+2(1-\lambda)^2M^2]\\
&=(1-\delta)(1+\frac{1}{\beta})[B_1\sigma^2+2(1-\lambda)^2M^2]\sum_{t'=0}^{t-1}[B_3+(1-\delta)(1+\beta)]^{t-1-t'}\eta^2\\
&<\underbrace{\frac{(1-\delta)(1+\frac{1}{\beta})}{1-B_3-(1-\delta)(1+\beta)}}_{h(\beta)}\eta^2[B_1\sigma^2+2(1-\lambda)^2M^2]\,.
\end{aligned}
\tag{B–9}
$$

Now we consider the minimum value of $h(\beta)$. Its gradient regarding $\beta$ is

$$\frac{\partial h(\beta)}{\partial \beta} = \frac{1-\delta}{\beta^2[1-B_3-(1-\delta)(1+\beta)]^2}[(1-\delta)\beta^2 + 2(1-\delta)\beta + B_3 - \delta]\,. \tag{B–10}$$

Therefore,

$$\begin{aligned}
\beta^* = -1 + \sqrt{\frac{1-B_3}{1-\delta}} \to B_3 = 1-(1-\delta)(1+\beta^*)^2 < 1-(1-\delta)(1+\beta^*)\,,\\
h(\beta^*) = \frac{1-\delta}{(\sqrt{1-B_3}-\sqrt{1-\delta})^2} = \frac{1}{(\sqrt{\frac{1-B_3}{1-\delta}}-1)^2}\,,\\
\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 \le \frac{1}{(\sqrt{\frac{1-B_3}{1-\delta}}-1)^2}\eta^2[B_1\sigma^2 + 2(1-\lambda)^2M^2]\,,
\end{aligned} \tag{B–11}$$

which completes the proof. For simplicity we can just set $B_3 \le \delta/2$ (we can choose a constant $> 1$ other than 2), which is valid as it leads to $\beta^* \le -1 + \sqrt{\frac{1-\delta/2}{1-\delta}}$ and $-1 + \sqrt{\frac{1-\delta/2}{1-\delta}} < \frac{\delta}{1-\delta}$ holds. Based on the definition of $B_3$, it also requires that

$$2B_2^2\eta^2L^2 < \frac{\delta/2}{(1-\delta)(-1+\sqrt{\frac{1-\delta/2}{1-\delta}})} = \frac{\delta/2}{-(1-\delta)+\sqrt{(1-\delta)(1-\delta/2)}}\,, \tag{B–12}$$

where the R.H.S. is monotonically increasing for $0 < \delta < 1$. Therefore, for all conditions above to hold, we only need to assume that

$$2B_2^2\eta^2L^2 \le 4(1+(1-\lambda)^2)\eta^2L^2 < \lim_{\delta\to 0}\frac{\delta/2}{-(1-\delta)+\sqrt{(1-\delta)(1-\delta/2)}} = 2\,. \tag{B–13}$$

As $0 < \lambda < 1$, we can simply assume $\eta < \frac{1}{2L}$. □

**Lemma B.1.3.** *Let Assumptions 2.3.1, 2.3.2, 2.4.2, and 2.3.3 hold. We have*

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 \le \frac{\eta^2(\sigma^2+M^2)}{(\sqrt{1/(1-\delta)}-1)^2}. \tag{B–14}$$

*Proof.* We have

$$\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 \overset{(a)}{=} \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\eta g_{k,t-1}+e_{k,t-1}-\mathcal{C}(\eta g_{k,t-1}+e_{k,t-1})\|_2^2\\
&\overset{(b)}{\leq} \frac{1-\delta}{K}\sum_{k=1}^{K}\mathbb{E}\|\eta g_{k,t-1}+e_{k,t-1}\|_2^2\\
&\leq (1-\delta)(1+\beta)\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t-1}\|_2^2+(1-\delta)(1+\frac{1}{\beta})\eta^2(\sigma^2+M^2) \qquad \text{(B–15)}\\
&\leq (1-\delta)(1+\frac{1}{\beta})(\sigma^2+M^2)\sum_{t'=0}^{t-1}[(1-\delta)(1+\beta)]^{t-1-t'}\eta^2\\
&\leq \frac{(1-\delta)(1+\frac{1}{\beta})}{1-(1-\delta)(1+\beta)}\eta^2(\sigma^2+M^2)=\frac{1}{(\sqrt{1/(1-\delta)}-1)^2}\eta^2(\sigma^2+M^2)\,,
\end{aligned}$$

where $\beta=-1+\frac{1}{\sqrt{1-\delta}}$, (a) follows Assumption 2.3.3, and (b) follows Assumption 2.4.2. □

### B.1.2 Main Proof

In this section, we need Assumptions 2.3.1, 2.3.2, 2.4.1, 2.4.2, 2.3.3, and $\eta\leq\frac{1}{4L}$.

Firstly, we have the update rule of $y_t$

$$\begin{aligned}
y_{t+1}&=x_{t+1}-e_{t+1}=x_{t+1}-\frac{1}{K}\sum_{k=1}^{K}e_{k,t+1}\\
&=x_t-\frac{1}{K}\sum_{k=1}^{K}\mathcal{C}(\eta g_{k,t}+e_{k,t})-\frac{1}{K}\sum_{k=1}^{K}(\eta g_{k,t}+e_{k,t}-\mathcal{C}(\eta g_{k,t}+e_{k,t})) \qquad \text{(B–16)}\\
&=x_t-\frac{1}{K}\sum_{k=1}^{K}(\eta g_{k,t}+e_{k,t})=y_t-\frac{1}{K}\sum_{k=1}^{K}\eta g_{k,t}=y_t-\eta g_t\,.
\end{aligned}$$

According to the Liptschitz gradient assumption,

$$\begin{aligned}
&\mathbb{E}[F_{\mathcal{S}}(y_{t+1})-F_{\mathcal{S}}(y_t)]\leq\mathbb{E}\langle\nabla F_{\mathcal{S}}(y_t),y_{t+1}-y_t\rangle+\frac{L}{2}\mathbb{E}\|y_{t+1}-y_t\|_2^2\\
&=\mathbb{E}\langle\nabla F_{\mathcal{S}}(y_t),-\frac{\eta}{K}\sum_{k=1}^{K}g_{k,t})\rangle+\frac{\eta^2L}{2}\mathbb{E}\|\frac{1}{K}\sum_{k=1}^{K}g_{k,t}\|_2^2\\
&=\underbrace{-\frac{\eta}{K}\sum_{k=1}^{K}\mathbb{E}\langle\nabla F_{\mathcal{S}}(y_t),\nabla F_{\mathcal{S}}(x_t-\lambda e_{k,t})\rangle}_{①}+\frac{\eta^2L}{2}\underbrace{\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t-\lambda e_{k,t})\|_2^2}_{②}+\frac{\eta^2L\sigma^2}{2K}\,.
\end{aligned} \qquad \text{(B–17)}$$

For term ①, we have

$$
\begin{aligned}
① &= -\eta\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 - \frac{\eta}{K}\sum_{k=1}^{K}\mathbb{E}\langle\nabla F_{\mathcal{S}}(y_t), \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)\rangle \\
&\leq -\frac{\eta}{2}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + \frac{\eta}{2K}\sum_{k=1}^{K}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)\|_2^2 \\
&\leq -\frac{\eta}{2}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + \frac{\eta L^2}{2K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 .
\end{aligned}
\tag{B–18}
$$

For term ②, we have

$$
\begin{aligned}
② &\leq \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}[2\|\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(y_t)\|_2^2 + 2\|\nabla F_{\mathcal{S}}(y_t)\|_2^2] \\
&\leq 2\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + \frac{2L^2}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 .
\end{aligned}
\tag{B–19}
$$

Replace ① and ② with their bounds and we have

$$
\begin{aligned}
&\mathbb{E}[f(y_{t+1}) - f(y_t)] \\
&\leq (-\frac{\eta}{2} + \eta^2 L)\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + (\frac{\eta L^2}{2} + \eta^2 L^3)\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 + \frac{\eta^2 L\sigma^2}{2K} \\
&\overset{(a)}{\leq} -\frac{\eta}{4}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + \eta L^2 \cdot \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 + \frac{\eta^2 L\sigma^2}{2K} ,
\end{aligned}
\tag{B–20}
$$

where (a) is due to the assumption $\eta \leq \frac{1}{4L}$ for simplicity. Rearrange and sum from $t = 0$ to $T - 1$, we will have

$$
\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 \leq \frac{4\mathbb{E}[F_{\mathcal{S}}(y_0) - F_{\mathcal{S}}(y_T)]}{\eta T} + \frac{2\eta L\sigma^2}{K} + 4L^2 \cdot \frac{1}{KT}\sum_{t=0}^{T-1}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 \\
&\overset{(a)}{\leq} \frac{4\mathbb{E}[F_{\mathcal{S}}(y_0) - F_{\mathcal{S}}(y^*)]}{\eta T} + \frac{2\eta L\sigma^2}{K} + \frac{4}{(\sqrt{\frac{1-\delta/2}{1-\delta}} - 1)^2}\eta^2 L^2[B_1\sigma^2 + 2(1-\lambda)^2 M^2] \\
&= \frac{4\mathbb{E}[F_{\mathcal{S}}(y_0) - F_{\mathcal{S}}(y^*)]}{\eta T} + \frac{2\eta L\sigma^2}{K} \\
&\quad + \frac{4}{(\sqrt{\frac{1-\delta/2}{1-\delta}} - 1)^2}\eta^2 L^2[\frac{K-1}{K^2}\sigma^2 + (\frac{1}{K} - \lambda)^2\sigma^2 + 2(1-\lambda)^2 M^2] ,
\end{aligned}
\tag{B–21}
$$

where (a) follows Lemma B.1.2. Let $\eta = \mathcal{O}(\sqrt{\frac{K}{T}})$, we have the convergence rate

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2\\&= \mathcal{O}(\frac{1}{\sqrt{KT}} + \frac{K}{T}) \overset{K=\mathcal{O}(T^{1/3})}{=} \mathcal{O}(\frac{1}{\sqrt{KT}})\,.\end{aligned} \tag{B–22}$$

If we are only interested in $\delta$, $\sigma^2$ and $M^2$, let $\lambda = \frac{\frac{1}{K}\sigma^2+2M^2}{\sigma^2+2M^2}$ and we have

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2\\&= \mathcal{O}(\frac{1}{(\sqrt{(1-\delta/2)/(1-\delta)}-1)^2}\cdot(\frac{K-1}{K^2}\sigma^2 + \frac{2(1-\frac{1}{K})^2\sigma^2M^2}{\sigma^2+2M^2}))\,.\end{aligned} \tag{B–23}$$

## B.2 Proof of Generalization of DEF(-A) (Theorem 2.4.2)

We use uniform stability to bound the generalization error. Let $\mathcal{S} = \{\xi_1, \xi_2, \cdots, \xi_N\}$ be the training dataset of size $N$, where each data $\xi_n$ is sampled from distribution $\mathcal{D}$. Let $\mathcal{S}^{(n)} = \{\xi_1', \xi_2', \cdots, \xi_N'\} = \{\xi_1, \xi_2, \cdots, \xi_{n-1}, \xi_n', \xi_{n+1}, \cdots, \xi_N\}$ be another training datasets of size $N$. We can see that $\mathcal{S}$ and $\mathcal{S}^{(n)}$ only differs in the $n^{\text{th}}$ data. Let the models trained on $\mathcal{S}$ and $\mathcal{S}^{(i)}$ be $x_t$ and $\widetilde{x}_t$ respectively for DEF-A. For DEF, they will be $y_t$ and $\widetilde{y}_t$ correspondingly.

In each iteration $t$ and under the same random sampling procedure, all workers select the same data from $\mathcal{S}$ and $\mathcal{S}^{(n)}$ with probability $\binom{N-1}{K}/\binom{N}{K} = \frac{N-K}{N}$, while one of the workers selects different data from $\mathcal{S}$ and $\mathcal{S}^{(n)}$ with probability $1 - \binom{N-1}{K}/\binom{N}{K} = \frac{K}{N}$. For simplicity, let $G = \sqrt{\sigma^2+M^2}$, $B_2 = |\frac{1}{K} - \lambda| + \frac{K-1}{K}$. Following [44] (Theorem 3.8), we only need to bound ($x_t$ for DEF-A and $y_t$ for DEF)

$$\mathbb{E}[f(y_t;\xi) - f(\widetilde{y}_T;\xi)] \le \frac{Kt_0}{N} + G\mathbb{E}[\|y_T - \widetilde{y}_T\|_2 | y_{t_0} - \widetilde{y}_{t_0} = 0]\,. \tag{B–24}$$

### B.2.1 Generalization Error of DEF

In this section, we consider non-convex objective functions. We need Assumptions 2.3.1, 2.3.2, 2.4.1, 2.4.2, 2.3.3 and $\eta_t \leq \frac{c}{t+1}$. We first consider $y_t$ selecting the same data at iteration $t$.

$$
\begin{aligned}
&\|y_{t+1} - \widetilde{y}_{t+1}\|_2 = \|y_t - \eta_t g_t - \widetilde{y}_t + \eta_t \widetilde{g}_t\|_2 \leq \|y_t - \widetilde{y}_t\|_2 + \eta_t \|g_t - \widetilde{g}_t\|_2 \\
&= \|y_t - \widetilde{y}_t\|_2 + \eta_t \|\frac{1}{K}\sum_{k=1}^{K}[\nabla f(x_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla f(\widetilde{x}_t - \lambda \widetilde{e}_{k,t}; \xi_{k,t})]\|_2 \\
&\leq \|y_t - \widetilde{y}_t\|_2 + \frac{\eta_t}{K}\sum_{k=1}^{K}\|\nabla f(y_t + e_t - \lambda e_{k,t}; \xi_{k,t}) - \nabla f(\widetilde{y}_t + \widetilde{e}_t - \lambda \widetilde{e}_{k,t}; \xi_{k,t})\|_2 \\
&\leq (1 + \eta_t L)\|y_t - \widetilde{y}_t\|_2 + \frac{\eta_t L}{K}\sum_{k=1}^{K}\|(e_t - \lambda e_{k,t}) - (\widetilde{e}_t - \lambda \widetilde{e}_{k,t})\|_2 \\
&= (1 + \eta_t L)\|y_t - \widetilde{y}_t\|_2 + \frac{\eta_t L}{K}\sum_{k=1}^{K}\|(\frac{1}{K} - \lambda)(e_{k,t} - \widetilde{e}_{k,t}) + \frac{1}{K}\sum_{k'=1, k' \neq k}^{K}(e_{k',t} - \widetilde{e}_{k',t})\|_2 \\
&\leq (1 + \eta_t L)\|y_t - \widetilde{y}_t\|_2 + \frac{\eta_t L B_2}{K}\sum_{k=1}^{K}\underbrace{\|e_{k,t} - \widetilde{e}_{k,t}\|_2}_{①} .
\end{aligned}
\tag{B–25}
$$

Now we consider term ①. Following the same procedures in Lemma B.1.3, but let $\eta_t \leq \frac{c}{t+1}$ and $\beta = \frac{\delta}{2(1-\delta)}$, we have

$$
\begin{aligned}
&\mathbb{E}\|e_{k,t} - \widetilde{e}_{k,t}\|_2^2 \leq (1-\delta)(1 + \frac{1}{\beta}) \cdot 2(\sigma^2 + M^2)\sum_{t'=0}^{t-1}[(1-\delta)(1+\beta)]^{t-1-t'}\eta_{t'}^2 \\
&\leq 2(1-\delta)(1 + \frac{1}{\beta})(\sigma^2 + M^2)c^2\sum_{t'=0}^{t-1}\frac{1}{(t'+1)^2} \\
&\leq 2(1-\delta)(1 + \frac{1}{\beta})(\sigma^2 + M^2)c^2[1 + (-\frac{1}{t'+1})|_0^{t-1}] \\
&\leq 4(1-\delta)(1 + \frac{1}{\beta})(\sigma^2 + M^2)c^2 \\
&= \frac{4(1-\delta)(2-\delta)}{\delta}G^2c^2 .
\end{aligned}
\tag{B–26}
$$

Because $(\mathbb{E}\|e_{k,t} - \widetilde{e}_{k,t}\|_2)^2 \leq \mathbb{E}\|e_{k,t} - \widetilde{e}_{k,t}\|_2^2$, we have

$$
\mathbb{E}① \leq \sqrt{\mathbb{E}\|e_{k,t} - \widetilde{e}_{k,t}\|_2^2} \leq 2Gc\underbrace{\sqrt{\frac{(1-\delta)(2-\delta)}{\delta}}}_{B_4} = 2B_4Gc .
\tag{B–27}
$$

At iteration $t$, if a worker $k' \in [K]$ selects different data from $\mathcal{S}$ and $\mathcal{S}^{(n)}$, we have

$$\|y_{t+1} - \widetilde{y}_{t+1}\|_2 \leq \|y_t - \widetilde{y}_t\|_2 + \eta_t \|g_t - \widetilde{g}_t\|_2 \leq \|y_t - \widetilde{y}_t\|_2 + 2G\eta_t \,. \tag{B–28}$$

When we consider both circumstances, we have

$$\begin{aligned}
&\mathbb{E}\|y_{t+1} - \widetilde{y}_{t+1}\|_2 \leq (1 - \frac{K}{N})[(1 + \eta_t L)\mathbb{E}\|y_t - \widetilde{y}_t\|_2 + \eta_t L B_2 \cdot 2B_4 Gc] + \frac{K}{N}[\mathbb{E}\|y_t - \widetilde{y}_t\|_2 + 2G\eta_t] \\
&= [1 + (1 - \frac{K}{N})\eta_t L]\mathbb{E}\|y_t - \widetilde{y}_t\|_2 + \underbrace{[\frac{2K}{N}G + 2(1 - \frac{K}{N})LB_2B_4Gc]}_{B_5}\eta_t \\
&\overset{(a)}{\leq} \exp((1 - \frac{K}{N})\eta_t L)\mathbb{E}\|y_t - \widetilde{y}_t\|_2 + B_5\eta_t \,.
\end{aligned} \tag{B–29}$$

Unwind the recurrence with $t = 0, 1, \cdots, T-1$, we have

$$\begin{aligned}
\mathbb{E}\|y_T - \widetilde{y}_T\|_2 &\leq \sum_{t=t_0}^{T-1} B_5 \frac{c}{t+1} \prod_{t'=t+1}^{T-1} \exp((1 - \frac{K}{N})\frac{c}{t+1}L) \\
&= B_5 c \sum_{t=t_0}^{T-1} \frac{1}{t+1} \exp((1 - \frac{K}{N})Lc \sum_{t'=t+1}^{T-1} \frac{1}{t+1}) \\
&\overset{(a)}{\leq} B_5 c \sum_{t=t_0}^{T-1} \frac{1}{t+1} \exp((1 - \frac{K}{N})Lc \log \frac{T}{t+1}) \\
&= B_5 c T^{(1-\frac{K}{N})Lc} \sum_{t=t_0}^{T-1} (t+1)^{-(1-\frac{K}{N})Lc-1} \\
&= B_5 c T^{(1-\frac{K}{N})Lc} \frac{t_0^{-(1-\frac{K}{N})Lc} - T^{-(1-\frac{K}{N})Lc}}{(1-\frac{K}{N})Lc} \\
&\leq \frac{B_5}{(1-\frac{K}{N})L} (\frac{T}{t_0})^{(1-\frac{K}{N})Lc}
\end{aligned} \tag{B–30}$$

where (a) is due to $\sum_{t'=t+1}^{T-1} \frac{1}{t'+1} \leq \int_t^{T-1} \log(t'+1)dt'$. Following Eq. ( B–24),

$$\mathbb{E}[f(y_t;\xi) - f(\widetilde{y}_T;\xi)] \leq \frac{Kt_0}{N} + \underbrace{\frac{B_5 G}{(1-\frac{K}{N})L}}_{B_6} (\frac{T}{t_0})^{(1-\frac{K}{N})Lc} \,. \tag{B–31}$$

The R.H.S is minimized when

$$t_0 = [(\frac{N}{K} - 1)LcB_6]^{1/((1-\frac{K}{N})Lc+1)} T^{(1-\frac{K}{N})Lc/((1-\frac{K}{N})Lc+1)} \,, \tag{B–32}$$

which gives us

$$\mathbb{E}[f(y_T;\xi) - f(\widetilde{y}_T;\xi)] \leq [\frac{K}{N} + \frac{1}{(\frac{N}{K}-1)Lc}][(\frac{N}{K}-1)LcB_6]^{1/((1-\frac{K}{N})Lc+1)}T^{(1-\frac{K}{N})Lc/((1-\frac{K}{N})Lc+1)}\,. \tag{B–33}$$

Note that when $K=1$, $\lambda=1$, we will have $B_2=0$, $B_5=\frac{2K}{N}G$, and $B_6=\frac{2G^2}{(N-1)L}$. Then the R.H.S. equals

$$\begin{aligned}
&[\frac{1}{N} + \frac{1}{(N-1)Lc}](2cG^2)^{1/((1-\frac{1}{N})Lc+1)}T^{(1-\frac{1}{N})Lc/((1-\frac{1}{N})Lc+1)} \\
&= \frac{1+1/(Lc)}{N-1}T(\frac{2cG^2}{T})^{1/((1-\frac{1}{N})Lc+1)} \\
&\overset{(a)}{\leq} \frac{1+1/(Lc)}{N-1}T(\frac{2cG^2}{T})^{1/(Lc+1)} \\
&= \frac{1+1/(Lc)}{N-1}(2cG^2)^{1/(Lc+1)}T^{Lc/(Lc+1)}\,,
\end{aligned} \tag{B–34}$$

which matches the result in [44] for SGD. (a) is due to $\frac{2cG^2}{T} \leq 1$ when $t_0 \leq T$ .

### B.2.2 Generalization Error of DEF-A

In this section, we consider non-convex objective functions and random sparsification which satisfies Assumptions 2.4.2 and 2.3.3. We need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t \leq \frac{c}{t+1}$.

Now we consider $x_t$.

$$\begin{aligned}
\mathbb{E}\|x_{t+1} - \widetilde{x}_{t+1}\|_2 &\overset{(a)}{=} \mathbb{E}\|x_t - \mathcal{C}(\eta_t g_t + e_t) - \widetilde{x}_t + \mathcal{C}(\eta_t\widetilde{g}_t + \widetilde{e}_t)\|_2 \\
&\overset{(a)}{\leq} \mathbb{E}\|x_t - \widetilde{x}_t\|_2 + \mathbb{E}\|\mathcal{C}(\eta_t g_t - \eta_t\widetilde{g}_t)\|_2 + \mathbb{E}\|\mathcal{C}(e_t - \widetilde{e}_t)\|_2 \\
&\leq \mathbb{E}\|x_t - \widetilde{x}_t\|_2 + \mathbb{E}\sqrt{\mathbb{E}_{\mathcal{C}}\|\mathcal{C}(\eta_t g_t - \eta_t\widetilde{g}_t)\|_2^2} + \mathbb{E}\sqrt{\mathbb{E}_{\mathcal{C}}\|\mathcal{C}(e_t - \widetilde{e}_t)\|_2^2} \\
&\overset{(a)}{=} \mathbb{E}\|x_t - \widetilde{x}_t\|_2 + \mathbb{E}\sqrt{\delta\eta_t^2\|g_t - \widetilde{g}_t\|_2^2} + \mathbb{E}\sqrt{\delta\|e_t - \widetilde{e}_t\|_2^2} \\
&= \mathbb{E}\|x_t - \widetilde{x}_t\|_2 + \delta^{\frac{1}{2}}\eta_t\mathbb{E}\|g_t - \widetilde{g}_t\|_2 + \delta^{\frac{1}{2}}\mathbb{E}\|e_t - \widetilde{e}_t\|_2 \\
&\leq \mathbb{E}\|x_t - \widetilde{x}_t\|_2 + \delta^{\frac{1}{2}}\eta_t\mathbb{E}\|g_t - \widetilde{g}_t\|_2 + \delta^{\frac{1}{2}}\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t} - \widetilde{e}_{k,t}\|_2\,.
\end{aligned} \tag{B–35}$$

where (a) is due to the random sparsification. When selecting the same data at iteration $t$, we have

$$\begin{aligned}
&\mathbb{E}\|x_{t+1}-\widetilde{x}_{t+1}\|_2 \\
&\leq \mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{\delta^{\frac{1}{2}}}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2 \\
&\quad+\delta^{\frac{1}{2}}\eta_t\mathbb{E}\|\frac{1}{K}\sum_{k=1}^{K}[\nabla f(x_t-\lambda e_{k,t};\xi_{k,t})-\nabla f(\widetilde{x}_t-\lambda\widetilde{e}_{k,t};\xi_{k,t})]\|_2 \\
&\leq \mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{\delta^{\frac{1}{2}}}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2+\frac{\delta^{\frac{1}{2}}\eta_t L}{K}\sum_{k=1}^{K}\|x_t-\lambda e_{k,t}-\widetilde{x}_t+\lambda\widetilde{e}_{k,t}\|_2 \\
&\leq (1+\delta^{\frac{1}{2}}\eta_t L)\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{\delta^{\frac{1}{2}}(1+\eta_t L\lambda)}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2\,.
\end{aligned} \tag{B–36}$$

When selecting different data at iteration $t$, we have

$$\mathbb{E}\|x_{t+1}-\widetilde{x}_{t+1}\|_2\leq\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\delta^{\frac{1}{2}}\eta_t\cdot 2G+\frac{\delta^{\frac{1}{2}}}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2\,. \tag{B–37}$$

When we consider both circumstances, we have

$$\begin{aligned}
\mathbb{E}\|x_{t+1}-\widetilde{x}_{t+1}\|_2 &\leq (1-\frac{K}{N})[(1+\delta^{\frac{1}{2}}\eta_t L)\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{\delta^{\frac{1}{2}}(1+\eta_t L\lambda)}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2] \\
&\quad+\frac{K}{N}[\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\delta^{\frac{1}{2}}\eta_t\cdot 2G+\frac{\delta^{\frac{1}{2}}}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2] \\
&=[1+(1-\frac{K}{N})\delta^{\frac{1}{2}}\eta_t L]\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{K}{N}2\delta^{\frac{1}{2}}\eta_t G+\delta^{\frac{1}{2}}[1+(1-\frac{K}{N})\eta_t L\lambda]\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}-\widetilde{e}_{k,t}\|_2 \\
&\overset{(a)}{\leq}[1+(1-\frac{K}{N})\delta^{\frac{1}{2}}\eta_t L]\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{K}{N}2\delta^{\frac{1}{2}}\eta_t G+\delta^{\frac{1}{2}}[1+(1-\frac{K}{N})\eta_t L\lambda]\cdot 2B_4Gc \\
&\overset{(b)}{\leq}\exp((1-\frac{K}{N})\delta^{\frac{1}{2}}\eta_t L)\mathbb{E}\|x_t-\widetilde{x}_t\|_2+\frac{K}{N}2\delta^{\frac{1}{2}}\eta_t G+\delta^{\frac{1}{2}}[1+(1-\frac{K}{N})\eta_t L\lambda]\cdot 2B_4Gc\,,
\end{aligned} \tag{B–38}$$

where (a) follows the Eq. ( B–27). Let $\eta_t \leq \frac{c}{t+1}$, we have

$$
\begin{aligned}
\mathbb{E}\|x_T - \widetilde{x}_T\|_2 &\leq \sum_{t=t_0}^{T-1}[\frac{K}{N}2\delta^{\frac{1}{2}}\eta_t G + \delta^{\frac{1}{2}}(1+(1-\frac{K}{N})\eta_t L\lambda)\cdot 2B_4 Gc]\prod_{t'=t+1}^{T-1}\exp((1-\frac{K}{N})\delta^{\frac{1}{2}}\eta_{t'}L) \\
&\leq 2\delta^{\frac{1}{2}}G\sum_{t=t_0}^{T-1}[\frac{K}{N}\frac{c}{t+1} + (1+(1-\frac{K}{N})\frac{c}{t+1}L\lambda)B_4 c]\exp((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc\sum_{t''=t+1}^{T-1}\frac{1}{t''+1}) \\
&\leq 2\delta^{\frac{1}{2}}G\sum_{t=t_0}^{T-1}[\frac{K}{N}\frac{c}{t+1} + (1+(1-\frac{K}{N})\frac{c}{t+1}L\lambda)B_4 c]\exp((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc\log(\frac{T}{t+1})) \\
&= 2\delta^{\frac{1}{2}}G\sum_{t=t_0}^{T-1}[\frac{K}{N}\frac{c}{t+1} + (1+(1-\frac{K}{N})\frac{c}{t+1}L\lambda)B_4 c](\frac{T}{t+1})^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc} \\
&\leq 2\delta^{\frac{1}{2}}Gc[\frac{K}{N}+1+(1-\frac{K}{N})Lc\lambda B_4]T^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc}\sum_{t=t_0}^{T-1}(t+1)^{-(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc} \\
&\leq 2\delta^{\frac{1}{2}}Gc[\frac{K}{N}+1+(1-\frac{K}{N})Lc\lambda B_4]T^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc}\frac{t_0^{-1-(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc} - T^{-1-(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc}}{1+(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc} \\
&\leq \frac{2\delta^{\frac{1}{2}}G[\frac{K}{N}+1+(1-\frac{K}{N})Lc\lambda B_4]}{1+(1-\frac{K}{N})\delta^{\frac{1}{2}}L}(\frac{T}{t_0})^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc}.
\end{aligned}
\tag{B–39}
$$

Following Eq. ( B–24),

$$
\mathbb{E}[f(x_T;\xi) - f(\widetilde{x}_T;\xi)] \leq \frac{Kt_0}{N} + \underbrace{\frac{2\delta^{\frac{1}{2}}G[\frac{K}{N}+1+(1-\frac{K}{N})Lc\lambda B_4]}{1+(1-\frac{K}{N})\delta^{\frac{1}{2}}L}}_{B_7}(\frac{T}{t_0})^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc}. \tag{B–40}
$$

The R.H.S is minimized when

$$
t_0 = [(\frac{N}{K}-1)LcB_7]^{1/((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc+1)}T^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc/((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc+1)}, \tag{B–41}
$$

which gives us

$$
\begin{aligned}
&\mathbb{E}[f(x_T;\xi) - f(\widetilde{x}_T;\xi)] \\
&\leq [\frac{K}{N} + \frac{1}{(\frac{N}{K}-1)Lc}][(\frac{N}{K}-1)LcB_7]^{1/((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc+1)}T^{(1-\frac{K}{N})\delta^{\frac{1}{2}}Lc/((1-\frac{K}{N})\delta^{\frac{1}{2}}Lc+1)}.
\end{aligned}
\tag{B–42}
$$

### B.2.3 Optimization Error of DEF

In this section, we consider non-convex objective functions under Polyak-Łojasiewicz (PL) condition, which establishes the relation between the objective function and the gradient norm. We consider $\delta$-approximate and ring-allreduce compatible compressor. We need Assumptions 2.3.1, 2.3.2, 2.4.1, 2.4.2, 2.3.3, and $\eta_t = \frac{c}{t+1} \leq \frac{1}{4L}$. From Eq. ( B–20) and the PL condition, we have

$$\begin{aligned}\mathbb{E}[F_{\mathcal{S}}(y_{t+1}) - F_{\mathcal{S}}(y_t)] &\leq -\frac{\eta_t}{4}\mathbb{E}\|\nabla F_{\mathcal{S}}(y_t)\|_2^2 + \eta_t L^2 \cdot \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 + \frac{\eta_t^2 L\sigma^2}{2K} \\ &\leq -\frac{\mu\eta_t}{2}\mathbb{E}[F_{\mathcal{S}}(y_t) - F_{\mathcal{S}}(y^*)] + \eta_t L^2 \cdot \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 + \frac{\eta_t^2 L\sigma^2}{2K}\,,\end{aligned} \tag{B–43}$$

where we need $\eta_t \leq \frac{1}{4L}$ following Section B.1. Rearrange,

$$\begin{aligned}\mathbb{E}[F_{\mathcal{S}}(y_{t+1}) - F_{\mathcal{S}}(y^*)] &\leq (1 - \frac{\mu\eta_t}{2})\mathbb{E}[F_{\mathcal{S}}(y_t) - F_{\mathcal{S}}(y^*)] + \eta_t L^2 \cdot \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_t - \lambda e_{k,t}\|_2^2 + \frac{\eta_t^2 L\sigma^2}{2K} \\ &\overset{(a)}{\leq} (1 - \frac{\mu\eta_t}{2})\mathbb{E}[F_{\mathcal{S}}(y_t) - F_{\mathcal{S}}(y^*)] + \frac{B_1\sigma^2 + 2(1-\lambda)^2M^2}{(\sqrt{\frac{1-\delta/2}{1-\delta}} - 1)^2}\eta_t^3 L^2 + \frac{\eta_t^2 L\sigma^2}{2K}\,,\end{aligned} \tag{B–44}$$

where (a) follows Lemma B.1.2. Let $\eta_t = \frac{c}{t+1}$, we have

$$\begin{aligned}&\mathbb{E}[F_{\mathcal{S}}(y_T) - F_{\mathcal{S}}(y^*)] \\ &\leq \mathbb{E}[F_{\mathcal{S}}(y_t) - F_{\mathcal{S}}(y^*)]\underbrace{\prod_{t=0}^{T-1}(1 - \frac{\mu c}{2(t+1)})}_{①} + [\frac{B_1\sigma^2 + 2(1-\lambda)^2M^2}{(\sqrt{\frac{1-\delta/2}{1-\delta}} - 1)^2} + \frac{\sigma^2}{2K}]L \\ &\quad \cdot \underbrace{\sum_{t=0}^{T-1}\frac{c^2}{(t+1)^2}\prod_{t'=t+1}^{T-1}(1 - \frac{\mu c}{2(t+1)})}_{②}\,,\end{aligned} \tag{B–45}$$

where

$$\begin{aligned}① &\leq \prod_{t=0}^{T-1}\exp(-\frac{\mu c}{2(t+1)}) = \exp(-\frac{\mu c}{2})\exp(-\frac{\mu c}{2}\sum_{t=1}^{T-1}\frac{1}{t+1}) \\ &\leq \exp(-\frac{\mu c}{2})\exp(-\frac{\mu c}{2}\log T) = \exp(-\frac{\mu c}{2})T^{-\frac{\mu c}{2}}\,,\end{aligned} \tag{B–46}$$

$$
\begin{aligned}
② &\leq \sum_{t=0}^{T-1} \frac{c^2}{(t+1)^2} \exp(-\frac{\mu c}{2} \sum_{t'=t+1}^{T-1} \frac{1}{t'+1}) \\
&\leq \sum_{t=0}^{T-1} \frac{c^2}{(t+1)^2} \exp(-\frac{\mu c}{2} \log \frac{T}{t+1}) = c^2 T^{-\frac{\mu c}{2}} \sum_{t=0}^{T-1} (t+1)^{\frac{\mu c}{2}-2} .
\end{aligned} \tag{B–47}
$$

When $\frac{\mu c}{2} - 2 \geq 0$,

$$
② \leq \frac{c^2 T^{-\frac{\mu c}{2}}}{\frac{\mu c}{2}-1} ((T+1)^{\frac{\mu c}{2}-1} - 1) \leq \frac{c^2}{\frac{\mu c}{2}-1} (\frac{T+1}{T})^{\frac{\mu c}{2}-1} T^{-1} . \tag{B–48}
$$

When $\frac{\mu c}{2} - 2 \leq 0$ and $\frac{\mu c}{2} - 2 \neq -1$,

$$
② \leq \frac{c^2 T^{-\frac{\mu c}{2}}}{\frac{\mu c}{2}-1} (1 + T^{\frac{\mu c}{2}-1} - 1) = \frac{c^2}{\frac{\mu c}{2}-1} T^{-1} . \tag{B–49}
$$

When $\frac{\mu c}{2} - 2 = -1$,

$$
② \leq c^2 T^{-1} (1 + \log T) . \tag{B–50}
$$

$$
\mathbb{E}[F_{\mathcal{S}}(y_T) - F_{\mathcal{S}}(y^*)] = \widetilde{\mathcal{O}}(T^{-\frac{\mu c}{2}} + T^{-1}) . \tag{B–51}
$$

### B.2.4 Optimization Error of DEF-A

In this section, we consider non-convex objective functions under Polyak-Łojasiewicz (PL) condition, which establishes the relation between the objective function and the gradient norm. The compressor we consider here is random sparsification which satisfies Assumptions 2.4.2 and 2.3.3. We need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t = \frac{c}{t+1} \leq \frac{1}{8L}$.

$$
\begin{aligned}
&\mathbb{E}[F_{\mathcal{S}}(x_{t+1}) - F_{\mathcal{S}}(x_t)] \leq \mathbb{E}\langle \nabla F_{\mathcal{S}}(x_t), x_{t+1} - x_t \rangle + \frac{L}{2} \mathbb{E}\|x_{t+1} - x_t\|_2^2 \\
&\overset{(a)}{=} \underbrace{\mathbb{E}\langle \nabla F_{\mathcal{S}}(x_t), -\mathcal{C}(\eta_t g_t + e_t) \rangle}_{①} + \frac{L}{2} \underbrace{\mathbb{E}\|\mathcal{C}(\eta_t g_t + e_t)\|_2^2}_{②} ,
\end{aligned} \tag{B–52}
$$

where (a) follows Assumption 2.3.3. For term ①,

$$
\begin{aligned}
&① \overset{(a)}{=} -\mathbb{E}\langle \nabla F_{\mathcal{S}}(x_t), \delta(\eta_t g_t + e_t)\rangle = -\delta\eta_t \mathbb{E}\langle \nabla F_{\mathcal{S}}(x_t), \frac{1}{K}\sum_{k=1}^{K} \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) + \frac{e_t}{\eta_t}\rangle \\
&= -\delta\eta_t \mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 - \delta\eta_t \mathbb{E}\langle \nabla F_{\mathcal{S}}(x_t), \frac{1}{K}\sum_{k=1}^{K} \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) + \frac{e_t}{\eta_t} - \nabla F_{\mathcal{S}}(x_t)\rangle \\
&\leq -\frac{\delta\eta_t}{2}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{\delta\eta_t}{2}\mathbb{E}\|\frac{1}{K}\sum_{k=1}^{K} \nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(x_t) + \frac{e_t}{\eta_t}\|_2^2 \\
&\leq -\frac{\delta\eta_t}{2}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{\delta\eta_t L^2}{2K}\sum_{k=1}^{K}\mathbb{E}\|\lambda e_{k,t}\|_2^2 + \frac{\delta}{2\eta_t}\mathbb{E}\|e_t\|_2^2 \\
&\leq -\frac{\delta\eta_t}{2}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{\delta(\eta_t^2 L^2\lambda^2 + 1)}{2\eta_t K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 ,
\end{aligned}
\tag{B–53}
$$

where (a) is due to the random sparsification compressor. For term ②,

$$
\begin{aligned}
&② \overset{(a)}{=} \delta\mathbb{E}\|\eta_t g_t + e_t\|_2^2 \leq 2\delta\eta_t^2\mathbb{E}\|g_t\|_2^2 + 2\delta\mathbb{E}\|e_t\|_2^2 \\
&= 2\delta\eta_t^2\mathbb{E}\|\frac{1}{K}\sum_{k=1}^{K}\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t})\|_2^2 + \frac{2\delta\eta_t^2\sigma^2}{K} + 2\delta\mathbb{E}\|e_t\|_2^2 \\
&\leq \frac{2\delta\eta_t^2}{K}\sum_{k=1}^{K}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t - \lambda e_{k,t}) - \nabla F_{\mathcal{S}}(x_t) + \nabla F_{\mathcal{S}}(x_t)\|_2^2 + 2\delta\mathbb{E}\|e_t\|_2^2 + \frac{2\delta\eta_t^2\sigma^2}{K} \\
&\leq 4\delta\eta_t^2\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{4\delta\eta_t^2 L^2}{K}\sum_{k=1}^{K}\mathbb{E}\|\lambda e_{k,t}\|_2^2 + 2\delta\mathbb{E}\|e_t\|_2^2 + \frac{2\delta\eta_t^2\sigma^2}{K} \\
&= 4\delta\eta_t^2\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{2\delta(2\eta_t^2 L^2\lambda^2 + 1)}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 + \frac{2\delta\eta_t^2\sigma^2}{K} ,
\end{aligned}
\tag{B–54}
$$

where (a) is due to the random sparsification compressor. Put them together, let $\eta_t \leq \frac{1}{8L}$, and we have

$$
\begin{aligned}
&\mathbb{E}[F_{\mathcal{S}}(x_{t+1}) - F_{\mathcal{S}}(x_t)] \\
&\leq -\frac{\delta\eta_t}{2}(1 - 4\eta_t L)\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{\delta(1 + 2\eta_t L)(1 + 2\eta_t^2 L^2\lambda^2)}{2\eta_t K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 + \frac{\delta\eta_t^2 L\sigma^2}{K} \\
&\leq -\frac{\delta\eta_t}{4}\mathbb{E}\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 + \frac{\delta(1+\lambda^2)}{\eta_t}\cdot\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|e_{k,t}\|_2^2 + \frac{\delta\eta_t^2 L\sigma^2}{K} \\
&\overset{(a)}{\leq} -\frac{\mu\delta\eta_t}{2}\mathbb{E}[F_{\mathcal{S}}(x_t) - F_{\mathcal{S}}(x^*)] + \frac{\delta(1+\lambda^2)\eta_t(\sigma^2 + M^2)}{(1/\sqrt{1-\delta} - 1)^2} + \frac{\delta\eta_t^2 L\sigma^2}{K} ,
\end{aligned}
\tag{B–55}
$$

where (a) is due to $\|\nabla F_{\mathcal{S}}(x_t)\|_2^2 \geq 2\mu(F_{\mathcal{S}}(x_t) - F_{\mathcal{S}}(x^*))$ according to the PL condition and Lemma B.1.3. Rearrange,

$$\mathbb{E}[F_{\mathcal{S}}(x_{t+1}) - F_{\mathcal{S}}(x^*)] \leq (1 - \frac{\mu\delta\eta_t}{2})\mathbb{E}[F_{\mathcal{S}}(x_t) - F_{\mathcal{S}}(x^*)] + \frac{\delta(1+\lambda^2)\eta_t(\sigma^2+M^2)}{(1/\sqrt{1-\delta}-1)^2} + \frac{\delta\eta_t^2 L\sigma^2}{K}\,. \tag{B–56}$$

Let $\eta_t = \frac{c}{t+1}$ and $G = \sqrt{\sigma^2+M^2}$. Take this recurrence from $t=0$ to $T-1$ and we have

$$\begin{aligned}\mathbb{E}[F_{\mathcal{S}}(x_T) - F_{\mathcal{S}}(x^*)] \leq\ & \mathbb{E}[F_{\mathcal{S}}(x_0) - F_{\mathcal{S}}(x^*)] \underbrace{\prod_{t=0}^{T-1}(1 - \frac{\mu\delta c}{2(t+1)})}_{③} \\ & + \underbrace{\frac{\delta(1+\lambda^2)G^2}{(1/\sqrt{1-\delta}-1)^2}\sum_{t'=0}^{T-1}\frac{c}{t'+1}\underbrace{\prod_{t=t'+1}^{T-1}(1-\frac{\mu\delta c}{2(t+1)})}_{④}}_{⑤} + \underbrace{\frac{\delta L\sigma^2}{K}\sum_{t'=0}^{T-1}\frac{c^2}{(t'+1)^2}\underbrace{\prod_{t=t'+1}^{T-1}(1-\frac{\mu\delta c}{2(t+1)})}_{④}}_{⑥}\,,\end{aligned} \tag{B–57}$$

where

$$\begin{aligned}③ &\leq \prod_{t=0}^{T-1}\exp(-\frac{\mu\delta c}{2(t+1)}) = \exp(-\frac{\mu\delta c}{2}\sum_{t=0}^{T-1}\frac{1}{t+1}) \leq \exp(-\frac{\mu\delta c}{2})\exp(-\frac{\mu\delta c}{2}\sum_{t=1}^{T-1}\frac{1}{t+1}) \\ &\leq \exp(-\frac{\mu\delta c}{2})\exp(-\frac{\mu\delta c}{2}\log(T)) = \exp(-\frac{\mu\delta c}{2})T^{-\frac{\mu\delta c}{2}}\,.\end{aligned} \tag{B–58}$$

$$\begin{aligned}④ &\leq \prod_{t=t'+1}^{T-1}\exp(-\frac{\mu\delta c}{2(t+1)}) = \exp(-\frac{\mu\delta c}{2}\sum_{t=t'+1}^{T-1}\frac{1}{t+1}) \\ &\leq \exp(-\frac{\mu\delta c}{2}\log(\frac{T}{t'+1})) = (\frac{T}{t'+1})^{-\frac{\mu\delta c}{2}}\,,\end{aligned} \tag{B–59}$$

When $\frac{\mu\delta c}{2} \geq 1$,

$$⑤ \leq cT^{-\frac{\mu\delta c}{2}}\sum_{t'=0}^{T-1}(t'+1)^{\frac{\mu\delta c}{2}-1} \leq cT^{-\frac{\mu\delta c}{2}}\cdot\frac{2}{\mu\delta c}[(T+1)^{\frac{\mu\delta c}{2}}-1] \leq \frac{2}{\mu\delta}(\frac{T+1}{T})^{\frac{\mu\delta c}{2}}\,. \tag{B–60}$$

When $0 < \frac{\mu\delta c}{2} \leq 1$,

$$⑤ \leq cT^{-\frac{\mu\delta c}{2}}\sum_{t'=0}^{T-1}(t'+1)^{\frac{\mu\delta c}{2}-1} \leq cT^{-\frac{\mu\delta c}{2}}\cdot[1+\frac{2}{\mu\delta c}(T^{\frac{\mu\delta c}{2}}-1)] \leq \frac{2}{\mu\delta}\,. \tag{B–61}$$

When $\frac{\mu\delta c}{2} \geq 2$,

$$\begin{aligned} ⑥ &\leq c^2 T^{-\frac{\mu\delta c}{2}} \sum_{t'=0}^{T-1} (t'+1)^{\frac{\mu\delta c}{2}-2} \leq c^2 T^{-\frac{\mu\delta c}{2}} \cdot \frac{1}{\mu\delta c/2-1} [(T+1)^{\frac{\mu\delta c}{2}-1} - 1] \\ &\leq \frac{c^2}{\mu\delta c/2-1} (\frac{T+1}{T})^{\frac{\mu\delta c}{2}-1} T^{-1} \,. \end{aligned} \tag{B–62}$$

When $\frac{\mu\delta c}{2} \leq 2$, $\frac{\mu\delta c}{2} \neq 1$,

$$⑥ \leq c^2 T^{-\frac{\mu\delta c}{2}} \sum_{t'=0}^{T-1} (t'+1)^{\frac{\mu\delta c}{2}-2} \leq c^2 T^{-\frac{\mu\delta c}{2}} \cdot [1 + \frac{1}{\mu\delta c/2-1} (T^{\frac{\mu\delta c}{2}-1} - 1)] \leq \frac{c^2}{\mu\delta c/2-1} T^{-1} \,. \tag{B–63}$$

When $\frac{\mu\delta c}{2} = 1$,

$$⑥ \leq c^2 T^{-1} \sum_{t'=0}^{T-1} (t'+1)^{-1} \leq c^2 T^{-1} \cdot (1 + \log T) \,. \tag{B–64}$$

$$\mathbb{E}[F_{\mathcal{S}}(x_T) - F_{\mathcal{S}}(x^*)] = \widetilde{\mathcal{O}}(T^{-\frac{\mu\delta c}{2}} + T^{-1} + (1/\sqrt{1-\delta} - 1)^{-2}) \,. \tag{B–65}$$

### B.3 Proof of Generalization of SGD-(IA) (Theorem 2.4.3)

For consistency, let $\{y_t\}$ be the SGD solution path, i.e.,

$$y_{t+1} = y_t - \eta_t \nabla f(y_t; \xi_t) = y_t - \eta_t g_t \,. \tag{B–66}$$

Let $\{x_t\}$ be the SGD-IA solution path with $x_0 = y_0$, where IA denotes momentum iterative averaging, i.e.,

$$x_{t+1} = (1-\delta) x_t + \delta y_{t+1} = x_t + \delta(y_{t+1} - x_t) \text{ and } x_0 = y_0 \,, \tag{B–67}$$

where $0 < 1 - \delta < 1$ is the momentum constant. Then,

$$x_t = (1-\delta)^t x_0 + \sum_{t'=1}^{t} \delta(1-\delta)^{t-t'} y_{t'} = (1-\delta)^t y_0 + \sum_{t'=1}^{t} \delta(1-\delta)^{t-t'} y_{t'} \,. \tag{B–68}$$

Let $K = 1$ and $\lambda = 1$, then DEF is identical to SGD. Following Lemma B.3.1, SGD-IA is a special case of DEF-A with $\mathcal{C}(\Delta) = \delta\Delta$. Note that this compressor does not compress the message volume.

**Lemma B.3.1.** *Let $g_t = \nabla f(y_t; \xi_t)$, $x_0 = y_0$, and $\mathcal{C}(-\Delta) = \mathcal{C}(\Delta)$. If $y_{t+1} = y_t - \eta_t g_t$ and $x_{t+1} = x_t + \mathcal{C}(y_{t+1} - x_t)$, then the update rule of $x_t$ is identical to DEF-A when $K = 1$, $\lambda = 1$ with compressor $\mathcal{C}$, i.e.,*

$$\begin{aligned} x_{t+1} &= x_t - \mathcal{C}(\eta_t g_t + e_t)\,, \\ e_{t+1} &= \eta_t g_t + e_t - \mathcal{C}(\eta_t g_t + e_t),\ e_0 = 0 \\ y_{t+1} &= y_t - \eta_t g_t\,. \end{aligned} \tag{B–69}$$

*Proof.* We just need to verify $x_{t+1} = x_t + \mathcal{C}(y_{t+1} - x_t)$ with the 3 equations above. We have

$$x_{t+1} = x_0 - \sum_{t'=0}^{t} \mathcal{C}(\eta_{t'} g_{t'} + e_{t'}),\ y_{t+1} = y_0 - \sum_{t'=0}^{t} \eta_t g_{t'}\,. \tag{B–70}$$

Then

$$\begin{aligned} x_{t+1} - e_{t+1} &= x_0 - e_{t+1} - \sum_{t'=0}^{t} \mathcal{C}(\eta_{t'} g_{t'} + e_{t'}) \\ &= x_0 - e_{t+1} - \mathcal{C}(\eta_t g_t + e_t) - \sum_{t'=0}^{t-1} \mathcal{C}(\eta_{t'} g_{t'} + e_{t'}) \\ &= x_0 - \eta_t g_t - e_t - \sum_{t'=0}^{t-1} \mathcal{C}(\eta_{t'} g_{t'} + e_{t'}) \\ &= \cdots = x_0 - \sum_{t'=0}^{t} \eta_t g_t = y_{t+1}\,, \end{aligned} \tag{B–71}$$

$$\begin{aligned} x_t + \mathcal{C}(y_{t+1} - x_t) &= x_t + \mathcal{C}(x_{t+1} - e_{t+1} - x_t) \\ &= x_t + \mathcal{C}(-\mathcal{C}(\eta_t g_t + e_t) - e_{t+1}) \\ &= x_t + \mathcal{C}(-\eta_t g_t - e_t) = x_{t+1}\,, \end{aligned} \tag{B–72}$$

which completes the proof. □

### B.3.1 Generalization Error of SGD

In this section, we need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t \le \frac{c}{t+1}$. Following Sec. B.2.1 with $K = 1$ and $\lambda = 1$, we have

$$\mathbb{E}[f(y_T; \xi) - f(\widetilde{y}_T; \xi)] = \mathcal{O}(T^{(1-\frac{1}{N})Lc/((1-\frac{1}{N})Lc+1)})\,. \tag{B–73}$$

### B.3.2 Generalization Error of SGD-IA

In this section, we need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t \leq \frac{c}{t+1}$. Following Sec. B.2.2 with $K = 1$, $\lambda = 1$ and the compressor replaced with $\mathcal{C}(\Delta) = \delta\Delta$ which satisfies Assumptions 2.4.2 and 2.3.3, we have $\delta^{\frac{1}{2}} \to \delta$ in Eq. ( B–35). All the other procedures are the same, thus

$$\mathbb{E}[f(x_T;\xi) - f(\widetilde{x}_t;\xi)] = \mathcal{O}(T^{(1-\frac{1}{N})\delta Lc/((1-\frac{1}{N})\delta Lc+1)})\,. \tag{B–74}$$

### B.3.3 Optimization Error of SGD

In this section, we need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t = \frac{c}{t+1} \leq \frac{1}{4L}$. Following Sec. B.2.3 with $K = 1$ and $\lambda = 1$, we have

$$\mathbb{E}[F_{\mathcal{S}}(y_T) - F_{\mathcal{S}}(y^*)] = \widetilde{\mathcal{O}}(T^{-\frac{\mu c}{2}} + T^{-1})\,. \tag{B–75}$$

### B.3.4 Optimization Error of SGD-IA

In this section, we need Assumptions 2.3.1, 2.3.2, 2.4.1, and $\eta_t = \frac{c}{t+1} \leq \frac{1}{8\delta L}$. Following Sec. B.2.4 with $K = 1$, $\lambda = 1$ and the compressor replaced with $\mathcal{C}(\Delta) = \delta\Delta$ which satisfies Assumptions 2.4.2 and 2.3.3, we have the same bound as Eq. ( B–53), but $\textcircled{2} = \delta^2\mathbb{E}\|\eta_t g_t + e_t\|_2^2$ in Eq. ( B–54), which leads to the need for $\eta_t \leq \frac{1}{8\delta L}$. All the other procedures are the same, therefore

$$\mathbb{E}[F_{\mathcal{S}}(x_T) - F_{\mathcal{S}}(x^*)] = \widetilde{\mathcal{O}}(T^{-\frac{\mu\delta c}{2}} + T^{-1} + (1/\sqrt{1-\delta} - 1)^{-2})\,. \tag{B–76}$$

# Appendix C "Improve the Performance with Data Privacy"

## C.1 Additional Dataset Information

Table 19: Prostate dataset: number of data (3D image) in each client.

| Client | 1 | 2 | 3 | 4 | 5 | 6 | Global |
|---|---|---|---|---|---|---|---|
| Train | 10 | 16 | 18 | 18 | 25 | 50 | 137 |
| Val | 5 | 8 | 9 | 9 | 12 | 25 | 68 |
| Test | 4 | 8 | 8 | 8 | 13 | 24 | 65 |

Figure 16: Representative original 2D image in retinal dataset (low data similarity). First row: client 1 to 3. Second row: client 4 to 6.

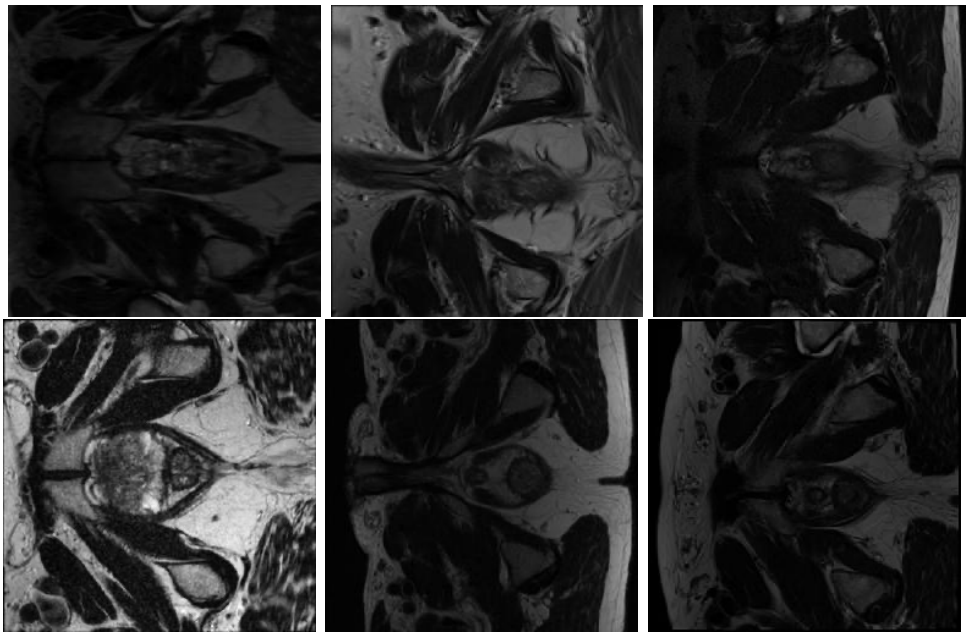

Figure 17: Representative original 2D image slices in prostate dataset (high data similarity). First row: client 1 to 3. Second row: client 4 to 6. E.g., the first slice comes from a 3D image in client 1.

**Algorithm 5** FedSM-extra inference.

**Input:** data $x$, model $(w_g, \{w_{p,k}\}_{k=1}^K, w_s)$
$\widehat{y}_s = f_s(w_s; x)$
$k = \arg\max(\widehat{y}_s) \in \{0, 1, \cdots, K\}$
**if** $k > 0$ **then**
    $\widehat{y} = f(w_{p,k}; x)$
**else**
    $\widehat{y} = f(w_g; x)$
**end if**
**Output:** $\widehat{y}$

### C.2 FedSM-extra Algorithm

For the training of FedSM-extra, we train the global model and personalized models first, and then train the model selector, which incurs extra $\Delta R$ training rounds. In each training round of FedSM-extra, the communication cost is $2w_g$ in the previous $R$ rounds (the global

and personalized models have the same model architecture). It becomes $w_s$ in the extra $\Delta R$ training rounds.

For the inference of FedSM-extra, both the global model and personalized models can be selected. Therefore $k \in \{0, 1, \cdots, K\}$ (For FedSM, $k \in \{1, 2, \cdots, K\}$).

### C.3 Proof of SoftPull Convergence

Let the current/total training rounds be $r/R$, current/total local training steps be $m/M$, the current/total global training step be $t/T$. We denote the personalized model during training as $w_{p,k}^{r,m}$. For simplicity we use $f_k$ to denote loss $L_{\mathcal{D}_k}$.

After the local training in the last training round $r-1$ finishes, we get model $w_{p,k}^{r-1,M}$ and want to

$$\min \sum_{k=1}^{K} f_k(\frac{1}{\lambda} w_{p,k}^{r-1,M} - \frac{1-\lambda}{\lambda} \frac{1}{K-1} \sum_{k'=1,k' \neq k}^{K} w_{p,k'}^{r-1,M}) \tag{C–1}$$

In the beginning of the current training round $r$, from Eq. (3–11), we will have

$$w_{p,k}^{r,0} = \lambda w_{p,k}^{r-1,M} + (1-\lambda) \frac{1}{K-1} \sum_{k'=1,k' \neq k}^{K} w_{p,k'}^{r-1,M} \tag{C–2}$$

In the current training round $r$, we consider two stages. The first stage is a transition from then end of training round $r-1$ to the start of the current training round $r$, while the second stage is the start to the end of current training round $r$. Let $\lambda' = \frac{K\lambda - 1}{K-1}$, $1 - \lambda' = \frac{K}{K-1}(1-\lambda)$, then

$$w_{p,k}^{r,0} = \lambda' w_{p,k}^{r-1,M} + (1-\lambda') \overline{w}_{p,k}^{r-1,M} \tag{C–3}$$

where the bar denotes an average over all clients $k \in \{1, 2, \cdots, K\}$. It can be clearly seen that when the data distributions of clients are very similar, we set $\lambda = \frac{1}{K}$, $\lambda' = 0$, i.e., the "hard averaging" in FedAvg. When the data distributions are not similar at all, we set $\lambda = 1$, $\lambda' = 1$ to only do local training. In other circumstances, theoretically we should set $\lambda \in [\frac{1}{K}, 1]$, $\lambda' \in [0, 1]$ according to the data similarity.

### C.3.1 Difference

Suppose the stochastic gradient at iteration $(r, m)$ is $\nabla f_k(w_{p,k}^{r,m}, x_{p,k}^{r,m})$ and the expected gradient is $\nabla f_k(w_{p,k}^{r,m}) = \mathbb{E}_{x_{p,k}^{r,m} \in \mathcal{D}_k} \nabla f_k(w_{p,k}^{r,m}, x_{p,k}^{r,m}) = \mathbb{E} \nabla f_k(w_{p,k}^{r,m}, x_{p,k}^{r,m})$. We need to bound

$$\begin{aligned} &\|(w_{p,k}^{r+1,0} - w_{p,k}^{r,M})\|_2^2 = \|(1-\lambda)(w_{p,k}^{r,M} - \frac{1}{K-1} \sum_{k'=1,k' \neq k}^{K} w_{p,k'}^{r,M})\|_2^2 \\ &= (1-\lambda)^2 \|w_{p,k}^{r,M} - \frac{1}{K-1}(K \overline{w}_{p,k}^{r,M} - w_{p,k}^{r,M})\|_2^2 = \frac{(1-\lambda)^2 K^2}{(K-1)^2} \|w_{p,k}^{r,M} - \overline{w}_{p,k}^{r,M}\|_2^2 \end{aligned} \tag{C–4}$$

where

$$\begin{aligned} &\mathbb{E}\|w_{p,k}^{r,M} - \overline{w}_{p,k}^{r,M}\|_2^2 \\ &= \eta^2 \mathbb{E}\| \sum_{r'=0}^{r} (\lambda')^{r-r'} \sum_{m=0}^{M-1} [\nabla f_k(w_{p,k}^{r',m}, x_{p,k}^{r',m}) - \overline{\nabla f_k}(w_{p,k}^{r',m}, x_{p,k}^{r',m})]\|_2^2 \\ &\leq \eta^2 (\sum_{r'=0}^{r} (\lambda')^{r-r'})^2 \mathbb{E}\| \sum_{r'=0}^{r} \frac{(\lambda')^{r-r'}}{\sum_{r'=0}^{r} (\lambda')^{r-r'}} \sum_{m=0}^{M-1} [\nabla f_k(w_{p,k}^{r',m}, x_{p,k}^{r',m}) - \overline{\nabla f_k}(w_{p,k}^{r',m}, x_{p,k}^{r',m})]\|_2^2 \\ &\leq \eta^2 (\sum_{r'=0}^{r} (\lambda')^{r-r'}) \sum_{r'=0}^{r} (\lambda')^{r-r'} \mathbb{E}\| \sum_{m=0}^{M-1} [\nabla f_k(w_{p,k}^{r',m}, x_{p,k}^{r',m}) - \overline{\nabla f_k}(w_{p,k}^{r',m}, x_{p,k}^{r',m})]\|_2^2 \\ &\leq M\eta^2 (\sum_{r'=0}^{r} (\lambda')^{r-r'}) \sum_{r'=0}^{r} (\lambda')^{r-r'} \sum_{m=0}^{M-1} \mathbb{E}\|\nabla f_k(w_{p,k}^{r',m}, x_{p,k}^{r',m}) - \overline{\nabla f_k}(w_{p,k}^{r',m}, x_{p,k}^{r',m})\|_2^2 \\ &\leq 2M^2(G^2 + \sigma^2)\eta^2 (\sum_{r'=0}^{r} (\lambda')^{r-r'})^2 \\ &\leq 2M^2(G^2 + \sigma^2)\eta^2 [\frac{1 - (\lambda')^{r+1}}{1 - \lambda'}]^2 \end{aligned} \tag{C–5}$$

where $\mathbb{E}\|\nabla f_k(w_{p,k}^{r',m}, x_{p,k}^{r',m})\|_2^2 \leq 2(G^2 + \sigma^2)$ based on Assumptions 3.3.3 and 3.3.2. Then

$$\begin{aligned} &\mathbb{E}\|(w_{p,k}^{r+1,0} - w_{p,k}^{r,M})\|_2^2 \\ &\leq \frac{[1 - (\lambda')^{r+1}]^2 (1-\lambda)^2 K^2}{(1-\lambda')^2 (K-1)^2} 2M^2(G^2 + \sigma^2)\eta^2 \\ &= [1 - (\lambda')^{r+1}]^2 2M^2(G^2 + \sigma^2)\eta^2 \end{aligned} \tag{C–6}$$

### C.3.2 Local Objective

Here we consider the local objective function to optimize. From $(r, 0)$ to $(r, M)$, i.e. $m \in \{0, 1, \cdots, M-1\}$, due to the Lipschitz smooth assumption we have

$$\begin{aligned}
&f_k(w_{k,p}^{r,m+1}) - f_k(w_{k,p}^{r,m}) \\
&\leq \langle \nabla f_k(w_{k,p}^{r,m}), w_{k,p}^{r,m+1} - w_{k,p}^{r,m} \rangle + \frac{L}{2}\|w_{k,p}^{r,m+1} - w_{k,p}^{r,m}\|_2^2 \\
&= -\eta \langle \nabla f_k(w_{k,p}^{r,m}), \nabla f_k(w_{k,p}^{r,m}, x_{k,p}^{r,m}) \rangle + \frac{\eta^2 L}{2}\|\nabla f_k(w_{k,p}^{r,m}, x_{k,p}^{r,m})\|_2^2 \\
&= -\eta \langle \nabla f_k(w_{k,p}^{r,m}), \nabla f_k(w_{k,p}^{r,m}, x_{k,p}^{r,m}) \rangle + \frac{\eta^2 L}{2}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + \frac{\eta^2 L \sigma^2}{2}
\end{aligned} \tag{C–7}$$

Take the expectation and suppose $\eta \leq \frac{1}{L}$,

$$\begin{aligned}
&\mathbb{E}[f_k(w_{k,p}^{r,m+1}) - f_k(w_{k,p}^{r,m})] \\
&\leq -\eta(1 - \frac{\eta L}{2})\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + \frac{\eta^2 L \sigma^2}{2} \\
&\leq -\frac{\eta}{2}\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + \frac{\eta^2 L \sigma^2}{2}
\end{aligned} \tag{C–8}$$

$$\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 \leq \frac{2}{\eta}\mathbb{E}[f_k(w_{k,p}^{r,m}) - f_k(w_{k,p}^{r,m+1})] + \eta L \sigma^2 \tag{C–9}$$

$$\sum_{m=0}^{M-1} \mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 \leq \frac{2}{\eta}\mathbb{E}[f_k(w_{k,p}^{r,0}) - f_k(w_{k,p}^{r,M})] + M\eta L \sigma^2 \tag{C–10}$$

While from $(r, M)$ to $(r+1, 0)$, we have

$$\begin{aligned}
&f_k(w_{k,p}^{r+1,0}) - f_k(w_{k,p}^{r,M}) \\
&\leq \langle \nabla f_k(w_{k,p}^{r,M}), w_{k,p}^{r+1,0} - w_{k,p}^{r,M} \rangle + \frac{L}{2}\|w_{k,p}^{r+1,0} - w_{k,p}^{r,M}\|_2^2 \\
&\leq \frac{\eta}{8}\|\nabla f_k(w_{k,p}^{r,M})\|_2^2 + (\frac{2}{\eta} + \frac{L}{2})\|w_{k,p}^{r+1,0} - w_{k,p}^{r,M}\|_2^2 \\
&\leq \frac{\eta}{4}\|\nabla f_k(w_{k,p}^{r,M-1})\|_2^2 + \frac{\eta L^2}{4}\|w_{k,p}^{r,M} - w_{k,p}^{r,M-1}\|_2^2 + (\frac{2}{\eta} + \frac{L}{2})\|w_{k,p}^{r+1,0} - w_{k,p}^{r,M}\|_2^2 \\
&= \frac{\eta}{4}\|\nabla f_k(w_{k,p}^{r,M-1})\|_2^2 + \frac{\eta^3 L^2}{4}\|\nabla f_k(w_{k,p}^{r,M-1}, x_{k,p}^{r,M-1})\|_2^2 + (\frac{2}{\eta} + \frac{L}{2})\|w_{k,p}^{r+1,0} - w_{k,p}^{r,M}\|_2^2
\end{aligned} \tag{C–11}$$

Therefore, from $(r,0)$ to $(r+1,0)$, we have

$$\begin{aligned}
&\sum_{m=0}^{M-1}\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 \\
&\leq \frac{2}{\eta}\mathbb{E}[f_k(w_{k,p}^{r,0})-f_k(w_{k,p}^{r+1,0})]+M\eta L\sigma^2+\frac{2}{\eta}\mathbb{E}[f_k(w_{k,p}^{r+1,0})-f_k(w_{k,p}^{r,M})] \\
&\leq \frac{2}{\eta}\mathbb{E}[f_k(w_{k,p}^{r,0})-f_k(w_{k,p}^{r+1,0})]+M\eta L\sigma^2+\frac{1}{2}\mathbb{E}\|\nabla f_k(w_{k,p}^{r,M-1})\|^2+\eta^2L^2(G^2+\sigma^2) \\
&\quad+(\frac{4}{\eta^2}+\frac{L}{\eta})\mathbb{E}\|w_{k,p}^{r+1,0}-w_{k,p}^{r,M}\|_2^2
\end{aligned} \tag{C–12}$$

$$\begin{aligned}
&\sum_{m=0}^{M-1}\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 \\
&\leq \frac{4}{\eta}\mathbb{E}[f_k(w_{k,p}^{r,0})-f_k(w_{k,p}^{r+1,0})]+2M\eta L\sigma^2+2\eta^2L^2(G^2+\sigma^2)+(\frac{8}{\eta^2}+\frac{2L}{\eta})\mathbb{E}\|w_{k,p}^{r+1,0}-w_{k,p}^{r,M}\|_2^2
\end{aligned} \tag{C–13}$$

From $r=0$ to $R-1$,

$$\begin{aligned}
&\frac{1}{RM}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 \\
&\leq \frac{4\mathbb{E}[f_k(w_{k,p}^{0,0})-f_k(w_{k,p}^{R,0})]}{\eta RM}+2\eta L\sigma^2+\frac{2\eta^2L^2(G^2+\sigma^2)}{M} \\
&\quad+\frac{1}{RM}(\frac{8}{\eta^2}+\frac{2L}{\eta})\sum_{r=0}^{R-1}\mathbb{E}\|w_{k,p}^{r+1,0}-w_{k,p}^{r,M}\|_2^2 \\
&= \frac{4\mathbb{E}[f_k(w_{k,p}^{0,0})-f_k(w_{k,p}^{R,0})]}{\eta RM}+2\eta L\sigma^2+\frac{2\eta^2L^2(G^2+\sigma^2)}{M} \\
&\quad+\frac{1}{RM}(\frac{8}{\eta^2}+\frac{2L}{\eta})\frac{(1-\lambda)^2K^2}{(K-1)^2}\sum_{r=0}^{R-1}\mathbb{E}\|w_{k,p}^{r,M}-\overline{w}_{k,p}^{r,M}\|_2^2
\end{aligned} \tag{C–14}$$

### C.3.3 Proposed Objective

Here we consider our proposed personalized FL objective function to optimize. For simplicity of notation, let

$$u_k^{r,m}=\frac{1}{\lambda}w_{p,k}^{r,m}-\frac{1-\lambda}{\lambda}\frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K}w_{p,k'}^{r,m} \tag{C–15}$$

Then

$$
\begin{aligned}
u_k^{r,m} - w_{p,k}^{r,m} &= \frac{1-\lambda}{\lambda}(w_{p,k}^{r,m} - \frac{1}{K-1}\sum_{k'=1,k'\neq k}^{K} w_{p,k'}^{r,m}) \\
&= \frac{1-\lambda}{\lambda}\frac{K}{K-1}(w_{p,k}^{r,m} - \overline{w}_{p,k}^{r,m})
\end{aligned}
\tag{C–16}
$$

Now we bound the gradient of the proposed objective.

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\nabla_{w_{p,k}^{r,m}}\sum_{k'=1}^{K} f_{k'}(u_{k'}^{r,m})\|_2^2 \\
&= \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\|\frac{1}{\lambda}\nabla f_k(u_k^{r,m}) - \frac{1-\lambda}{\lambda}\frac{1}{K-1}\nabla f_{k'}(u_{k'}^{r,m})\|_2^2 \\
&\leq \frac{2}{K}\sum_{k=1}^{K}(\frac{1}{\lambda^2} + \frac{(1-\lambda)^2}{\lambda^2(K-1)})\mathbb{E}\|\nabla f_k(u_k^{r,m})\|_2^2 \\
&= \frac{2}{K}\sum_{k=1}^{K}(\frac{1}{\lambda^2} + \frac{(1-\lambda)^2}{\lambda^2(K-1)})[\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + L^2\mathbb{E}\|u_k^{r,m} - w_{k,p}^{r,m}\|_2^2] \\
&= (\frac{1}{\lambda^2} + \frac{(1-\lambda)^2}{\lambda^2(K-1)})\frac{2}{K}\sum_{k=1}^{K}[\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + \frac{L^2(1-\lambda)^2K^2}{\lambda^2(K-1)^2}\mathbb{E}\|w_{k,p}^{r,m} - \overline{w}_{k,p}^{r,m}\|_2^2]
\end{aligned}
\tag{C–17}
$$

$$
\begin{aligned}
&\frac{1}{KRM}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\sum_{k=1}^{K}\mathbb{E}\|\nabla_{w_{p,k}^{r,m}}\sum_{k'=1}^{K} f_{k'}(u_{k'}^{r,m})\|_2^2 \\
&\leq (\frac{1}{\lambda^2} + \frac{(1-\lambda)^2}{\lambda^2(K-1)})\frac{2}{KRM}\sum_{k=1}^{K}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}[\mathbb{E}\|\nabla f_k(w_{k,p}^{r,m})\|_2^2 + \frac{L^2(1-\lambda)^2K^2}{(K-1)^2}\mathbb{E}\|w_{k,p}^{r,m} - \overline{w}_{k,p}^{r,m}\|_2^2] \\
&\leq 2(\frac{1}{\lambda^2} + \frac{(1-\lambda)^2}{\lambda^2(K-1)})[\frac{\frac{4}{K}\sum_{k=1}^{K}(f_k^0 - f_k^*)}{\eta RM} + 2\eta L\sigma^2 + \frac{2\eta^2L^2(G^2+\sigma^2)}{M} \\
&\quad + \frac{1}{KRM}(\frac{8}{\eta^2} + \frac{2L}{\eta})\frac{(1-\lambda)^2K^2}{(K-1)^2}\sum_{k=1}^{K}\sum_{r=0}^{R-1}\mathbb{E}\|w_{k,p}^{r,M} - \overline{w}_{k,p}^{r,M}\|_2^2 \\
&\quad + \frac{1}{KRM}\frac{L^2(1-\lambda)^2K^2}{\lambda^2(K-1)^2}\sum_{k=1}^{K}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\mathbb{E}\|w_{k,p}^{r,m} - \overline{w}_{k,p}^{r,m}\|_2^2]
\end{aligned}
\tag{C–18}
$$

which converges to

$$\begin{aligned}
&\mathcal{O}(\frac{1}{\eta RM\lambda^2} + \frac{(1-\lambda)^2}{KRM\eta^2\lambda^2}\sum_{K=1}^{K}\sum_{r=0}^{R-1}\mathbb{E}\|w_{k,p}^{r,M} - \overline{w}_{k,p}^{r,M}\|_2^2 \\
&\quad + \frac{(1-\lambda)^2}{KRM\lambda^4}\sum_{k=1}^{K}\sum_{r=0}^{R-1}\sum_{m=0}^{M-1}\mathbb{E}\|w_{k,p}^{r,m} - \overline{w}_{k,p}^{r,m}\|_2^2) \\
&= \mathcal{O}(\frac{1}{\eta RM\lambda^2} + \frac{M\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^2} + \frac{M^2\eta^2\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^4})
\end{aligned} \tag{C–19}$$

Suppose $\eta = \mathcal{O}(\frac{1}{\sqrt{RM}})$ and $M = \mathcal{O}(R^{\frac{1}{3}})$, the convergence rate is $\mathcal{O}(\frac{1}{\lambda^4\sqrt{RM}})$ with an error $\mathcal{O}(\frac{M\sum_{r=0}^{R-1}(1-\lambda)^2}{R\lambda^2})$.

## C.4 Additional Experimental Results

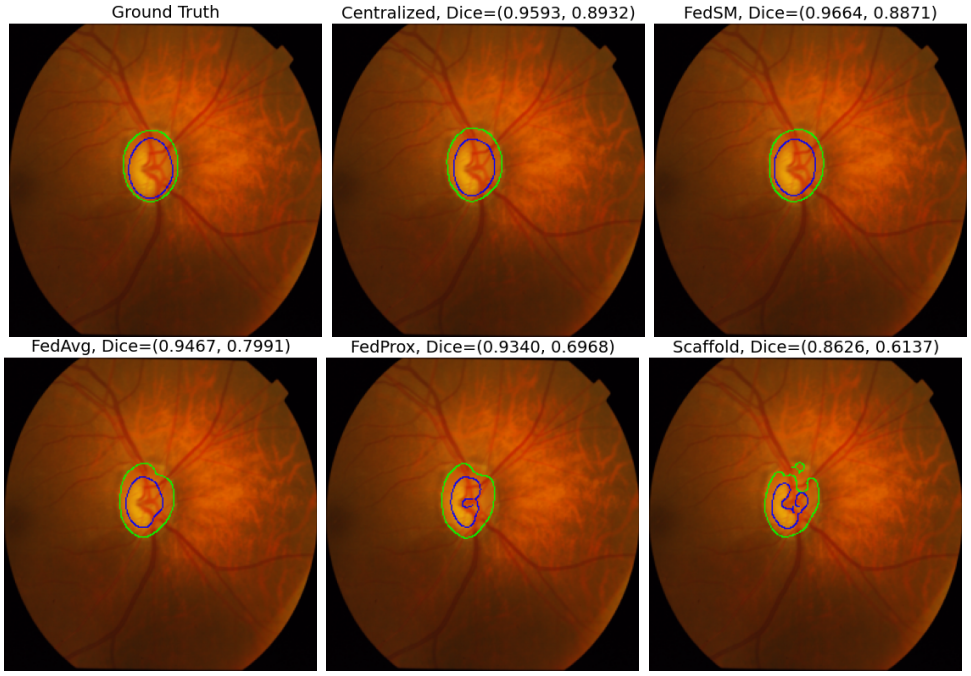


Figure 18: Visual comparison of retinal disc (green) and cup (blue) segmentation. Dice denotes the retinal disc and cup Dice coefficient.

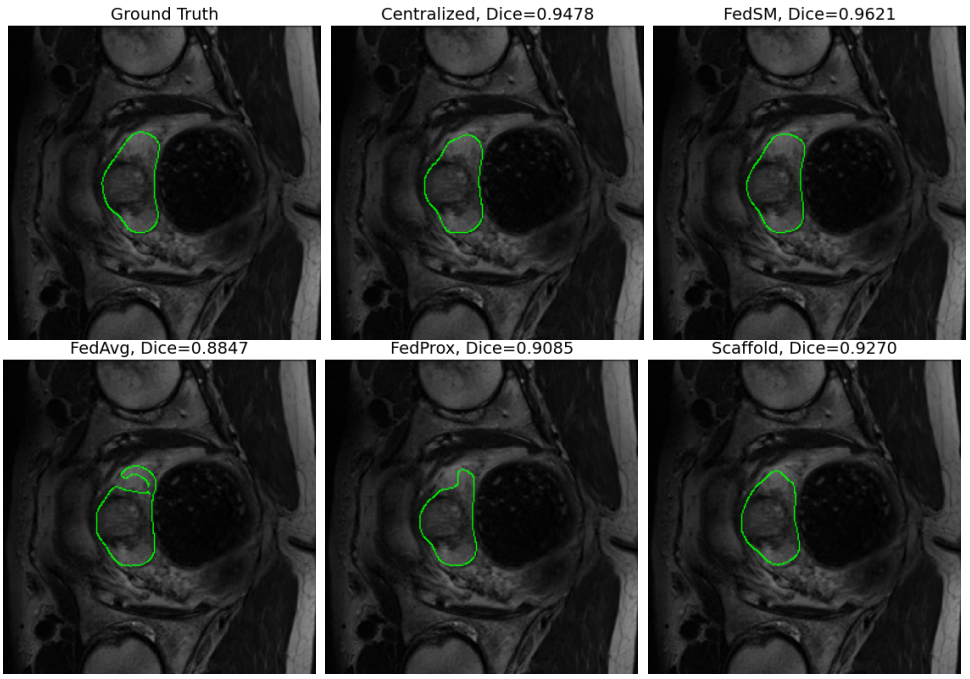


Figure 19: Visual comparison of prostate (green) segmentation. Dice denotes the Dice coefficient.

Table 20: Test Dice coefficient comparison of retinal disc segmentation.

| Method | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Client Avg Dice | Global Dice |
|---|---|---|---|---|---|---|---|---|
| Centralized | 0.9628 | 0.9486 | 0.9489 | 0.9539 | 0.9242 | 0.9565 | 0.9492 | 0.9522 |
| Client 1 Local | 0.9454 | 0.4357 | 0.8956 | 0.6073 | 0.4464 | 0.8409 | 0.6952 | 0.7129 |
| Client 2 Local | 0.2936 | 0.9431 | 0.1099 | 0.8371 | 0.2439 | 0.4575 | 0.4809 | 0.5589 |
| Client 3 Local | 0.9420 | 0.3998 | 0.9468 | 0.6399 | 0.4349 | 0.8256 | 0.6982 | 0.7120 |
| Client 4 Local | 0.6830 | 0.9400 | 0.4805 | 0.9526 | 0.3088 | 0.7803 | 0.6909 | 0.7796 |
| Client 5 Local | 0.6102 | 0.2169 | 0.4601 | 0.2518 | 0.9033 | 0.7064 | 0.5248 | 0.5373 |
| Client 6 Local | 0.8806 | 0.7937 | 0.8354 | 0.8475 | 0.4413 | 0.9547 | 0.7922 | 0.8555 |
| FedAvg | 0.9554 | 0.9410 | 0.9372 | 0.9535 | 0.8653 | 0.9549 | 0.9346 | 0.9444 |
| FedProx | 0.9447 | 0.9343 | 0.9229 | 0.9469 | 0.7573 | 0.9480 | 0.9090 | 0.9283 |
| Scaffold | 0.9207 | 0.9297 | 0.9026 | 0.9474 | 0.6347 | 0.9528 | 0.8813 | 0.9170 |
| FedSM | **0.9653** | **0.9489** | **0.9545** | **0.9551** | **0.9241** | **0.9560** | **0.9507** | **0.9527** |

Table 21: Test Dice coefficient comparison of retinal cup segmentation.

| Method | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Client Avg Dice | Global Dice |
|---|---|---|---|---|---|---|---|---|
| Centralized | 0.8649 | 0.8033 | 0.8027 | 0.8507 | 0.7778 | 0.8793 | 0.8298 | 0.8507 |
| Client 1 Local | 0.8216 | 0.2306 | 0.5733 | 0.3793 | 0.2351 | 0.5621 | 0.4670 | 0.4675 |
| Client 2 Local | 0.1756 | 0.7810 | 0.0673 | 0.7184 | 0.1143 | 0.3637 | 0.3701 | 0.4511 |
| Client 3 Local | 0.7256 | 0.2807 | 0.8064 | 0.5621 | 0.2939 | 0.7333 | 0.5670 | 0.6068 |
| Client 4 Local | 0.3385 | 0.7749 | 0.2109 | 0.8491 | 0.1632 | 0.5842 | 0.4868 | 0.6024 |
| Client 5 Local | 0.4380 | 0.1000 | 0.3305 | 0.1560 | 0.7414 | 0.5380 | 0.3840 | 0.3952 |
| Client 6 Local | 0.7011 | 0.5360 | 0.6296 | 0.6886 | 0.3073 | 0.8752 | 0.6230 | 0.7198 |
| FedAvg | 0.8140 | 0.7949 | 0.7963 | 0.8495 | 0.7101 | 0.8795 | 0.8074 | 0.8402 |
| FedProx | 0.7822 | 0.7702 | 0.7864 | 0.8437 | 0.6132 | 0.8712 | 0.7778 | 0.8216 |
| Scaffold | 0.7554 | 0.7729 | 0.7405 | 0.8396 | 0.4995 | 0.8732 | 0.7469 | 0.8081 |
| FedSM | **0.8610** | **0.8049** | **0.8186** | **0.8530** | **0.7724** | **0.8830** | **0.8322** | **0.8529** |

Table 22: (retinal segmentation, Dice = average of disc and cup Dice coefficients) Model selection frequency from the model selector when FL train with clients $\{1, 2, \cdots, 6\}/\{k\}$ and test on the **unseen** client $k \in \{1, 2, \cdots, 6\}$. From left to right, GM denotes the global model and PM denotes the personalized model $\{1, 2, \cdots, 6\}/\{k\}$. We choose the best $\gamma$.

| Unseen Client $k$ | GM | PM1 | PM2 | PM3 | PM4 | PM5 | PM6 | Best $\gamma$, Dice |
|---|---|---|---|---|---|---|---|---|
| Client $k = 6$ | 1.00 | 0 | 0 | 0 | 0 | 0 | N/A | 1, 0.8906 |
| Client $k = 5$ | 0.69 | 0.18 | 0 | 0 | 0.10 | N/A | 0.03 | 0.9, 0.4304 |
| Client $k = 4$ | 0.03 | 0 | 0.97 | 0 | N/A | 0 | 0 | <0.95, 0.8870 |
| Client $k = 3$ | 0 | 0 | 0.57 | N/A | 0 | 0 | 0.43 | <0.9, 0.8446 |
| Client $k = 2$ | 0 | 0 | N/A | 0 | 0.92 | 0.08 | 0 | <1, 0.8409 |
| Client $k = 1$ | 0 | N/A | 0 | 1.00 | 0 | 0 | 0 | <0.99, 0.8839 |

Table 23: (retinal segmentation, Dice = average of disc and cup Dice coefficients) Dice performance when FL train with clients $\{1, 2, \cdots, 6\}/\{k\}$ and test on the **unseen** client $k \in \{1, 2, \cdots, 6\}$.

| Method/Unseen | Client 1 | Client 2 | Client 3 | Client 4 | Client 5 | Client 6 | Avg |
|---|---|---|---|---|---|---|---|
| Centralized | 0.8842 | 0.8454 | 0.8214 | 0.8866 | 0.4064 | 0.8811 | 0.7875 |
| FedAvg | 0.8598 | 0.8313 | 0.8224 | 0.8551 | 0.4064 | **0.8887** | 0.7773 |
| FedProx | 0.8380 | 0.7856 | 0.8267 | 0.8746 | 0.4171 | 0.8784 | 0.7701 |
| Scaffold | 0.8085 | 0.7998 | 0.8211 | 0.8568 | 0.4121 | 0.8708 | 0.7615 |
| FedSM | **0.8818** | 0.8619 | **0.8498** | **0.8901** | 0.4118 | 0.8646 | 0.7933 |
| FedSM-extra | 0.8747 | **0.8685** | 0.8467 | 0.8794 | **0.4265** | 0.8809 | **0.7963** |

**Algorithm 6** FedSM-extra training.

**Input:** local dataset $\mathcal{D}_k$, rounds $R$, #sites $K$, learning rate $\eta$, coefficient $\lambda$, weight $\frac{n_k}{n}$.
**Initialize:** global model $w_g$, personalized model $w_{p,k}$, model selector $w_s$, optimizer OPT.
**for** round $r = 1, 2, \cdots, R$ **do**
  SERVER: send models ($w_g$, $w_{p,k}$) to client $k$.
  **for** CLIENT $k \in \{1, 2, \cdots, K\}$ in parallel **do**
    initialize $w_{g,k} \leftarrow w_g$
    **for** batch $(x, y) \in \mathcal{D}_n$ **do**
      $w_{g,k} \leftarrow \text{OPT}(w_{g,k}, \eta, \nabla_{w_{g,k}} L(f(w_{g,k}; x), y))$
      $w_{p,k} \leftarrow \text{OPT}(w_{p,k}, \eta, \nabla_{w_{p,k}} L(f(w_{p,k}; x), y))$
    **end for**
    send ($w_{g,k}$, $w_{p,k}$) to server
  **end for**
  SERVER: $w_g \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{g,k}$ *// FedAvg*
  SERVER: $\forall k \in \{1, 2, \cdots, K\}$, $w_{p,k} \leftarrow \lambda w_{p,k} + (1 - \lambda) \frac{1}{K-1} \sum_{k'=1, k' \neq k}^{K} w_{p,k'}$ *// SoftPull*
**end for**
*// extra training rounds*
SERVER: send models ($w_g$, $w_{p,n}$) to clients.
**for** round $r = 1, 2, \cdots, \Delta R$ **do**
  SERVER: send model $w_s$ to clients.
  **for** CLIENT $k \in \{1, 2, \cdots, K\}$ in parallel **do**
    Initialize $w_{s,k} \leftarrow w_s$
    **for** batch $(x, y) \in \mathcal{D}_n$ **do**
      $w_{s,k} \leftarrow \text{OPT}(w_{s,k}, \eta_s, \nabla_{w_{s,k}} L_s(f_s(w_{s,k}; x), y_s))$ *// $y_s$ from Eq. (3–1)*
    **end for**
    send $w_{s,k}$ to server
  **end for**
  SERVER: $w_s \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{s,k}$
**end for**
**Output:** model ($w_g$, $\{w_{p,k}\}_{k=1}^{K}$, $w_s$)

## Appendix D "A New Optimizer with Data Privacy"

### D.1 Task Settings

**CIFAR-10.** We train VGG-16 and ResNet-56 models on CIFAR-10 image classification task. For VGG-16, there is no batch normalization layer. For ResNet-56, we replace the batch normalization layer with the group normalization layer because *non-i.i.d.* data distribution causes inaccurate batch statistics estimation and worsens the client drift issue. The number of groups in group normalization is set to 8. The local batch size $b = 32$ and the total batch size $B = Kb = 512$. The weight decay is $5 \times 10^{-4}$. The model is trained for 200 epochs with a learning rate decay of 0.1 at epoch 120 and 160. Random cropping, random flipping, and standardization are applied as data augmentation techniques.

**SVHN.** We train ResNet-20 on SVHN dataset. The number of groups in group normalization is set to 8. This task is simpler than CIFAR-10 and we set $E = 5$, $s = 2\%$ to enlarge the difference of different methods with a smaller model ResNet-20. The local batch size $b = 32$ and the total batch size $B = Kb = 512$. The model is trained for 200 epochs with a learning rate decay of 0.1 at epoch 120 and 160. The weight decay is $1 \times 10^{-4}$. We do not apply data augmentation in this task.

**CIFAR-100.** We train VGG-16 and ResNet-56 on CIFAR-100 dataset. The number of groups in group normalization is set to 8. This task is harder than CIFAR-10 and we set $E = 0.4$, $s = 40\%$ to make all methods converge. The local batch size $b = 32$ and the total batch size $B = Kb = 512$. The model is trained for 200 epochs with a learning rate decay of 0.1 at epoch 120 and 160. The weight decay is $5 \times 10^{-4}$. Random cropping, random flipping, and standardization are applied as data augmentation techniques.

### D.2 Proof of Theorem 1

The update of server momentum $\mathbf{m}_r$ and model $\mathbf{x}_r$ follows

$$\mathbf{m}_{r+1} = \mu_s \mathbf{m}_r + \frac{1}{KP} \sum_{k=0}^{K-1} \sum_{p=0}^{P-1} \mathbf{m}_{r,p+1}^{(k)} \quad \text{and} \quad \mathbf{x}_{r+1} = \mathbf{x}_r - \alpha\eta P \mathbf{m}_{r+1} \,. \tag{D–1}$$

The update of local momentum $\mathbf{m}_{r,p}^{(k)}$ follows momentum SGD excepts that it will be averaged or reset to zero in the end of each training round. To facilitate the analysis involving both the Polyak's server momentum and local momentum for the first time, we propose to define a sequence $\{\mathbf{y}_r\}$ as

$$\begin{aligned} \mathbf{y}_r &= \mathbf{x}_r + \frac{\mu_s}{1-\mu_s}(\mathbf{x}_r - \mathbf{x}_{r-1}) = \mathbf{x}_r - \frac{\mu_s}{1-\mu_s}\alpha\eta P \mathbf{m}_r \\ &= \frac{1}{1-\mu_s}\mathbf{x}_r - \frac{\mu_s}{1-\mu_s}\mathbf{x}_{r-1} \ (r \geq 1) \quad \text{and} \quad \mathbf{y}_0 = \mathbf{x}_0 \,. \end{aligned} \tag{D–2}$$

We can set $\mathbf{x}_{-1} = \mathbf{x}_0$ to remove the condition in the bracket. It is easy to see that

$$\begin{aligned} \mathbf{y}_{r+1} - \mathbf{y}_r &= \frac{1}{1-\mu_s}(\mathbf{x}_{r+1} - \mathbf{x}_r) - \frac{\mu_s}{1-\mu_s}(\mathbf{x}_r - \mathbf{x}_{r-1}) = -\frac{\alpha}{1-\mu_s}\eta P(\mathbf{m}_{r+1} - \mu_s \mathbf{m}_r) \\ &= -\frac{\alpha\eta}{(1-\mu_s)K} \sum_{k=0}^{K-1} \sum_{p=0}^{P-1} \mathbf{m}_{r,p+1}^{(k)} \,. \end{aligned} \tag{D–3}$$

We also let

$$\widehat{\mathbf{y}}_{r,p} = \mathbf{y}_r - \frac{\alpha\eta}{(1-\mu_s)K} \sum_{k=0}^{K-1} \sum_{p'=0}^{p-1} \mathbf{m}_{r,p'+1}^{(k)} \quad \text{such that} \quad \widehat{\mathbf{y}}_{r,0} = \mathbf{y}_r \quad \text{and} \quad \widehat{\mathbf{y}}_{r,P} = \mathbf{y}_{r+1} = \widehat{\mathbf{y}}_{r+1,0} \,. \tag{D–4}$$

It is easy to see that

$$\widehat{\mathbf{y}}_{r,p+1} - \widehat{\mathbf{y}}_{r,p} = -\frac{\alpha\eta}{(1-\mu_s)K} \sum_{k=0}^{K-1} \mathbf{m}_{r,p+1}^{(k)} \,. \tag{D–5}$$

To facilitate the analysis involving local momentum in $\widehat{\mathbf{y}}_{r,p}$, we define another novel sequence $\{\mathbf{z}_{r,p}\}$ as

$$\begin{aligned} \mathbf{z}_{r,p} &= \widehat{\mathbf{y}}_{r,p} + \frac{\mu_l}{1-\mu_l}(\widehat{\mathbf{y}}_{r,p} - \widehat{\mathbf{y}}_{r,p-1}) = \widehat{\mathbf{y}}_{r,p} - \frac{\mu_l \alpha\eta}{(1-\mu_l)(1-\mu_s)K} \sum_{k=0}^{K-1} \mathbf{m}_{r,p}^{(k)} \\ &= \frac{1}{1-\mu_l}\widehat{\mathbf{y}}_{r,p} - \frac{\mu_l}{1-\mu_l}\widehat{\mathbf{y}}_{r,p-1} \ (r \geq 1 \text{ or } p \geq 1) \quad \text{and} \quad \mathbf{z}_{0,0} = \widehat{\mathbf{y}}_{0,0} = \mathbf{y}_0 = \mathbf{x}_0 \,. \end{aligned} \tag{D–6}$$

We can set $\widehat{\mathbf{y}}_{0,-1} = \widehat{\mathbf{y}}_{0,0}$ to remove the condition in the bracket. Then the update of $\mathbf{z}_{r,p}$ becomes

$$\begin{aligned}\mathbf{z}_{r,p+1} - \mathbf{z}_{r,p} &= \frac{1}{1-\mu_l}(\widehat{\mathbf{y}}_{r,p+1} - \widehat{\mathbf{y}}_{r,p}) - \frac{\mu_l}{1-\mu_l}(\widehat{\mathbf{y}}_{r,p} - \widehat{\mathbf{y}}_{r,p-1}) \\ &= -\frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}(\mathbf{m}_{r,p+1}^{(k)} - \mu_l \mathbf{m}_{r,p}^{(k)})\,.\end{aligned} \tag{D–7}$$

If the local momentum is reset (*i.e.*, $\mathbf{m}_{r,0}^{(k)} \leftarrow \mathbf{0}$) instead of being averaged, *i.e.*,

$$\mathbf{m}_{r,0}^{(k)} \leftarrow \frac{1}{K}\sum_{k=0}^{K-1}\mathbf{m}_{r-1,P}^{(k)} \tag{D–8}$$

at the end of each training round, we need to separately consider this update rule when $p = -1$ for the last equality in the above equation. Therefore, we consider **two cases**:

**(a)** average momentum;

**(b)** reset momentum. In particular, for this case we have either **(b.1)** $p \neq -1$ or **(b.2)** $p = -1$.

For $0 \leq p \leq P-1$ (cases (a) and (b.1)), we have

$$\mathbf{z}_{r,p+1} - \mathbf{z}_{r,p} = -\frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})\,. \tag{D–9}$$

We also consider $p = -1$ (case (b.2)) for completeness as $-1$ has been used in some previous definitions:

$$\begin{aligned}&\mathbf{z}_{r,p+1} - \mathbf{z}_{r,p} = \mathbf{z}_{r,0} - \mathbf{z}_{r,-1} = \mathbf{z}_{r-1,P} - \mathbf{z}_{r-1,P-1} \\ &= -\frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}(\mathbf{m}_{r-1,P}^{(k)} - \mu_l \mathbf{m}_{r-1,P-1}^{(k)}) \\ &= -\frac{\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\nabla F^{(k)}(\mathbf{x}_{r-1,P-1}^{(k)}, \xi_{r-1,P-1}^{(k)})\,.\end{aligned} \tag{D–10}$$

In this way the critical property $\mathbf{z}_{r,P} = \mathbf{z}_{r+1,0}$ is preserved from $\{\widehat{\mathbf{y}}_{r,p}\}$. In contrast, the analysis of the update between $\overline{\mathbf{x}}_{r+1,0}$ and $\overline{\mathbf{x}}_{r,P-1}$ is more tricky. Now we analyze the convergence of $\{\mathbf{z}_{r,p}\}$ and compare its difference with $\{\overline{\mathbf{x}}_{r,p}\}$.

### D.2.1 Inconsistency Bound of $\|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2$ (Lemma 2)

In this section, some procedures in the proof of case (a) and case (b) can be different. Let's first consider case (a).

$$\begin{aligned}
\mathbf{z}_{r,p} &= \widehat{\mathbf{y}}_{r,p} + \frac{\mu_l}{1-\mu_l}(\widehat{\mathbf{y}}_{r,p} - \widehat{\mathbf{y}}_{r,p-1}) = \widehat{\mathbf{y}}_{r,p} - \frac{\mu_l \alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\mathbf{m}_{r,p}^{(k)} \\
&= \mathbf{y}_r - \frac{\alpha\eta}{(1-\mu_s)K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)} - \frac{\mu_l\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\mathbf{m}_{r,p}^{(k)} \\
&= \mathbf{x}_r - \frac{\mu_s}{1-\mu_s}\alpha\eta P\mathbf{m}_r - \frac{\alpha\eta}{(1-\mu_s)K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)} - \frac{\mu_l\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\mathbf{m}_{r,p}^{(k)}\,.
\end{aligned} \tag{D–11}$$

For DOMO algorithm, we have

$$\overline{\mathbf{x}}_{r,p} = \frac{1}{K}\sum_{k=0}^{K-1}\mathbf{x}_{r,p}^{(k)} = \mathbf{x}_r - \beta\eta P\mathbf{m}_r - \frac{\eta}{K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)}\,. \tag{D–12}$$

Let $\beta = \frac{\mu_s}{1-\mu_s}\alpha$ and we have,

$$\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p} = (1 - \frac{\alpha}{1-\mu_s})\frac{\eta}{K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)} - \frac{\mu_l\alpha\eta}{(1-\mu_l)(1-\mu_s)K}\sum_{k=0}^{K-1}\mathbf{m}_{r,p}^{(k)}\,. \tag{D–13}$$

We define $t = rP + p$, and $\nabla F_t^{(k)} = \nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})$. Then for case (a) we have

$$\begin{aligned}
&\|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 \\
&= \|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)} - \frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\mathbf{m}_{r,p}^{(k)}]\|_2^2 \\
&= \|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\sum_{\tau=0}^{t-p+p'}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)} - \frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\sum_{\tau=0}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}]\|_2^2 \\
&= \|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\sum_{\tau=0}^{t-p}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)} - \frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\sum_{\tau=0}^{t-p}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)} \\
&\quad + (1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\sum_{\tau=t-p+1}^{t-p+p'}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)} - \frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\sum_{\tau=t-p+1}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}\|_2^2
\end{aligned} \tag{D–14}$$

$$
\begin{aligned}
&\|\mathbf{z}_{r,p}-\overline{\mathbf{x}}_{r,p}\|_2^2\\
&=\|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\sum_{\tau=0}^{t-p}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)}-\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\sum_{\tau=0}^{t-p}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}\\
&\quad+(1-\frac{\alpha}{1-\mu_s})\sum_{\tau=t-p+1}^{t-1}\sum_{p'=\tau-t+p}^{p-1}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)}-\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}\sum_{\tau=t-p+1}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}\|_2^2\\
&=\|\frac{\eta}{K}\sum_{k=0}^{K-1}[\sum_{\tau=0}^{t-p}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}((1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\mu_l^{-p+p'+1}-\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s})\\
&\quad+\sum_{\tau=t-p+1}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}((1-\frac{\alpha}{1-\mu_s})\sum_{p'=\tau-t+p}^{p-1}\mu_l^{-p+p'+1}-\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s})]\|_2^2\,.
\end{aligned}
\tag{D–15}
$$

For simplicity, let

$$
\begin{aligned}
h_1&=\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}-(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\mu_l^{-p+p'+1}=\frac{\alpha}{1-\mu_s}\frac{1+\mu_l-\mu_l^p}{1-\mu_l}-\frac{1-\mu_l^p}{1-\mu_l}\\
&\le h\coloneqq\frac{\alpha}{1-\mu_s}\frac{1+\mu_l-\mu_l^p}{1-\mu_l}-1
\end{aligned}
\tag{D–16}
$$

$$
h_2=\frac{\mu_l}{1-\mu_l}\frac{\alpha}{1-\mu_s}-(1-\frac{\alpha}{1-\mu_s})\sum_{p'=\tau-t+p}^{p-1}\mu_l^{-p+p'+1}=\frac{\alpha}{1-\mu_s}\frac{1+\mu_l-\mu_l^{t-\tau}}{1-\mu_l}-\frac{1-\mu_l^{t-\tau}}{1-\mu_l}
\tag{D–17}
$$

When $t-p+1\le\tau\le t-1$, i.e., $1\le t-\tau\le p-1<p$, we have $h_2<h$. Suppose

$$
\alpha\ge(1-\mu_s)(1-\mu_l)\ge\max\{\frac{(1-\mu_s)(1-\mu_l^p)}{1+\mu_l-\mu_l^p},\frac{(1-\mu_s)(1-\mu_l^{t-\tau})}{1+\mu_l-\mu_l^{t-\tau}}\}
\tag{D–18}
$$

Then it is easy to see that $h_1,h_2\ge 0$.

$$
\begin{aligned}
&\|\mathbf{z}_{r,p}-\overline{\mathbf{x}}_{r,p}\|_2^2\le\|\frac{\eta}{K}\sum_{k=0}^{K-1}(\sum_{\tau=0}^{t-p}h_1\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}+\sum_{\tau=t-p+1}^{t-1}h_2\mu_l^{t-1-\tau}\nabla F_\tau^{(k)})\|_2^2\\
&\le\eta^2(\sum_{\tau=0}^{t-p}h_1\mu_l^{t-1-\tau}+\sum_{\tau=t-p+1}^{t-1}h_2\mu_l^{t-1-\tau})\\
&\quad\cdot(\sum_{\tau=0}^{t-p}h_1\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2+\sum_{\tau=t-p+1}^{t-1}h_2\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2)\\
&\le\eta^2(\sum_{\tau=0}^{t-1}h\mu_l^{t-1-\tau})\sum_{\tau=0}^{t-1}h\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2\le\frac{\eta^2h^2}{1-\mu_l}\sum_{\tau=0}^{t-1}\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2\,.
\end{aligned}
\tag{D–19}
$$

Let $T = RP$. Note that $h$ is a function of $\alpha$ and $p$ (or $t$). Summing from $t = 0$ to $T - 1$ yields

$$\begin{aligned}
\sum_{t=0}^{T-1} \|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 &\leq \frac{\eta^2}{1-\mu_l} \sum_{t=0}^{T-1} h^2 \sum_{\tau=0}^{t-1} \mu_l^{t-1-\tau} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \\
&= \frac{\eta^2}{1-\mu_l} \sum_{\tau=0}^{T-2} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \sum_{t=\tau+1}^{T-1} h^2 \mu_l^{t-1-\tau} \\
&\leq \frac{\eta^2}{1-\mu_l} \sum_{\tau=0}^{T-2} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \sum_{t=0}^{+\infty} h^2 \mu_l^{t} \\
&\leq \frac{\eta^2}{1-\mu_l} \sum_{\tau=0}^{T-2} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \sum_{p=0}^{P-1} h^2 \sum_{n=0}^{+\infty} \mu_l^{p+nP} \\
&= \frac{\eta^2}{1-\mu_l} \sum_{\tau=0}^{T-2} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \sum_{p=0}^{P-1} h^2 \frac{\mu_l^p}{1-\mu_l^P} \,.
\end{aligned} \tag{D–20}$$

The R.H.S. is minimized when $\alpha = (1-\mu_s)(1-\mu_l)$, $h1- = \mu_l - \mu_l^p$ and we have

$$\begin{aligned}
\sum_{t=0}^{T-1} \|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 &\leq \frac{\eta^2}{1-\mu_l} \sum_{\tau=0}^{T-2} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_\tau^{(k)} \|_2^2 \sum_{p=0}^{P-1} \mu_l^2 \frac{\mu_l^p}{1-\mu_l^P} \\
&= \frac{\eta^2 \mu_l^2}{(1-\mu_l)^2} \sum_{t=0}^{T-1} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_t^{(k)} \|_2^2 \,.
\end{aligned} \tag{D–21}$$

In particular, for local momentum SGD, $\alpha = 1$, $\mu_s = 0$, and $h_1 = h_2 = \frac{\mu_l}{1-\mu_l}$. Moreover, for DOMO with $\alpha = 1 - \mu_s$ and $\beta = \mu_s$, we still have $h_1 = h_2 = \frac{\mu_l}{1-\mu_l}$. For both of them, we can get the inconsistency bound following the above precedure by replacing $h$ with $\frac{\mu_l}{1-\mu_l}$:

$$\sum_{t=0}^{T-1} \|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 \leq \frac{\eta^2 \mu_l^2}{(1-\mu_l)^4} \sum_{t=0}^{T-1} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F_t^{(k)} \|_2^2 \,. \tag{D–22}$$

Compare the above two upper bounds and we can see that DOMO achieves $(1-\mu_l)^2$ of the inconsistency bound in local momentum SGD. However, we note that a potential higher improvement may be achieved as $\alpha = (1-\mu_s)(1-\mu_l)$ may not be optimal due to inequality scaling. Therefore in experiments hyper-parameters $\alpha$ and $\beta$ require further tuning.

As the improvement is a constant factor and does not affect the convergence rate (though it helps empirical training), in later analysis we simply set $\alpha = 1$ and $\beta = \mu_s$ to preserve the same inconsistency bound.

Following similar procedures except that the local momentum is reset to $\mathbf{0}$ every $P$ iterations, for case (b) we have

$$\begin{aligned}
&\|\mathbf{z}_{r,p}-\overline{\mathbf{x}}_{r,p}\|_2^2=\|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\mathbf{m}_{r,p'+1}^{(k)}-\frac{\mu_l}{1-\mu_l}\mathbf{m}_{r,p}^{(k)}]\|_2^2\\
&=\|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{p'=0}^{p-1}\sum_{\tau=t-p}^{t-p+p'}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)}-\frac{\mu_l}{1-\mu_l}\sum_{\tau=t-p}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}]\|_2^2\\
&=\|\frac{\eta}{K}\sum_{k=0}^{K-1}[(1-\frac{\alpha}{1-\mu_s})\sum_{\tau=t-p}^{t-1}\sum_{p'=\tau-t+p}^{p-1}\mu_l^{t-p+p'-\tau}\nabla F_\tau^{(k)}-\frac{\mu_l}{1-\mu_l}\sum_{\tau=t-p}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}]\|_2^2\\
&=\|\frac{\eta}{K}\sum_{k=0}^{K-1}\sum_{\tau=t-p}^{t-1}\mu_l^{t-1-\tau}\nabla F_\tau^{(k)}h_2\|_2^2\le\|\eta h_1\sum_{\tau=t-p}^{t-1}\mu_l^{t-1-\tau}\sum_{k=0}^{K-1}\frac{1}{K}\nabla F_\tau^{(k)}\|_2^2\\
&\le\frac{\eta^2h_1^2}{1-\mu_l}\sum_{\tau=t-p}^{t-1}\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2\le\frac{\eta^2h_1^2}{1-\mu_l}\sum_{\tau=0}^{t-1}\mu_l^{t-1-\tau}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla F_\tau^{(k)}\|_2^2\,.
\end{aligned}\tag{D–23}$$

Compare it with Eq. ( D–19) and we can see that the inconsistency bound in case (a) can also bound that in case (b).

### D.2.2 Divergence Bound of $\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2$

We note that in this section, the proof is identical for either case (a) or case (b). We first consider

$$\begin{aligned}
&\frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})-\frac{1}{K}\sum_{k'=0}^{K-1}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2\\
&\le\frac{1}{K}\sum_{k=0}^{K-1}(3\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})-\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p})\|_2^2+3\|\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p})-\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2\\
&\quad+3\|\nabla f(\overline{\mathbf{x}}_{r,p})-\frac{1}{K}\sum_{k'=0}^{K-1}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2)\\
&\le\frac{6L^2}{K}\sum_{k=0}^{K-1}\|\mathbf{x}_{r,p}^{(k)}-\overline{\mathbf{x}}_{r,p}\|_2^2+3G^2\,.
\end{aligned}\tag{D–24}$$

Then,

$$
\begin{aligned}
&\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2\\
&=\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\frac{1}{K}\sum_{k'=0}^{K-1}(\mathbf{x}_r-\mu_s\eta P\mathbf{m}_r-\sum_{p'=0}^{p-1}\eta\mathbf{m}_{r,p'+1}^{(k')})-(\mathbf{x}_r-\mu_s\eta P\mathbf{m}_r-\sum_{p'=0}^{p-1}\eta\mathbf{m}_{r,p'+1}^{(k)})\|_2^2\\
&=\frac{\eta^2}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\sum_{p'=0}^{p-1}(\frac{1}{K}\sum_{k'=0}^{K-1}\mathbf{m}_{r,p'+1}^{(k')}-\mathbf{m}_{r,p'+1}^{(k)})\|_2^2\\
&=\frac{\eta^2}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\sum_{p'=0}^{p-1}(\frac{1}{K}\sum_{k'=0}^{K-1}\nabla F^{(k')}(\mathbf{x}_{r,p'}^{(k')},\xi_{r,p'}^{(k')})-\nabla F^{(k)}(\mathbf{x}_{r,p'}^{(k)},\xi_{r,p'}^{(k)}))\frac{1-\mu_l^{p-p'}}{1-\mu_l}\|_2^2\\
&\le\frac{2\eta^2}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\sum_{p'=0}^{p-1}[\frac{1}{K}\sum_{k'=0}^{K-1}(\nabla F^{(k')}(\mathbf{x}_{r,p'}^{(k')},\xi_{r,p'}^{(k')})-\nabla f^{(k)}(\mathbf{x}_{r,p'}^{(k')}))\\
&\qquad\qquad-(\nabla F^{(k)}(\mathbf{x}_{r,p'}^{(k)},\xi_{r,p'}^{(k)})-\nabla f^{(k)}(\mathbf{x}_{r,p'}^{(k)}))]\frac{1-\mu_l^{p-p'}}{1-\mu_l}\|_2^2\\
&\quad+\frac{2\eta^2}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\sum_{p'=0}^{p-1}[\frac{1}{K}\sum_{k'=0}^{K-1}\nabla f^{(k')}(\mathbf{x}_{r,p'}^{(k')})-\nabla f^{(k)}(\mathbf{x}_{r,p'}^{(k)})]\frac{1-\mu_l^{p-p'}}{1-\mu_l}\|_2^2\\
&\le\frac{2\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{2\eta^2P}{(1-\mu_l)^2K}\sum_{k=0}^{K-1}\sum_{p'=0}^{p-1}\mathbb{E}\|\frac{1}{K}\sum_{k'=0}^{K-1}\nabla f^{(k')}(\mathbf{x}_{r,p'}^{(k')})-\nabla f^{(k)}(\mathbf{x}_{r,p'}^{(k)})\|_2^2\\
&\le\frac{2\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{2\eta^2P}{(1-\mu_l)^2}\sum_{p'=0}^{p-1}(\frac{6L^2}{K}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p'}-\mathbf{x}_{r,p'}^{(k)}\|_2^2+3G^2)\\
&\le\frac{12\eta^2PL^2}{(1-\mu_l)^2K}\sum_{p'=0}^{p-1}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p'}-\mathbf{x}_{r,p'}^{(k)}\|_2^2+\frac{2\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{6\eta^2P^2G^2}{(1-\mu_l)^2}
\end{aligned}
\tag{D–25}
$$

Sum $t$ from 0 to $T-1=RP-1$ (i.e., sum $p$ from 0 to $P-1$ and sum $r$ from 0 to $R-1$) and let $P\le\frac{1-\mu_l}{6\eta L}$,

$$
\begin{aligned}
&\frac{1}{KT}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2\\
&\le\frac{12\eta^2P^2L^2}{(1-\mu_l)^2KT}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2+\frac{2\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{6\eta^2P^2G^2}{(1-\mu_l)^2}\\
&\le\frac{1}{3KT}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2+\frac{2\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{6\eta^2P^2G^2}{(1-\mu_l)^2}\frac{3\eta^2P\sigma^2}{(1-\mu_l)^2}+\frac{9\eta^2P^2G^2}{(1-\mu_l)^2}\,.
\end{aligned}
\tag{D–26}
$$

### D.2.3 Main Proof

Consider the improvement in one training round ($0 \leq p \leq P-1$). By the smoothness assumption,

$$\begin{aligned}
&\mathbb{E}_{r,p} f(\mathbf{z}_{r,p+1}) - \mathbb{E}_{r,p} f(\mathbf{z}_{r,p}) \leq \mathbb{E}_{r,p} \langle \nabla f(\mathbf{z}_{r,p}), \mathbf{z}_{r,p+1} - \mathbf{z}_{r,p} \rangle + \frac{L}{2} \mathbb{E}_{r,p} \|\mathbf{z}_{r,p+1} - \mathbf{z}_{r,p}\|_2^2 \\
&= -\frac{\eta}{1-\mu_l} \langle \nabla f(\mathbf{z}_{r,p}), \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \rangle + \frac{L\eta^2}{2(1-\mu_l)^2} \mathbb{E}_{r,p} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla F^{(k)}(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)}) \|_2^2 \\
&= -\frac{\eta}{1-\mu_l} \langle \nabla f(\mathbf{z}_{r,p}), \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \rangle + \frac{L\eta^2}{2(1-\mu_l)^2} \mathbb{E}_{r,p} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \|_2^2 \\
&\quad + \frac{L\eta^2 \sigma^2}{2(1-\mu_l)^2 K} \,.
\end{aligned} \tag{D–27}$$

$\forall \gamma \in \mathbb{R}^+$, the first term

$$\begin{aligned}
&-\frac{\eta}{1-\mu_l} \langle \nabla f(\mathbf{z}_{r,p}), \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \rangle \\
&= -\frac{\eta}{1-\mu_l} \langle \nabla f(\mathbf{z}_{r,p}) - \nabla f(\overline{\mathbf{x}}_{r,p}), \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \rangle \\
&\quad - \frac{\eta}{1-\mu_l} \langle \nabla f(\overline{\mathbf{x}}_{r,p}), \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \rangle \\
&\leq \frac{\eta}{2(1-\mu_l)\gamma} \|\nabla f(\mathbf{z}_{r,p}) - \nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta\gamma}{2(1-\mu_l)} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \|_2^2 \\
&\quad - \frac{\eta}{2(1-\mu_l)} (\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \|_2^2 - \|\nabla f(\overline{\mathbf{x}}_{r,p}) - \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \|_2^2) \\
&\leq \frac{\eta L^2}{2(1-\mu_l)\gamma} \|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 - \frac{\eta(1-\gamma)}{2(1-\mu_l)} \| \frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) \|_2^2 - \frac{\eta}{2(1-\mu_l)} \|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 \\
&\quad + \frac{\eta L^2}{2(1-\mu_l) K} \sum_{k=0}^{K-1} \|\overline{\mathbf{x}}_{r,p} - \mathbf{x}_{r,p}^{(k)}\|_2^2 \,.
\end{aligned} \tag{D–28}$$

Combine the above two equations,

$$\begin{aligned}
&\mathbb{E}_{r,p} f(\mathbf{z}_{r,p+1}) - \mathbb{E}_{r,p} f(\mathbf{z}_{r,p}) \\
&\leq -\frac{\eta}{2(1-\mu_l)} \|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta L^2}{2(1-\mu_l)\gamma} \|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 + \frac{\eta L^2}{2(1-\mu_l)K} \sum_{k=0}^{K-1} \|\overline{\mathbf{x}}_{r,p} - \mathbf{x}_{r,p}^{(k)}\|_2^2 \\
&\quad + \frac{L\eta^2\sigma^2}{2(1-\mu_l)^2 K} - \frac{\eta}{2(1-\mu_l)}(1-\gamma-L\eta) \|\frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2 .
\end{aligned} \tag{D–29}$$

Now we take the total expectation and sum $t$ from 0 to $T-1 = RP-1$,

$$\begin{aligned}
\frac{1}{T}[\mathbb{E} f(\mathbf{z}_{R-1,P}) - f(\mathbf{z}_{0,0})] \leq &-\frac{\eta}{2(1-\mu_l)T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta L^2}{2\gamma(1-\mu_l)T} \sum_{t=0}^{T-1} \mathbb{E}\|\mathbf{z}_{r,p} - \overline{\mathbf{x}}_{r,p}\|_2^2 \\
&+ \frac{\eta L^2}{2(1-\mu_l)TK} \sum_{t=0}^{T-1} \sum_{k=0}^{K-1} \mathbb{E}\|\overline{\mathbf{x}}_{r,p} - \mathbf{x}_{r,p}^{(k)}\|_2^2 + \frac{L\eta^2\sigma^2}{2(1-\mu_l)^2 K} \\
&- \frac{\eta}{2(1-\mu_l)T}(1-\gamma-L\eta) \sum_{t=0}^{T-1} \mathbb{E}\|\frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2 .
\end{aligned} \tag{D–30}$$

Based on bounds derived in section D.2.1 and D.2.2, let $\gamma = \frac{1}{2}$ and $1-\gamma-L\eta-\frac{\mu_l^2\eta^2L^2}{\gamma(1-\mu_l)^4} \geq 0$,

$$\begin{aligned}
&\frac{1}{T}[f_* - f(\mathbf{x}_0)] \leq \frac{1}{T}[\mathbb{E} f(\mathbf{z}_{R-1,P}) - f(\mathbf{z}_{0,0})] \\
&\leq -\frac{\eta}{2(1-\mu_l)T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta L^2}{2\gamma(1-\mu_l)T} \frac{\mu_l^2\eta^2}{(1-\mu_l)^4} \sum_{t=0}^{T-1} \|\frac{1}{K} \sum_{k=0}^{K-1} \nabla F(\mathbf{x}_{r,p}^{(k)}, \xi_{r,p}^{(k)})\|_2^2 \\
&\quad + \frac{\eta L^2}{2(1-\mu_l)}[\frac{3\eta^2 P\sigma^2}{(1-\mu_l)^2} + \frac{9\eta^2P^2G^2}{(1-\mu_l)^2}] \\
&\quad + \frac{L\eta^2\sigma^2}{2(1-\mu_l)^2K} - \frac{\eta}{2(1-\mu_l)T}(1-\gamma-L\eta) \sum_{t=0}^{T-1} \mathbb{E}\|\frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2 \\
&= -\frac{\eta}{2(1-\mu_l)T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta^2L\sigma^2}{2(1-\mu_l)^2}(\frac{1}{K} + \frac{3\eta LP}{2(1-\mu_l)} + \frac{\mu_l^2\eta L}{\gamma(1-\mu_l)^4K}) \\
&\quad + \frac{9\eta^3L^2P^2G^2}{2(1-\mu_l)^3} - \frac{\eta}{2(1-\mu_l)T}(1-\gamma-L\eta-\frac{\mu_l^2\eta^2L^2}{\gamma(1-\mu_l)^4}) \sum_{t=0}^{T-1} \mathbb{E}\|\frac{1}{K} \sum_{k=0}^{K-1} \nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2 \\
&\leq \frac{-\eta}{2(1-\mu_l)T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 + \frac{\eta^2L\sigma^2}{2(1-\mu_l)^2}(\frac{1}{K} + \frac{3\eta LP}{2(1-\mu_l)} + \frac{2\mu_l^2\eta L}{(1-\mu_l)^4K}) + \frac{9\eta^3L^2P^2G^2}{}.
\end{aligned} \tag{D–31}$$

Rearrange and we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2 \leq \frac{2(1-\mu_l)(f(\mathbf{x}_0)-f_*)}{\eta T} + \frac{\eta L\sigma^2}{(1-\mu_l)}(\frac{1}{K} + \frac{3\eta LP}{2(1-\mu_l)} + \frac{2\mu_l^2\eta L}{(1-\mu_l)^4 K}) + \frac{9\eta^2L^2P^2G^2}{(1-\mu_l)^2}\,, \tag{D–32}$$

which completes the proof.

### D.3 Extension to Partial Participation

In this section we show that it is possible to extend the theoretical analysis of DOMO to cross-device FL with partial clients participation in each training round.

For the algorithm side, suppose we randomly sample a client set $\mathcal{V}_r$ to participate in training round $r$. Let $|\mathcal{V}_r| = S$. For the computation of client $k$, we can just replace client $k \in [K]$ (full participation) with $k \in \mathcal{V}\_r$ (partial participation). Correspondingly, the server should only communicate with clients in $\mathcal{V}_r$.

That is, $k \in [K] \to k \in \mathcal{V}_r$ and $\frac{1}{K}\sum_{k=0}^{K-1} \to \frac{1}{S}\sum_{k\in\mathcal{V}_r}$.

For the convergence analysis, we should also do such replacement. But besides the replacement, we note that now $\nabla f(\mathbf{x}) = \frac{1}{K}\sum_{k=0}^{K-1}\nabla f^{(k)}(\mathbf{x}) \neq \frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k)}(\mathbf{x})$, which will affect the following two inequalities of the proof in the previous section.

(a) Eq. ( D–24):

$$\frac{1}{K}\sum_{k=0}^{K-1}\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) - \frac{1}{K}\sum_{k'=0}^{K-1}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2 \leq \frac{6L^2}{K}\sum_{k=0}^{K-1}\|\mathbf{x}_{r,p}^{(k)} - \overline{\mathbf{x}}_{r,p}\|_2^2 + 3G^2 \tag{D–33}$$

New (partial participation):

$$\begin{aligned}
&\frac{1}{S}\sum_{k\in\mathcal{V}_r}\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) - \frac{1}{S}\sum_{k'\in\mathcal{V}_r}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2 \leq \frac{1}{S}\sum_{k\in\mathcal{V}_r}\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) - \frac{1}{S}\sum_{k'\in\mathcal{V}_r}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2 \\
&\leq \frac{1}{S}\sum_{k\in\mathcal{V}_r}[3\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)}) - \nabla f^{(k)}(\overline{\mathbf{x}}_{r,p})\|_2^2 + 3\|\frac{1}{S}\sum_{k'\in\mathcal{V}_r}(\nabla f^{(k')}(\overline{\mathbf{x}}_{r,p}) - \nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')}))\|_2^2 \\
&+ 3\|\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p}) - \frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k')}(\overline{\mathbf{x}}_{r,p})\|_2^2]
\end{aligned} \tag{D–34}$$

$$
\begin{aligned}
&\frac{1}{S}\sum_{k\in\mathcal{V}_r}\|\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})-\frac{1}{S}\sum_{k'\in\mathcal{V}_r}\nabla f^{(k')}(\mathbf{x}_{r,p}^{(k')})\|_2^2\\
&\le\frac{6L^2}{S}\sum_{k\in\mathcal{V}_r}\|\mathbf{x}_{r,p}^{(k)}-\overline{\mathbf{x}}_{r,p}\|_2^2\\
&\quad+\frac{1}{S}\sum_{k\in\mathcal{V}_r}[6\|\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p})-\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2+6\|\nabla f(\overline{\mathbf{x}}_{r,p})-\frac{1}{S}\sum_{k'\in\mathcal{V}_r}\nabla f^{(k')}(\overline{\mathbf{x}}_{r,p})\|_2^2]\\
&\le\frac{6L^2}{S}\sum_{k\in\mathcal{V}_r}\|\mathbf{x}_{r,p}^{(k)}-\overline{\mathbf{x}}_{r,p}\|_2^2+12G^2
\end{aligned}
\tag{D–35}
$$

This larger constant coefficient of $G^2$ does not affect the convergence rate. This leads to a new bound in section D.2.2

$$
\frac{1}{KT}\sum_{t=0}^{T-1}\sum_{k=0}^{K-1}\mathbb{E}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2\le\frac{3\eta^2P^2\sigma^2}{(1-\mu_l)^2}+\frac{36\eta^2P^2G^2}{(1-\mu_l)^2}
\tag{D–36}
$$

(b) In Eq. ( D–28), we showed that

$$
\begin{aligned}
&-\frac{\eta}{1-\mu_l}\langle\nabla f(\overline{\mathbf{x}}_{r,p}),\frac{1}{K}\sum_{k=0}^{K-1}\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\rangle\\
&\le-\frac{\eta}{2(1-\mu_l)}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2-\frac{\eta}{2(1-\mu_l)}\|\frac{1}{K}\sum_{k=0}^{K-1}\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2\\
&\quad+\frac{\eta L^2}{2(1-\mu_l)K}\sum_{k=0}^{K-1}\|\overline{\mathbf{x}}_{r,p}-\mathbf{x}_{r,p}^{(k)}\|_2^2
\end{aligned}
\tag{D–37}
$$

New (partial participation): suppose the client set $\mathcal{V}_r$ is randomly and uniformly sampled following common practice, such that $\mathbb{E}_{\mathcal{V}_r}[\frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k)}(\mathbf{x})]=\nabla f(\mathbf{x})$. Then

$$
\begin{aligned}
&-\frac{\eta}{1-\mu_l}\mathbb{E}_{\mathcal{V}_r}\langle\nabla f(\overline{\mathbf{x}}_{r,p}),\frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\rangle\\
&=-\frac{\eta}{1-\mu_l}\mathbb{E}_{\mathcal{V}_r}\langle\nabla f(\overline{\mathbf{x}}_{r,p}),\frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p})\rangle\\
&\quad-\frac{\eta}{1-\mu_l}\mathbb{E}_{\mathcal{V}_r}\langle\nabla f(\overline{\mathbf{x}}_{r,p}),\frac{1}{S}\sum_{k\in\mathcal{V}_r}(\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})-\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p}))\\
&\le-\frac{\eta}{1-\mu_l}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2+\frac{\eta}{2(1-\mu_l)}\mathbb{E}_{\mathcal{V}_r}[\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2+\|\frac{1}{S}\sum_{k\in\mathcal{V}_r}(\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})-\nabla f^{(k)}(\overline{\mathbf{x}}_{r,p}))\|_2^2]\\
&\le-\frac{\eta}{2(1-\mu_l)}\|\nabla f(\overline{\mathbf{x}}_{r,p})\|_2^2+\mathbb{E}_{\mathcal{V}_r}[\frac{\eta L^2}{2(1-\mu_l)S}\sum_{k\in\mathcal{V}_r}\|\mathbf{x}_{r,p}^{(k)}-\overline{\mathbf{x}}_{r,p}\|_2^2]
\end{aligned}
\tag{D–38}
$$

Compared with the previous result, this bound is larger by the term

$$\frac{\eta}{2(1-\mu_l)}\|\frac{1}{S}\sum_{k\in\mathcal{V}_r}\nabla f^{(k)}(\mathbf{x}_{r,p}^{(k)})\|_2^2\,, \tag{D–39}$$

which we will need to bound in the main proof. A simple way to bound it will be an additional assumption to bound the gradient norm. In summary, not much of the current proof in the previous section needs to be modified. For partial participation, we will need a a stronger assumption that $\|\nabla f^{(k)}(\mathbf{x})\|_2^2 \leq M^2$. Following the modification above and the main proof, we will have a convergence rate $\mathcal{O}(S^{\frac{1}{2}}R^{-\frac{1}{2}}P^{-\frac{1}{2}})$.

## Bibliography


[1] Nci_isbi dataset. `https://www.cancerimagingarchive.net/`.

[2] Promise12 dataset. `https://promise12.grand-challenge.org/`.

[3] Prostatex dataset. `https://prostatex.grand-challenge.org/`.

[4] Refuge dataset. `https://refuge.grand-challenge.org/details/`.

[5] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI' 16)*, pages 265–283, 2016.

[6] Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *International Conference on Learning Representations*, 2020.

[7] Alham Fikri Aji and Kenneth Heafield. Sparse communication for distributed gradient descent. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 440–445, 2017.

[8] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30:1709–1720, 2017.

[9] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, pages 5973–5983, 2018.

[10] Ahmed Almazroa, Sami Alodhayb, Essameldin Osman, Eslam Ramadan, Mohammed Hummadi, Mohammed Dlaim, Muhannad Alkatee, Kaamran Raahemifar, and Vasudevan Lakshminarayanan. Retinal fundus images for glaucoma analysis: the riga dataset. In *Medical Imaging 2018: Imaging Informatics for Healthcare, Research, and Applications*, volume 10579, page 105790B. International Society for Optics and Photonics, 2018.

[11] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, Bram van Ginneken, et al. The medical segmentation decathlon. *arXiv preprint arXiv:2106.05735*, 2021.

[12] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning*, pages 344–353. PMLR, 2019.

[13] Dmitrii Avdiukhin and Grigory Yaroslavtsev. Escaping saddle points with compressed sgd. *arXiv preprint arXiv:2105.10090*, 2021.

[14] Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. *Advances in Neural Information Processing Systems*, 32, 2019.

[15] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of cnns. *arXiv preprint arXiv:1901.08164*, 2019.

[16] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[17] Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, and Anima Anandkumar. signsgd with majority vote is communication efficient and fault tolerant. In *International Conference on Learning Representations*, 2018.

[18] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konečnỳ, Stefano Mazzocchi, H Brendan McMahan, et al. Towards federated learning at scale: System design. *arXiv preprint arXiv:1902.01046*, 2019.

[19] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

[20] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

[21] Kai Chen and Qiang Huo. 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)*, pages 5880–5884. IEEE, 2016.

[22] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015.

[23] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016.

[24] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, pages 15236–15245, 2019.

[25] Jeffrey Dean, Greg S Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Quoc V Le, Mark Z Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, et al. Large scale distributed deep networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems-Volume 1*, pages 1223–1231, 2012.

[26] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27:1646–1654, 2014.

[27] Aaron Defazio and Léon Bottou. On the ineffectiveness of variance reduced optimization for deep learning. *arXiv preprint arXiv:1812.04529*, 2018.

[28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[29] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

[30] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[31] Aritra Dutta, El Houcine Bergou, Ahmed M Abdelmoniem, Chen-Yu Ho, Atal Narayan Sahu, Marco Canini, and Panos Kalnis. On the discrepancy between the theoretical analysis and practical implementations of compressed communication for distributed deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3817–3824, 2020.

[32] Melih Elibol, Lihua Lei, and Michael I Jordan. Variance reduction with sparse gradients. In *International Conference on Learning Representations*, 2019.

[33] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in Neural Information Processing Systems*, 33, 2020.

[34] Ilyas Fatkhullin, Igor Sokolov, Eduard Gorbunov, Zhize Li, and Peter Richtárik. Ef21 with bells & whistles: Practical algorithmic extensions of modern error feedback. *arXiv preprint arXiv:2110.03294*, 2021.

[35] Francisco Fumero, Silvia Alayón, José L Sanchez, Jose Sigut, and M Gonzalez-Hernandez. Rim-one: An open retinal image database for optic nerve evaluation. In *2011 24th international symposium on computer-based medical systems (CBMS)*, pages 1–6. IEEE, 2011.

[36] Hongchang Gao, An Xu, and Heng Huang. On the convergence of communication-efficient local sgd for federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence, Virtual*, pages 18–19, 2021.

[37] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting gradients–how easy is it to break privacy in federated learning? *arXiv preprint arXiv:2003.14053*, 2020.

[38] Jemin George and Prudhvi Gurram. Distributed stochastic gradient descent with event-triggered communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7169–7178, 2020.

[39] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[40] Andreas Griewank. An implementation of checkpointing for the reverse or adjoint model of differentiation. *ACM Trans. Math. Software*, 26(1):1–19, 1999.

[41] Bin Gu, An Xu, Zhouyuan Huo, Cheng Deng, and Heng Huang. Privacy-preserving asynchronous vertical federated learning algorithms for multiparty collaborative learning. *IEEE transactions on neural networks and learning systems*, 33(11):6103–6115, 2021.

[42] Pengfei Guo, Puyang Wang, Jinyuan Zhou, Shanshan Jiang, and Vishal M Patel. 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*, pages 2423–2432, 2021.

[43] Pengfei Guo, Dong Yang, Ali Hatamizadeh, An Xu, Ziyue Xu, Wenqi Li, Can Zhao, Daguang Xu, Stephanie Harmon, Evrim Turkbey, et al. Auto-fedrl: Federated hyperparameter optimization for multi-institutional medical image segmentation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXI*, pages 437–455. Springer, 2022.

[44] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.

[45] Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. *arXiv preprint arXiv:1902.00744*, 2019.

[46] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[48] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[49] Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B Gibbons, Garth A Gibson, Greg Ganger, and Eric P Xing. More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in neural information processing systems*, pages 1223–1231, 2013.

[50] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pages 4387–4398. PMLR, 2020.

[51] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. *arXiv preprint arXiv:1909.06335*, 2019.

[52] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[53] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017.

[54] Yanping Huang, Yonglong Cheng, Dehao Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *arXiv preprint arXiv:1811.06965*, 2018.

[55] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32:103–112, 2019.

[56] Yuzhen Huang, Xiao Yan, Guanxian Jiang, Tatiana Jin, James Cheng, An Xu, Zhanhao Liu, and Shuo Tu. Tangram: bridging immutable and mutable abstractions for distributed data analytics. In *2019 {USENIX} Annual Technical Conference ({USENIX}{ATC} 19)*, pages 191–206, 2019.

[57] Zhouyuan Huo, Bin Gu, and Heng Huang. Training neural networks using features replay. In *Advances in Neural Information Processing Systems*, pages 6659–6668, 2018.

[58] Zhouyuan Huo, Bin Gu, and Heng Huang. Large batch training does not need warmup. *arXiv preprint arXiv:2002.01576*, 2020.

[59] Zhouyuan Huo, Bin Gu, qian Yang, and Heng Huang. Decoupled parallel backpropagation with convergence guarantee. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80

of *Proceedings of Machine Learning Research*, pages 2098–2106, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[60] Zhouyuan Huo and Heng Huang. Straggler-agnostic and communication-efficient distributed primal-dual algorithm for high-dimensional data mining. *arXiv preprint arXiv:1910.04235*, 2019.

[61] Zhouyuan Huo, Qian Yang, Bin Gu, Lawrence Carin Huang, et al. Faster on-device training using new federated momentum algorithm. *arXiv preprint arXiv:2002.02090*, 2020.

[62] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[63] P Izmailov, AG Wilson, D Podoprikhin, D Vetrov, and T Garipov. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*, pages 876–885, 2018.

[64] Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1627–1635. JMLR. org, 2017.

[65] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[66] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26:315–323, 2013.

[67] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[68] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 795–811. Springer, 2016.

[69] Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank J Reddi, Sebastian U Stich, and Ananda Theertha Suresh. Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*, 2020.

[70] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[71] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, 2019.

[72] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[73] Anastasia Koloskova, Tao Lin, Sebastian U Stich, and Martin Jaggi. Decentralized deep learning with arbitrary communication compression. In *International Conference on Learning Representations*, 2019.

[74] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[75] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[76] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[77] Ilja Kuzborskij and Christoph Lampert. Data-dependent stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 2815–2824. PMLR, 2018.

[78] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[79] Seunghak Lee, Jin Kyu Kim, Xun Zheng, Qirong Ho, Garth A Gibson, and Eric P Xing. On model parallelization and scheduling strategies for distributed machine

learning. In *Advances in neural information processing systems*, pages 2834–2842, 2014.

[80] Guillaume Lemaître, Robert Martí, Jordi Freixenet, Joan C Vilanova, Paul M Walker, and Fabrice Meriaudeau. Computer-aided detection and diagnosis for prostate cancer based on mono and multi-parametric mri: a review. *Computers in biology and medicine*, 60:8–31, 2015.

[81] Jinfeng Li, Xiao Yan, Jian Zhang, An Xu, James Cheng, Jie Liu, Kelvin KW Ng, and Ti-chung Cheng. A general and efficient querying method for learning to hash. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1333–1347, 2018.

[82] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. Scaling distributed machine learning with the parameter server. In *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pages 583–598, 2014.

[83] Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pages 19–27, 2014.

[84] Mu Li, David G Andersen, Alexander J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. *Advances in Neural Information Processing Systems*, 27:19–27, 2014.

[85] Mu Li, Li Zhou, Zichao Yang, Aaron Li, Fei Xia, David G Andersen, and Alexander Smola. Parameter server for distributed machine learning. In *Big Learning NIPS Workshop*, volume 6, page 2, 2013.

[86] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[87] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

[88] Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations*, 2020.

[89] Youjie Li, Mingchao Yu, Songze Li, Salman Avestimehr, Nam Sung Kim, and Alexander Schwing. Pipe-sgd: A decentralized pipelined sgd framework for distributed deep net training. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[90] Zhize Li, Dmitry Kovalev, Xun Qian, and Peter Richtarik. Acceleration for compressed gradient descent in distributed and federated optimization. In *International Conference on Machine Learning*, pages 5895–5904. PMLR, 2020.

[91] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5336–5346, 2017.

[92] Xianfeng Liang, Shuheng Shen, Jingchang Liu, Zhen Pan, Enhong Chen, and Yifei Cheng. Variance reduced local sgd with lower communication complexity. *arXiv preprint arXiv:1912.12844*, 2019.

[93] Yujun Lin, Song Han, Huizi Mao, Yu Wang, and Bill Dally. Deep gradient compression: Reducing the communication bandwidth for distributed training. In *International Conference on Learning Representations*, 2018.

[94] Quande Liu, Cheng Chen, Jing Qin, Qi Dou, and Pheng-Ann Heng. 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*, pages 1013–1023, 2021.

[95] Yuejiang Liu, An Xu, and Zichong Chen. Map-based deep imitation learning for obstacle avoidance. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8644–8649. IEEE, 2018.

[96] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

[97] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[98] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[99] Konstantin Mishchenko, Eduard Gorbunov, Martin Takáč, and Peter Richtárik. Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*, 2019.

[100] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International Conference on Machine Learning*, pages 4615–4625, 2019.

[101] Deepak Narayanan, Aaron Harlap, Amar Phanishayee, Vivek Seshadri, Nikhil R Devanur, Gregory R Ganger, Phillip B Gibbons, and Matei Zaharia. Pipedream: generalized pipeline parallelism for dnn training. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, pages 1–15, 2019.

[102] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[103] Gergely Neu and Lorenzo Rosasco. Iterate averaging as regularization for stochastic gradient descent. In *Conference On Learning Theory*, pages 3222–3242. PMLR, 2018.

[104] Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. In *Advances in neural information processing systems*, pages 1037–1045, 2016.

[105] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8026–8037, 2019.

[106] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[107] Boris T Polyak. New stochastic approximation type procedures. *Automat. i Telemekh*, 7(98-107):2, 1990.

[108] Xun Qian, Peter Richtárik, and Tong Zhang. Error compensated distributed SGD can be accelerated. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[109] Sashank J Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and Hugh Brendan McMahan. Adaptive federated optimization. In *International Conference on Learning Representations*, 2020.

[110] Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. Ef21: A new, simpler, theoretically better, and practically faster error feedback. *arXiv preprint arXiv:2106.05203*, 2021.

[111] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[112] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[113] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[114] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[115] David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

[116] Atal Sahu, Aritra Dutta, Ahmed M Abdelmoniem, Trambak Banerjee, Marco Canini, and Panos Kalnis. Rethinking gradient sparsification as total error minimization. *Advances in Neural Information Processing Systems*, 34, 2021.

[117] Frank Seide and Amit Agarwal. Cntk: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2135–2135, 2016.

[118] Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. *arXiv preprint arXiv:2103.04628*, 2021.

[119] Anil Shanbhag, Holger Pirk, and Samuel Madden. Efficient top-k query processing on massively parallel hardware. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1557–1570, 2018.

[120] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[121] J. Sivaswamy, S. R. Krishnadas, G. Datt Joshi, M. Jain, and A. U. Syed Tabish. Drishti-gs: Retinal image dataset for optic nerve head(onh) segmentation. In *2014 IEEE 11th International Symposium on Biomedical Imaging (ISBI)*, pages 53–56, April 2014.

[122] Liuyihan Song, Kang Zhao, Pan Pan, Yu Liu, Yingya Zhang, Yinghui Xu, and Rong Jin. Communication efficient sgd via gradient sampling with bayes prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12065–12074, 2021.

[123] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[124] Sebastian U Stich. Local sgd converges fast and communicates little. In *International Conference on Learning Representations*, 2018.

[125] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, pages 4447–4458, 2018.

[126] Nikko Strom. Scalable distributed dnn training using commodity gpu cloud computing. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[127] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[128] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[129] Canh T Dinh, Nguyen Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. *Advances in Neural Information Processing Systems*, 33, 2020.

[130] Hanlin Tang, Yao Li, Ji Liu, and Ming Yan. Errorcompensatedx: error compensation for variance reduced algorithms. *Advances in Neural Information Processing Systems*, 34, 2021.

[131] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. Decentralized training over decentralized data. In *International Conference on Machine Learning*, pages 4848–4856. PMLR, 2018.

[132] Hanlin Tang, Chen Yu, Xiangru Lian, Tong Zhang, and Ji Liu. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *International Conference on Machine Learning*, pages 6155–6165. PMLR, 2019.

[133] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[134] Leslie G Valiant. A bridging model for parallel computation. *Communications of the ACM*, 33(8):103–111, 1990.

[135] Thijs Vogels, Sai Praneeth Karinireddy, and Martin Jaggi. Powersgd: Practical low-rank gradient compression for distributed optimization. *Advances In Neural Information Processing Systems 32 (Nips 2019)*, 32(CONF), 2019.

[136] Jianyu Wang and Gauri Joshi. Adaptive communication strategies to achieve the best error-runtime trade-off in local-update sgd. *arXiv preprint arXiv:1810.08313*, 2018.

[137] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. In *Advances in Neural Information Processing Systems*, volume 33, pages 7611–7623, 2020.

[138] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. In *International Conference on Learning Representations*, 2019.

[139] Kangkang Wang, Rajiv Mathews, Chloé Kiddon, Hubert Eichner, Françoise Beaufays, and Daniel Ramage. Federated evaluation of on-device personalization. *arXiv preprint arXiv:1910.10252*, 2019.

[140] Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. In *Advances in Neural Information Processing Systems*, pages 1299–1309, 2018.

[141] Wei Wen, Cong Xu, Feng Yan, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Terngrad: Ternary gradients to reduce communication in distributed deep learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[142] Jingfeng Wu, Vladimir Braverman, and Lin Yang. Obtaining adjustable regularization for free via iterate averaging. In *International Conference on Machine Learning*, pages 10344–10354. PMLR, 2020.

[143] Cong Xie, Shuai Zheng, Oluwasanmi O Koyejo, Indranil Gupta, Mu Li, and Haibin Lin. Cser: Communication-efficient sgd with error reset. *Advances in Neural Information Processing Systems*, 33, 2020.

[144] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[145] An Xu and Heng Huang. Coordinating momenta for cross-silo federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8735–8743, 2022.

[146] An Xu and Heng Huang. Detached error feedback for distributed sgd with random sparsification. In *International Conference on Machine Learning*, pages 24550–24575. PMLR, 2022.

[147] An Xu, Zhouyuan Huo, and Heng Huang. On the acceleration of deep learning model parallelism with staleness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2088–2097, 2020.

[148] An Xu, Zhouyuan Huo, and Heng Huang. Optimal gradient quantization condition for communication-efficient distributed training. *arXiv preprint arXiv:2002.11082*, 2020.

[149] An Xu, Zhouyuan Huo, and Heng Huang. Step-ahead error feedback for distributed training with compressed gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10478–10486, 2021.

[150] Yan Yan, Tianbao Yang, Zhe Li, Qihang Lin, and Yi Yang. A unified analysis of stochastic momentum methods for deep learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 2955–2961. AAAI Press, 2018.

[151] Guandao Yang, Tianyi Zhang, Polina Kirichenko, Junwen Bai, Andrew Gordon Wilson, and Chris De Sa. Swalp: Stochastic weight averaging in low precision training. In *International Conference on Machine Learning*, pages 7015–7024. PMLR, 2019.

[152] Qian Yang, Zhouyuan Huo, Wenlin Wang, and Lawrence Carin. Ouroboros: On accelerating training of transformer-based language models. In *Advances in Neural Information Processing Systems 32*, pages 5519–5529. Curran Associates, Inc., 2019.

[153] Tianbao Yang, Qihang Lin, and Zhe Li. Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization. *arXiv preprint arXiv:1604.03257*, 2016.

[154] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M Alvarez, Jan Kautz, and Pavlo Molchanov. See through gradients: Image batch recovery via gradinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16337–16346, 2021.

[155] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. In *International Conference on Machine Learning*, pages 7184–7193. PMLR, 2019.

[156] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[157] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. In *Advances in neural information processing systems*, pages 9793–9803, 2018.

[158] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in Neural Information Processing Systems*, 32:9597–9608, 2019.

[159] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. idlg: Improved deep leakage from gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[160] Yang Zhao, Chunyuan Li, Ping Yu, Jianfeng Gao, and Changyou Chen. Feature quantization improves gan training. In *International Conference on Machine Learning*, pages 11376–11386. PMLR, 2020.

[161] Shuai Zheng, Ziyue Huang, and James Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. *Advances in Neural Information Processing Systems*, 32:11450–11460, 2019.

[162] Pan Zhou, Hanshu Yan, Xiaotong Yuan, Jiashi Feng, and Shuicheng Yan. Towards understanding why lookahead generalizes better than sgd and beyond. *Advances in Neural Information Processing Systems*, 34, 2021.

[163] Ligeng Zhu, Hongzhou Lin, Yao Lu, Yujun Lin, and Song Han. Delayed gradient averaging: Tolerate the communication latency for federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[164] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in Neural Information Processing Systems*, 32:14774–14784, 2019.