Accelerating Federated Learning via Momentum Gradient Descent

Wei Liu, Li Chen, Yunfei Chen, *Senior Member, IEEE*, and Wenyi Zhang, *Senior Member, IEEE*

Abstract—Federated learning (FL) provides a communication-efficient approach to solve machine learning problems concerning distributed data, without sending raw data to a central server. However, existing works on FL only utilize first-order gradient descent (GD) and do not consider the preceding iterations to gradient update which can potentially accelerate convergence. In this paper, we consider momentum term which relates to the last iteration. The proposed momentum federated learning (MFL) uses momentum gradient descent (MGD) in the local update step of FL system. We establish global convergence properties of MFL and derive an upper bound on MFL convergence rate. Comparing the upper bounds on MFL and FL convergence rate, we provide conditions in which MFL accelerates the convergence. For different machine learning models, the convergence performance of MFL is evaluated based on experiments with MNIST and CIFAR-10 datasets. Simulation results confirm that MFL is globally convergent and further reveal significant convergence improvement over FL.

Index Terms—Accelerating convergence, distributed machine learning, federated learning, momentum gradient descent.

1 Introduction

Recently, data-intensive machine learning has been applied in various fields, such as autonomous driving [1], speech recognition [2], image classification [3] and disease detection [4] since this technique provides beneficial solutions to extract the useful information hidden in data. It now becomes a common tendency that machine-learning systems are deploying in architectures that include ten of thousands of processors [5]. Great amount of data is generated by various parallel and distributed physical objects.

Collecting data from edge devices to the central server is necessary for distributed machine learning scenarios. In the process of distributed data collection, there exist significant challenges such as energy efficiency problems and system latency problems. The energy efficiency of distributed data collection was considered in wireless sensor networks (WSNs) due to limited battery capacity of sensors [6]; In fifth-generation (5G) cellular networks, a round-trip delay from terminals through the network back to terminals demands much lower latencies, potentially down to 1 ms, to facilitate human tactile to visual feedback control [7]. Thus, the challenges of data aggregation in distributed system urgently require communication-efficient solutions.

In order to overcome these challenges, cutting down transmission distance and reducing the amount of uploaded data from edge devices to the network center are two effective ways. To reduce transmission distance, mobile edge computing (MEC) in [8] is an emerging technique where the computation and storage resources are pushed to proximity of edge devices where the local task and data offloaded by users can be processed. In this way, the distance of large-scale data transmission is greatly shortened and the latency has a significant reduction [9]. Using machine learning for the prediction of uploaded task execution time achieves a shorter processing delay [10], and dynamic resource scheduling was studied to optimize resources allocation of MEC system in [11]. To reduce the uploaded data size, model-based compression approaches, where raw data are compressed and represented by well-established model parameters, demonstrate significant compression performance [12]. Lossy compression is also an effective strategy to decrease the uploaded data size [13], [14]. Compressed sensing, where the sparse data of the edge can be efficiently sampled and reconstructed with transmitting a much smaller data size, was applied to data acquisition of Internet of Things (IoT) network [15]. All the aforementioned works need to collect raw data from individual device.

To avoid collecting raw data for machine learning in distributed scenarios, a novel approach named *Federated Learning* (FL) has emerged as a promising solution [16]. The work in [17] provided a fundamental architecture design of FL. Considering the growing computation capability of edge nodes (devices), FL decentralizes the centralized machine learning task and assigns the decomposed computing tasks to the edge nodes where the raw data are stored and learned at the edge nodes. After a fixed iteration interval, each edge node transmits its learned model parameter to the central server. This strategy can substantially decrease consumption of communication resources and improve communication-efficiency. To further improve the energy efficiency of FL, an adaptive FL approach was proposed in [17], where the aggregation frequency can be adjusted adaptively to minimize the loss function under a fixed resource budget. To reduce the uplink communication costs, the work in [18] proposed structured and sketched updates method, and

- *W. Liu, L. Chen and W. Zhang are with Department of Electronic Engineering and Information Science, University of Science and Technology of China. E-mail: liuwei93@mail.ustc.edu.cn, {chenli87, wenyizha}@ustc.edu.cn.*
- *Y. Chen is with the School of Engineering, University of Warwick, Coventry CV4 7AL, U.K. E-mail: Yunfei.Chen@warwick.ac.uk.*

Manuscript received 6 Oct. 2019; revised 14 Jan. 2020; accepted 15 Feb. 2020. (Corresponding author: Li Chen.)
Recommended for acceptance by M. Parashar.

compression techniques were adopted to reduce parameter dimension in this work. In [19], gradient selection and adaptive adjustment of learning rate were used for efficient compression. For security aggregation of high-dimensional data, the work in [20] provided a communication-efficient approach, where the server can compute the sum of model parameters from edge nodes without knowing the contribution of each individual node. In [21], under unbalanced resource distribution in network edge, FL with client (edge node) selection was proposed for actively managing the clients aggregation according to their resources condition. In [22], non-i.i.d data distribution was studied.

However, existing FL solutions generally use gradient descent (GD) for loss function minimization. GD is a one-step method where the next iteration depends only on the current gradient. Convergence rate of GD can be improved by accounting for more preceding iterations [23]. Thus, by introducing the last iteration, which is named momentum term, momentum gradient descent (MGD) can accelerate the convergence [24] [25]. Due to the improved convergence of gradient methods brought by momentum, there are several works which apply stochastic gradient descent (SGD) with momentum in the field of distributed machine learning. In [26], momentum is applied to the update at each aggregation rounds for improving both optimization and generalization. In [27], the linear convergence of distributed SGD with momentum is proven. All these works with momentum are based on stochastic GD generally. Compared with SGD, deterministic gradient descent (DGD) can realize more precise training results with improved generalization and fast convergence under convex optimization [28].

Motivated by the above observations, we propose a new federated learning design of *Momentum Federated Learning* (MFL) in this paper. In the proposed MFL design, we introduce momentum term in FL local update and leverage MGD (in our paper, MGD means DGD with momentum) to perform local iterations. Further, the global convergence of the proposed MFL is proven. We derive the theoretical convergence bound of MFL. Compared with FL [17], the proposed MFL has an accelerated convergence rate under certain conditions. On the basis of MNIST and CIFAR-10 datasets, we numerically study the proposed MFL and obtain its loss function curve. The experiment results show that MFL converges faster than FL for different machine learning models. The contributions of this paper are summarized as follows:

- *MFL design*: According to the characteristic that MGD facilitates machine learning convergence in the centralized situation, we propose MFL design where MGD is adopted to optimize loss function in local update. The proposed MFL can improve the convergence rate of distributed learning problem significantly.
- *Convergence analysis for MFL*: We prove that the proposed MFL is globally convergent on convex optimization problems, and derive its theoretical upper bound on convergence rate. We make a comparative analysis of convergence performance between the proposed MFL and FL. It is proven that MFL improves convergence rate of FL under certain conditions.
- *Evaluation based on MNIST and CIFAR-10 datasets*: We evaluate the proposed MFL's convergence performance via simulation based on MNIST and CIFAR-10 datasets with different machine learning models such as support vector machine (SVM), linear regression, logistic regression and convolutional neural network (CNN). Then an experimental comparison is made between FL and the proposed MFL. The simulation results show that MFL is convergent and confirm that MFL provides a significant improvement of convergence rate.

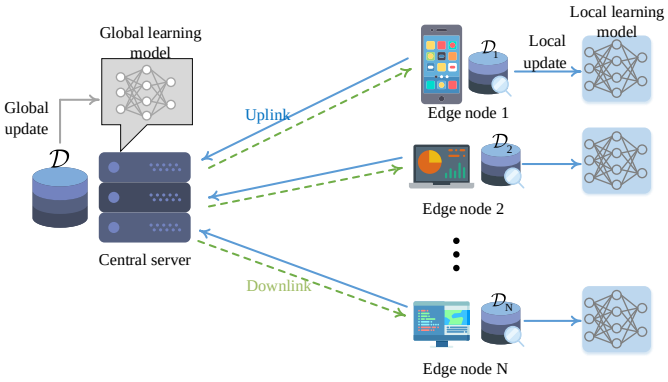


Fig. 1: The simplified structure of learning system for distributed user data

The remaining part of this paper is organized as follows. We introduce the system model to solve the learning problem in distributed scenarios in Section 2 and subsequently elaborate the existing solutions in Section 3. In Section 4, we describe the design of MFL in detail. Then in Section 5 and 6, we present the convergence analysis of MFL and the comparison between FL and MFL, respectively. Finally, we show experimental results in Section 7 and draw a conclusion in Section 8.

2 System Model

In this paper, considering a simplified system model, we discuss the distributed network as shown in Fig. 1. This model has N edge nodes and a central server. These N edge nodes, which have limited communication and computation resources, contain local datasets $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_i, ..., \mathcal{D}_N$, respectively. So the global dataset is $\mathcal{D} \triangleq \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_N$. Assume that $\mathcal{D}_i \cap \mathcal{D}_j = \emptyset$ for $i \neq j$. We define the number of samples in node i as $|\mathcal{D}_i|$ where $|\cdot|$ denotes the size of the set. The total number of all nodes' samples is $|\mathcal{D}|$, and $|\mathcal{D}| = \sum_{i=1}^{N} |\mathcal{D}_i|$. The central server connects all the edge nodes for information transmission.

We define *the global loss function* at the central server as $F(\mathbf{w})$, where $\mathbf{w}$ denotes the model parameter. Different machine learning models correspond to different $F(\cdot)$ and $\mathbf{w}$. We use $\mathbf{w}^*$ to represent the optimal parameter for minimizing the value of $F(\mathbf{w})$. Based on the presented model, the learning problem is to minimize $F(\mathbf{w})$ and it can be formulated as follows:

$$\mathbf{w}^* \triangleq \arg\min F(\mathbf{w}). \tag{1}$$

Because of the complexity of machine learning model and original dataset, finding a closed-form solution of the above optimization problem is usually impossible. So algorithms based on gradient iterations are used to solve (1). If raw user data are collected and stored in the central server, we can use centralized learning solutions to (1) while if raw user data are distributed over the edge nodes, FL and the proposed MFL can be applied to optimize this learning problem.

Under the situation where FL or MFL solutions are used, the local loss function of node i is denoted by $F_i(\mathbf{w})$ which is defined merely on $\mathcal{D}_i$. Then we define the global loss function $F(\mathbf{w})$ on $\mathcal{D}$ as follows:

Definition 1 (Global loss function)**.** *Given the loss function $F_i(\mathbf{w})$ of edge node i, we define the global loss function on all the distributed datasets as*

$$F(\mathbf{w}) \triangleq \frac{\sum_{i=1}^{N} |\mathcal{D}_i| F_i(\mathbf{w})}{|\mathcal{D}|}. \tag{2}$$

3 Existing Solutions

In this section, we introduce two existing solutions to solve the learning problem expressed by (1). These two solutions are centralized learning solution and FL solution, respectively.

3.1 Centralized Learning Solution

Centralized machine learning is for machine learning model embedded in the central server and each edge node needs to send its raw data to the central sever. In this situation, edge nodes will consume communication resources for data transmission, but without incurring computation resources consumption.

After the central server has collected all datasets from the edge nodes, a usual way to solve the learning problem expressed by (1) is GD as a basic gradient method. Further, MGD is an improved gradient method with adding a momentum term to speed up learning process [24].

3.1.1 GD

The update rule for GD is as follows:

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \eta \nabla F(\mathbf{w}(t-1)). \tag{3}$$

In (3), t denotes the iteration index and $\eta > 0$ is the learning step size. The model parameter $\mathbf{w}$ is updated along the direction of negative gradient. Using the above update rule, GD can solve the learning problem with continuous iterations.

3.1.2 MGD

As an improvement of GD, MGD introduces the momentum term and we present its update rules as follows:

$$\mathbf{d}(t) = \gamma \mathbf{d}(t-1) + \nabla F(\mathbf{w}(t-1)) \tag{4}$$

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \eta \mathbf{d}(t), \tag{5}$$

where $\mathbf{d}(t)$ is the momentum term which has the same dimension as $\mathbf{w}(t)$, γ is the momentum attenuation factor, η is the learning step size and t is the iteration index. By iterations of (4) and (5) with t, $F(\mathbf{w})$ can potentially converge to the minimum faster compared with GD. The convergence range of MGD is $-1 < \gamma < 1$ with a bounded η and if $0 < \gamma < 1$, MGD has an accelerated convergence rate than GD under a small η typically used in simulations [29, Result 3].

3.2 FL Solution

In contrast with centralized learning solutions, FL avoids collecting and uploading the distributed data because of the limited communication resources at edge nodes and privacy protection for local data. It decouples the machine learning task from the central server to each edge node to avoid storing user data in the server and reduce the communication consumption. All of edge nodes make up a federation in coordination with the central server.

The FL design and convergence analysis are presented in [17] where FL network is studied thoroughly. In an FL system, each edge node uses the same machine learning model. We use τ to denote the global aggregation frequency, i.e., the update interval. Each node i has its local model parameter $\widetilde{\mathbf{w}}_i(t)$, where the iteration index is denoted by $t = 0, 1, 2, ...$ (in this paper, an iteration means a local update). We use $[k]$ to denote the aggregation interval $[(k-1)\tau, k\tau]$ for $k = 1, 2, 3, ...$ At $t = 0$, local model parameters of all nodes are initialized to the same value. When $t > 0$, $\widetilde{\mathbf{w}}_i(t)$ is updated locally based on GD, which is the *local update*. After τ local updates, *global aggregation* is performed and all edge nodes send the updated model parameters to the centralized server synchronously.

The learning process of FL is described as follows.

3.2.1 Local Update

When $t \in [k]$, local updates are performed in each edge node by

$$\widetilde{\mathbf{w}}_i(t) = \widetilde{\mathbf{w}}_i(t-1) - \eta \nabla F_i(\widetilde{\mathbf{w}}_i(t-1)),$$

which follows GD exactly.

3.2.2 Global Aggregation

When $t = k\tau$, global aggregation is performed. Each node sends $\widetilde{\mathbf{w}}_i(k\tau)$ to the central server synchronously. The central server takes a weighted average of the received parameters from N nodes to obtain the globally updated parameter $\mathbf{w}(k\tau)$ by

$$\mathbf{w}(k\tau) = \frac{\sum_{i=1}^{N} |\mathcal{D}_i| \widetilde{\mathbf{w}}_i(k\tau)}{|\mathcal{D}|}.$$

Then $\mathbf{w}(k\tau)$ is sent back to all edge nodes as their new parameters and edge nodes perform local update for the next iteration interval.

In [17, Lemma 2], the FL solution has been proven to be globally convergent for convex optimization problems and exhibits good convergence performance. So FL is an effective solution to the distributed learning problem presented in (1).

4 Design of MFL

In this section, we introduce the design of MFL to solve the distributed learning problem shown in (1). We first discuss the motivation of our work. Then we present the design of MFL detailedly and the learning problem based on federated system. The main notations of MFL design and analysis are summarized in Table 1.

TABLE 1: MFL notation summary

Notation	Definition
T; K; N	number of total local iterations; number of global aggregations/number of intervals; number of edge nodes
t; k; τ; $[k]$	iteration index; interval index; aggregation frequency with $\tau = T/K$; the interval $[(k-1)\tau, k\tau]$
$\mathbf{w}^*$; $\mathbf{w}^{\mathrm{f}}$	global optimal parameter of $F(\cdot)$; the optimal parameter that MFL can obtain in Algorithm 1
η; β; ρ; γ	the learning step size of MGD or GD; the β-smooth parameter of $F_i(\cdot)$; the ρ-Lipschitz parameter of $F_i(\cdot)$; the momentum attenuation factor which decides the proportion of momentum term in MGD
$\mathcal{D}_i$; $\mathcal{D}$	the local dataset of node i; the global dataset
δ_i; δ	the upper bound between $\nabla F(\mathbf{w})$ and $\nabla F_i(\mathbf{w})$; the average of δ_i over all nodes
$F_i(\cdot)$; $F(\cdot)$	the loss function of node i; the global loss function
$\mathbf{d}(t)$; $\mathbf{w}(t)$	the global momentum parameter at iteration round t; the global model parameter at iteration round t
$\widetilde{\mathbf{d}}_i(t)$; $\widetilde{\mathbf{w}}_i(t)$	the local momentum parameter of node i at iteration round t; the local model parameter at iteration round t
$\mathbf{d}_{[k]}(t)$; $\mathbf{w}_{[k]}(t)$	the momentum parameter of centralized MGD at iteration round t in $[k]$; the model parameter of centralized MGD at iteration round t in $[k]$
$\theta_{[k]}(t)$; θ; p	the angle between vector $\nabla F(\mathbf{w}_{[k]}(t))$ and $\mathbf{d}_{[k]}(t)$; θ is the maximum of $\theta_{[k]}(t)$ for $1 \leq k \leq K$ with $t \in [k]$; p is the maximum ratio of $\|\mathbf{d}_{[k]}(t)\|$ and $\|\nabla F(\mathbf{w}_{[k]}(t))\|$ for $1 \leq k \leq K$ with $t \in [k]$

4.1 Motivation

Since MGD improves the convergence rate of GD [24], we want to apply MGD to local update steps of FL and hope that the proposed MFL will accelerate the convergence rate for federated networks.

Firstly, we illustrate the intuitive influence on optimization problem after introducing the momentum term into gradient updating methods. Considering GD, the update reduction of the parameter is $\eta\nabla F(\mathbf{w}(t-1))$ which is only proportional to the gradient of $\mathbf{w}(t-1)$. The update direction of GD is always along gradient descent so that an oscillating update path could be caused, as shown by the GD update path in Fig. 2. However, the update reduction of parameter for MGD is a superposition of $\eta\nabla F(\mathbf{w}(t-1))$ and $\gamma(\mathbf{w}(t-2)-\mathbf{w}(t-1))$ which is the momentum term. As shown by the MGD update path in Fig. 2, utilizing the momentum term can deviate the direction of parameter update to the optimal decline significantly and mitigate the oscillation caused by GD. In Fig. 2, GD has an oscillating update path and costs seven iterations to reach the optimal point while MGD only needs three iterations to do that, which demonstrates mitigating the oscillation by MGD leads to a faster convergence rate.

Because edge nodes of distributed networks are usually resource-constrained, solutions to convergence acceleration can attain higher resources utilization efficiency. Thus, motivated by the property that MGD improves convergence rate, we use MGD to perform local update of FL and this approach is named MFL.

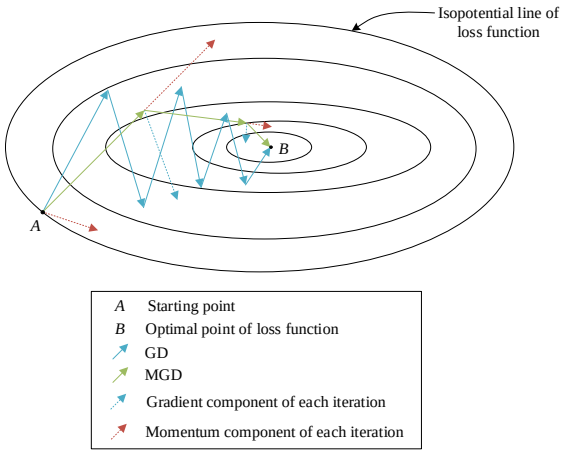


Fig. 2: Comparison of MGD and GD

In the following subsections, we design the MFL learning paradigm and propose the learning problem based on the MFL design.

4.2 MFL

In the MFL design, we use $\widetilde{\mathbf{d}}_i(t)$ and $\widetilde{\mathbf{w}}_i(t)$ to denote momentum parameter and model parameter for node i, respectively. All edge nodes are set to embed the same machine learning models. So the local loss functions $F_i(\mathbf{w})$ is the same for all nodes, and the dimension of both the model parameters and the momentum parameters are consistent. The parameters setup of MFL is similar to that of FL. We use t to denote the local iteration index for $t = 0, 1, ..., \tau$ to denote the aggregation frequency and $[k]$ to denote the interval $[(k-1)\tau, k\tau]$ where k denotes the interval index for $k = 1, 2, ...$ At $t = 0$, the momentum parameters and the model parameters of all nodes are initialized to the same values, respectively. When $t \in [k]$, $\widetilde{\mathbf{d}}_i(t)$ and $\widetilde{\mathbf{w}}_i(t)$ are updated based on MGD, which is called *local update steps*. When $t = k\tau$, MFL performs *global aggregation steps* where $\widetilde{\mathbf{d}}_i(t)$ and $\widetilde{\mathbf{w}}_i(t)$ are sent to the central server synchronously. Then in the central server, the global momentum parameter $\mathbf{d}(t)$ and the global model parameter $\mathbf{w}(t)$ are obtained by taking a weighted average of the received parameters, respectively, and are sent back to all edge nodes for the next interval.

The learning rules of MFL include the local update and the global aggregation steps. By continuous alternations of local update and global aggregation, MFL can perform its learning process to minimize the global loss function $F(\mathbf{w})$. We describe the MFL learning process as follows.

First of all, we set initial values for $\widetilde{\mathbf{d}}_i(0)$ and $\widetilde{\mathbf{w}}_i(0)$. Then

1) Local Update: When $t \in [k]$, local update is performed at each edge node by

$$\widetilde{\mathbf{d}}_i(t) = \gamma\widetilde{\mathbf{d}}_i(t-1) + \nabla F_i(\widetilde{\mathbf{w}}_i(t-1)) \tag{6}$$

$$\widetilde{\mathbf{w}}_i(t) = \widetilde{\mathbf{w}}_i(t-1) - \eta\widetilde{\mathbf{d}}_i(t). \tag{7}$$

According to (6) and (7), node i performs MGD to optimize the loss function $F_i(\cdot)$ defined on its own dataset.

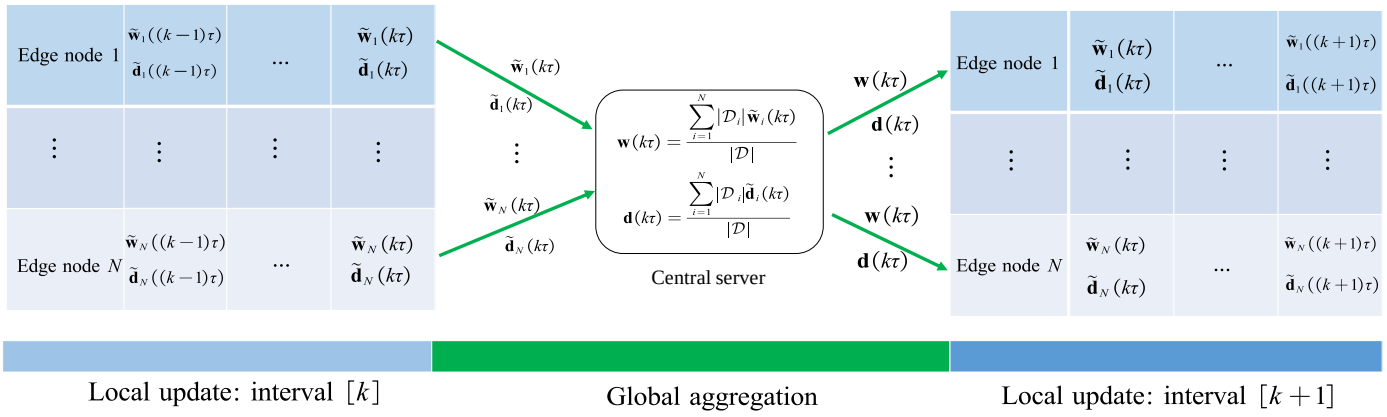


Fig. 3: Illustration of MFL local update and global aggregation steps from interval $[k]$ to $[k+1]$.

Algorithm 1 *MFL* The dataset in each node has been set, and the machine learning model embedded in edge nodes has been chosen. We have set appropriate model parameters η and γ.

Input:
The limited number of local updates in each node T
A given aggregation frequency τ
Output:
The final global model weight vector $\mathbf{w}^{\mathrm{f}}$

1: Set the initial value of $\mathbf{w}^{\mathrm{f}}$, $\widetilde{\mathbf{w}}_i(0)$ and $\widetilde{\mathbf{d}}_i(0)$.
2: **for** $t = 1, 2, ..., T$ **do**
3: Each node i performs local update in parallel according to (6) and (7).//*Local update*
4: **if** $t == k\tau$ where k is a positive integer **then**
5: Set $\widetilde{\mathbf{d}}_i(t) \leftarrow \mathbf{d}(t)$ and $\widetilde{\mathbf{w}}_i(t) \leftarrow \mathbf{w}(t)$ for all nodes where $\mathbf{d}(t)$ and $\mathbf{w}(t)$ is obtained by (8) and (9) respectively. //*Global aggregation*
Update $\mathbf{w}^{\mathrm{f}} \leftarrow \arg\min_{\mathbf{w}\in\{\mathbf{w}^{\mathrm{f}}, \mathbf{w}(k\tau)\}} F(\mathbf{w})$
6: **end if**
7: **end for**

2) Global Aggregation: When $t = k\tau$, node i transmits $\widetilde{\mathbf{w}}_i(k\tau)$ and $\widetilde{\mathbf{d}}_i(k\tau)$ to the central server which takes weighted averages of the received parameters from N nodes to obtain the global parameters $\mathbf{w}(k\tau)$ and $\mathbf{d}(k\tau)$, respectively. The aggregation rules are presented as follows:

$$\mathbf{d}(t) = \frac{\sum_{i=1}^{N} |\mathcal{D}_i| \widetilde{\mathbf{d}}_i(t)}{|\mathcal{D}|} \tag{8}$$

$$\mathbf{w}(t) = \frac{\sum_{i=1}^{N} |\mathcal{D}_i| \widetilde{\mathbf{w}}_i(t)}{|\mathcal{D}|}. \tag{9}$$

Then the central server sends $\mathbf{d}(k\tau)$ and $\mathbf{w}(k\tau)$ back to all edge nodes where $\widetilde{\mathbf{d}}_i(k\tau) = \mathbf{d}(k\tau)$ and $\widetilde{\mathbf{w}}_i(k\tau) = \mathbf{w}(k\tau)$ are set to enable the local update in the next interval $[k+1]$. Note that only if $t = k\tau$, the value of the global parameters $\mathbf{w}(t)$ and $\mathbf{d}(t)$ can be observed. But we define $\mathbf{d}(t)$ and $\mathbf{w}(t)$ for all t to facilitate the following analysis. A typical alternation is shown in Fig. 3 which intuitively illustrates the learning steps of MFL in interval $[k]$ and $[k+1]$.

The learning problem of MFL to attain the optimal model parameter is presented as (1). However, the edge nodes have limited computation resources with a finite number of local iterations. We assume that T is the number of local iterations and K is the corresponding number of global aggregations. Thus, we have $t \leq T$ and $k \leq K$ with $T = K\tau$. Considering that $\mathbf{w}(t)$ is unobservable for $t \neq k\tau$, we use $\mathbf{w}^{\mathrm{f}}$ to denote the achievable optimal model parameter defined on resource-constrained MFL network. Hence, the learning problem is to obtain $\mathbf{w}^{\mathrm{f}}$ within T local iterations particularly, i.e.,

$$\mathbf{w}^{\mathrm{f}} \triangleq \arg\min_{\mathbf{w}\in\{\mathbf{w}(k\tau): k=1,2,...,K\}} F(\mathbf{w}). \tag{10}$$

The optimization algorithm of MFL is explained in Algorithm 1.

5 Convergence Analysis

In this section, we firstly make some definitions and assumptions for MFL convergence analysis. Then based on these preliminaries, global convergence properties of MFL following Algorithm 1 are established and an upper bound on MFL convergence rate is derived. Also MFL convergence performance with related parameters is analyzed.

5.1 Preliminaries

First of all, to facilitate the analysis, we assume that $F_i(\mathbf{w})$ satisfies the following conditions:

Assumption 1. *For $F_i(\mathbf{w})$ in node i, we assume the following conditions:*
1) $F_i(\mathbf{w})$ is convex
2) $F_i(\mathbf{w})$ is ρ-Lipschitz, i.e., $|F_i(\mathbf{w}_1) - F_i(\mathbf{w}_2)| \leq \rho\|\mathbf{w}_1 - \mathbf{w}_2\|$ for some $\rho > 0$ and any $\mathbf{w}_1$, $\mathbf{w}_2$
3) $F_i(\mathbf{w})$ is β-smooth, i.e., $\|\nabla F_i(\mathbf{w}_1) - \nabla F_i(\mathbf{w}_2)\| \leq \beta\|\mathbf{w}_1 - \mathbf{w}_2\|$ for some $\beta > 0$ and any $\mathbf{w}_1$, $\mathbf{w}_2$
4) $F_i(\mathbf{w})$ is μ-strong, i.e., $aF_i(\mathbf{w}_1)+(1-a)F_i(\mathbf{w}_2) \geq F_i(a\mathbf{w}_1+(1-a)\mathbf{w}_2) + \frac{a(1-a)\mu}{2}\|\mathbf{w}_1 - \mathbf{w}_2\|^2$, $a \in [0,1]$ for some $\mu > 0$ and any $\mathbf{w}_1$, $\mathbf{w}_2$ [30, Theorem 2.1.9]

Because guaranteeing the global convergence of centralized MGD requires that the objective function is strongly convex [24], it is necessary to assume the condition 4. Assumption 1 is satisfied for some learning models such as SVM, linear regression and logistic regression whose loss functions are presented in Table 2. Experimental results

as presented in Section 7.2.1 show that for non-convex model such as CNN whose loss function does not satisfy Assumption 1, MFL also performs well. From Assumption 1, we can obtain the following lemma:

Lemma 1. *$F(\mathbf{w})$ is convex, ρ-Lipschitz, β-smooth and μ-strong.*

Proof. According to the definition of $F(\mathbf{w})$ from (2), triangle inequality and the definition of ρ-Lipschitz, β-smooth and μ-strong, we can derive that $F(\mathbf{w})$ is convex, ρ-Lipschitz, β-smooth and μ-strong directly. □

Then we introduce the gradient divergence between $\nabla F(\mathbf{w})$ and $\nabla F_i(\mathbf{w})$ for any node i. It comes from the nature of the difference in datasets distribution.

Definition 2 (Gradient divergence)**.** *We define δ_i as the upper bound between $\nabla F(\mathbf{w})$ and $\nabla F_i(\mathbf{w})$ for any node i, i.e.,*

$$\|\nabla F(\mathbf{w}) - \nabla F_i(\mathbf{w})\| \leq \delta_i. \tag{11}$$

Also, we define the average gradient divergence

$$\delta \triangleq \frac{\sum_i |\mathcal{D}_i|\delta_i}{|\mathcal{D}|}. \tag{12}$$

Boundedness of δ_i and δ: Based on condition 3 of Assumption 1, we let $\mathbf{w}_2 = \mathbf{w}_i^*$ where $\mathbf{w}_i^*$ is the optimal value for minimizing $F_i(\mathbf{w})$. Because $F_i(\mathbf{w})$ is convex, we have $\|\nabla F_i(\mathbf{w}_1)\| \leq \beta\|\mathbf{w}_1 - \mathbf{w}_i^*\|$ for any $\mathbf{w}_1$, which means $\|\nabla F_i(\mathbf{w})\|$ is finite for any $\mathbf{w}$. According to Definition 1 and the linearity of gradient operator, global loss function $\nabla F(\mathbf{w})$ is obtained by taking a weighted average of $\nabla F_i(\mathbf{w})$. Therefore, $\|\nabla F(\mathbf{w})\|$ is finite, and $\|\nabla F(\mathbf{w}) - \nabla F_i(\mathbf{w})\|$ has an upper bound, i.e., δ_i is bounded. Further, δ is still bounded from the linearity in (12).

Since local update steps of MFL perform MGD, the upper bounds of MFL and MGD convergence rate exist certain connections in the same interval. For the convenience of analysis, we use variables $\mathbf{d}_{[k]}(t)$ and $\mathbf{w}_{[k]}(t)$ to denote the momentum parameter and the model parameter of *centralized MGD* in each interval $[k]$, respectively. This centralized MGD is defined on global dataset and updated based on global loss function $F(\mathbf{w})$. In interval $[k]$, the update rules of centralized MGD follow:

$$\mathbf{d}_{[k]}(t) = \gamma \mathbf{d}_{[k]}(t-1) + \nabla F(\mathbf{w}_{[k]}(t-1)) \tag{13}$$
$$\mathbf{w}_{[k]}(t) = \mathbf{w}_{[k]}(t-1) - \eta \mathbf{d}_{[k]}(t). \tag{14}$$

At the beginning of interval $[k]$, the momentum parameter $\mathbf{d}_{[k]}(t)$ and the model parameter $\mathbf{w}_{[k]}(t)$ of centralized MGD are synchronized with the corresponding parameters of MFL, i.e.,

$$\mathbf{d}_{[k]}((k-1)\tau) \triangleq \mathbf{d}((k-1)\tau)$$
$$\mathbf{w}_{[k]}((k-1)\tau) \triangleq \mathbf{w}((k-1)\tau).$$

For each interval $[k]$, the centralized MGD is performed by iterations of (13) and (14). In the Fig. 4, we illustrate the distinctions between $F(\mathbf{w}(t))$ and $F(\mathbf{w}_{[k]}(t))$ intuitively.

Comparing with centralized MGD, MFL aggregation interval with $\tau > 1$ brings global update delay because of the fact that centralized MGD performs global update on every iteration while MFL is allowed to spread its global parameter to edge nodes after τ local updates. Therefore, the convergence performance of MFL is worse than that of MGD, which is essentially from the imbalance between several computation rounds and one communication round in MFL design. The following subsection provides the resulting convergence performance gap between these two approaches.

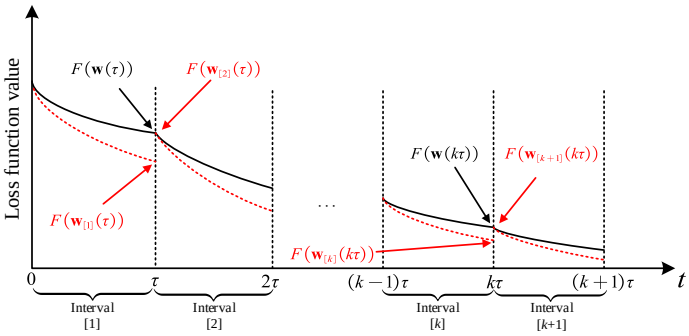


Fig. 4: Illustration of the difference between MGD and MFL in intervals

5.2 Gap between MFL and Centralized MGD in Interval $[k]$

Firstly, considering a special case, we consider the gap between MFL and centralized MGD for $\tau = 1$. From physical perspective, MFL performs global aggregation after every local update and there does not exist global parameter update delay, i.e, the performance gap is zero. In Appendix A, we prove that MFL is equivalent to MGD for $\tau = 1$ from theoretical perspective.

Now considering general case for any $\tau \geq 1$, the upper bound of gap between $\mathbf{w}(t)$ and $\mathbf{w}_{[k]}(t)$ can be derived as follows.

Proposition 1 (Gap of MFL&centralized MGD in intervals)**.** *Given $t \in [k]$, the gap between $\mathbf{w}(t)$ and $\mathbf{w}_{[k]}(t)$ can be expressed by*

$$\|\mathbf{w}(t) - \mathbf{w}_{[k]}(t)\| \leq h(t-(k-1)\tau), \tag{15}$$

where we define

$$A \triangleq \frac{(1+\gamma+\eta\beta)+\sqrt{(1+\gamma+\eta\beta)^2-4\gamma}}{2\gamma},$$
$$B \triangleq \frac{(1+\gamma+\eta\beta)-\sqrt{(1+\gamma+\eta\beta)^2-4\gamma}}{2\gamma},$$

$$E \triangleq \frac{A}{(A-B)(\gamma A-1)},$$
$$F \triangleq \frac{B}{(A-B)(1-\gamma B)}$$

and $h(x)$ yields

$$h(x) = \eta\delta\left[E(\gamma A)^x + F(\gamma B)^x - \frac{1}{\eta\beta} - \frac{\gamma(\gamma^x-1)-(\gamma-1)x}{(\gamma-1)^2}\right] \tag{16}$$

for $0 < \gamma < 1$ and any $x = 0, 1, 2, ...$

Because $F(\mathbf{w})$ is ρ-Lipschitz from Lemma 1*, it holds that*

$$F(\mathbf{w}(t)) - F(\mathbf{w}_{[k]}(t)) \leq \rho h(t-(k-1)\tau). \tag{17}$$

Proof. Firstly, we derive an upper bound of $\|\widetilde{\mathbf{w}}_i(t)-\mathbf{w}_{[k]}(t)\|$ for node i. On the basis of this bound, we extend this result from the local cases to the global one to obtain the final result. The detailed proving process is presented in Appendix B. □

Because $h(1) = h(0) = 0$ and $h(x)$ increases with x for $x \geq 1$, which are proven in Appendix C, we always have $h(x) \geq 0$ for $x = 0, 1, 2, ...$

From Proposition 1, in any interval $[k]$, we have $h(0) = 0$ for $t = (k-1)\tau$, which fits the definition $\mathbf{w}_{[k]}((k-1)\tau) = \mathbf{w}((k-1)\tau)$. We still have $h(1) = 0$ for $t = (k-1)\tau+1$. This means that there is no gap between MFL and centralized MGD when local update is only performed once after the global aggregation.

It is easy to find that if $\tau = 1$, $t-(k-1)\tau$ is either 0 or 1. Because $h(1) = h(0) = 0$, the upper bound in (15) is zero, and there is no gap between $F(\mathbf{w}(t))$ and $F(\mathbf{w}_{[k]}(t))$ from (17). This is consistent with Appendix A where MFL yields centralized MGD for $\tau = 1$. In any interval $[k]$, we have $t-(k-1)\tau \in [0, \tau]$. If $\tau > 1$, $t-(k-1)\tau$ can be larger than 1. When $x > 1$, we know that $h(x)$ increases with x. According to the definition of A, B, E and F, we can obtain $\gamma A > 1$, $\gamma B < 1$ and $E, F > 0$ easily. Because $0 < \gamma < 1$, the last term will linearly decrease with x when x is large. Therefore, the first exponential term $E(\gamma A)^x$ in (16) will be dominant when x is large and the gap between $\mathbf{w}(t)$ and $\mathbf{w}_{[k]}(t)$ increases exponentially with t.

Also we find $h(x)$ is proportional to the average gradient gap δ. It is because the greater the local gradient divergences at different nodes are, the larger the gap will be. So considering the extreme situation where all nodes have the same data samples ($\delta = 0$ because the local loss function are the same), the gap between $\mathbf{w}(t)$ and $\mathbf{w}_{[k]}(t)$ is zero and MFL is equivalent to centralized MGD.

5.3 Global Convergence

We have derived an upper bound between $F(\mathbf{w}(t))$ and $F(\mathbf{w}_{[k]}(t))$ for $t \in [k]$. According to the definition of MFL, in the beginning of each interval $[k]$, we set $\mathbf{d}_{[k]}((k-1)\tau) = \mathbf{d}((k-1)\tau)$ and $\mathbf{w}_{[k]}((k-1)\tau) = \mathbf{w}((k-1)\tau)$. The global upper bound on the convergence rate of MFL can be derived based on Proposition 1.

The following definitions are made to facilitate analysis. Firstly, we use $\theta_{[k]}(t)$ to denote the angle between vector $\nabla F(\mathbf{w}_{[k]}(t)$ and $\mathbf{d}_{[k]}(t)$ for $t \in [k]$, i.e.,

$$\cos\theta_{[k]}(t) \triangleq \frac{\nabla F(\mathbf{w}_{[k]}(t))^{\mathrm{T}}\mathbf{d}_{[k]}(t)}{\|\nabla F(\mathbf{w}_{[k]}(t)\|\|\mathbf{d}_{[k]}(t)\|}$$

where θ is defined as the maximum value of $\theta_{[k]}(t)$ for $1 \leq k \leq K$ with $t \in [k]$, i.e.,

$$\theta \triangleq \max_{1\leq k\leq K, t\in[k]} \theta_{[k]}(t).$$

Then we define

$$p \triangleq \max_{1\leq k\leq K, t\in[k]} \frac{\|\mathbf{d}_{[k]}(t)\|}{\|\nabla F(\mathbf{w}_{[k]}(t))\|}$$

and

$$\omega \triangleq \min_{k} \frac{1}{\|\mathbf{w}((k-1)\tau) - \mathbf{w}^*\|^2}.$$

Based on Proposition 1 which gives an upper bound of loss function difference between MFL and centralized MGD, global convergence rate of MFL can be derived as follows.

Lemma 2. *If the following conditions are satisfied:*
1) $\cos\theta \geq 0$, $0 < \eta\beta < 1$ *and* $0 \leq \gamma < 1$;
There exists $\varepsilon > 0$ *which makes*
2) $F(\mathbf{w}_{[k]}(k\tau)) - F(\mathbf{w}^*) \geq \varepsilon$ *for all* k;
3) $F(\mathbf{w}(T)) - F(\mathbf{w}^*) \geq \varepsilon$;
4) $\omega\alpha - \frac{\rho h(\tau)}{\tau\varepsilon^2} > 0$ *hold,*
then we have

$$F(\mathbf{w}(T)) - F(\mathbf{w}^*) \leq \frac{1}{T\left(\omega\alpha - \frac{\rho h(\tau)}{\tau\varepsilon^2}\right)} \tag{18}$$

where we defined

$$\alpha \triangleq \eta(1-\frac{\beta\eta}{2})+\eta\gamma(1-\beta\eta)\cos\theta-\frac{\beta\eta^2\gamma^2p^2}{2}.$$

Proof. The proof is presented in Appendix D. □

On the basis of Lemma 2, we further derive the following proposition which demonstrates the global convergence of MFL and gives its upper bound on convergence rate.

Proposition 2 (MFL global convergence)**.** *Given* $\cos\theta \geq 0$, $0 < \eta\beta < 1$, $0 \leq \gamma < 1$ *and* $\alpha > 0$, *we have*

$$F(\mathbf{w}^{\mathrm{f}}) - F(\mathbf{w}^*) \leq \frac{1}{2T\omega\alpha} + \sqrt{\frac{1}{4T^2\omega^2\alpha^2} + \frac{\rho h(\tau)}{\omega\alpha\tau}} + \rho h(\tau). \tag{19}$$

Proof. The specific proving process is shown in Appendix E. □

According to the above Proposition 2, we get an upper bound of $F(\mathbf{w}^{\mathrm{f}}) - F(\mathbf{w}^*)$ which is a function of T and τ. From inequality (19), we can find that MFL linearly converges to a lower bound $\sqrt{\frac{\rho h(\tau)}{\omega\alpha\tau}} + \rho h(\tau)$. Because $h(\tau)$ is related to τ and δ, aggregation intervals ($\tau > 1$) and different data distribution collectively lead to that MFL does not converge to the optimum.

In the following, we discuss the influence of τ on the convergence bound. If $\tau = 1$, we have $\rho h(\tau) = 0$ so that $F(\mathbf{w}^{\mathrm{f}}) - F(\mathbf{w}^*)$ linearly converges to zero as $T \to \infty$, and the convergence rate yields $\frac{1}{T\omega\alpha}$. Noting $h(\tau) > 0$ if $\tau > 1$, we can find that in this case, $F(\mathbf{w}^{\mathrm{f}}) - F(\mathbf{w}^*)$ converges to a non-zero bound $\sqrt{\frac{\rho h(\tau)}{\omega\alpha\tau}} + \rho h(\tau)$ as $T \to \infty$. On the one hand, if there does not exist communication resources limit, setting aggregation frequency $\tau = 1$ and performing global aggregation after each local update can reach the optimal convergence performance of MFL. On the other hand, aggregation interval ($\tau > 1$) can let MFL effectively utilize the communication resources of each node, but bring about a decline of convergence performance.

6 Comparison Between FL and MFL

In this section, we make a comparison of convergence performance between MFL and FL.

The closed-form solution of the upper bound on FL convergence rate has been derived in [17, Theorem 2]. It is presented as follows.

$$F(\mathbf{w}_{FL}^{\mathrm{f}})-F(\mathbf{w}^*)\leq\frac{1}{2\eta\varphi T}+\sqrt{\frac{1}{4\eta^2\varphi^2T^2}+\frac{\rho h_{FL}(\tau)}{\eta\varphi\tau}}+\rho h_{FL}(\tau). \tag{20}$$

According to [17],

$$h_{FL}(\tau)=\frac{\delta}{\beta}((\eta\beta+1)^\tau-1)-\eta\delta\tau$$

and $\varphi=\omega_{FL}(1-\frac{\eta\beta}{2})$ where the expression of ω_{FL} is consistent with ω. Differing from that of ω, $\mathbf{w}((k-1)\tau)$ in the definition of ω_{FL} is the global model parameter of FL.

We assume that both MFL and FL solutions are applied in the system model proposed in Fig. 1. They are trained based on the same training dataset with the same machine learning model. The loss function $F_i(\cdot)$ and global loss function $F(\cdot)$ of MFL and FL are the same, respective. The corresponding parameters of MFL and FL are equivalent including τ, η, ρ, δ and β. We set the same initial value $\mathbf{w}(0)$ of MFL and FL. Because both MFL and FL are convergent, we have $\omega=\frac{1}{\|\mathbf{w}(0)-\mathbf{w}^*\|^2}$. Then according to the definitions of ω and ω_{FL}, we have $w=w_{FL}$. Therefore, the corresponding parameters of MFL and FL are the same and we can compare the convergences between FL and MFL conveniently.

For convenience, we use $f_1(T)$ and $f_2(T)$ to denote the upper bound on convergence rate of MFL and FL, respectively. Then we have

$$f_1(T)\triangleq\frac{1}{2T\omega\alpha}+\sqrt{\frac{1}{4T^2\omega^2\alpha^2}+\frac{\rho h(\tau)}{\omega\alpha\tau}}+\rho h(\tau) \tag{21}$$

and

$$f_2(T)\triangleq\frac{1}{2\eta\varphi T}+\sqrt{\frac{1}{4\eta^2\varphi^2T^2}+\frac{\rho h_{FL}(\tau)}{\eta\varphi\tau}}+\rho h_{FL}(\tau). \tag{22}$$

We consider the special case of $\gamma\to0$. For $\omega\alpha$ and $\eta\varphi$, we can obtain $\omega\alpha\to\omega\eta(1-\frac{\beta\eta}{2})=\eta\varphi$ from the definition of α. Then for $h(\tau)$ and $h_{FL}(\tau)$, we have $\gamma A\to\eta\beta+1$ and $\gamma B\to0$. Because $\frac{A}{A-B}\to1$ and $\frac{B}{A-B}\to0$, we can further get $E\to\frac{1}{\eta\beta}$ and $F\to0$ from the definitions of E and F. So, according to (16), we have

$$\begin{aligned}\lim_{\gamma\to0}h(\tau)&=\eta\delta\left[\frac{1}{\eta\beta}(\eta\beta+1)^\tau-\frac{1}{\eta\beta}-\tau\right]\\&=\frac{\delta}{\beta}((1+\eta\beta)^\tau-1)-\eta\delta\tau=h_{FL}(\tau).\end{aligned}$$

Hence, by the above analysis under $\gamma\to0$, we can find MFL and FL have the same upper bound on convergence rate. This fact is consistent with the property that if $\gamma=0$, MFL degenerates into FL and has the same convergence rate with FL.

To avoid complicated calculations over the expressions of $f_1(T)$ and $f_2(T)$, we have the following lemma.

Lemma 3. *If there exists $T_1\geq1$ which satisfies that $\frac{1}{2T\omega\alpha}$ dominates in $f_1(T)$ and $\frac{1}{2\eta\varphi T}$ dominates in $f_2(T)$ for $T<T_1$, i.e.,*

$$\frac{1}{2T\omega\alpha}\gg\max\left\{\rho h(\tau),\sqrt{\frac{\rho h(\tau)}{\omega\alpha\tau}}\right\}$$

and

$$\frac{1}{2\eta\varphi T}\gg\max\left\{\rho h_{FL}(\tau),\sqrt{\frac{\rho h_{FL}(\tau)}{\eta\varphi\tau}}\right\},$$

then we have

$$f_1(T)\approx\frac{1}{T\omega\alpha}$$

and

$$f_2(T)\approx\frac{1}{T\eta\varphi}$$

for $T<T_1$.

Proof. This obviously holds. We can find a such T_1. For example, considering (21), if $\eta\to0$, we have $\alpha\to0$ from the definition of α and $h(\tau)\to0$ from Appendix F. So we can easily derive $\omega\alpha\rho h(\tau)\to0$ and $\sqrt{\frac{\omega\alpha\rho h(\tau)}{\tau}}\to0$. Then we can find $T_1\geq1$ which satisfies $\frac{1}{2T}\gg\omega\alpha\rho h(\tau)$ and $\frac{1}{2T}\gg\sqrt{\frac{\omega\alpha\rho h(\tau)}{\tau}}$ for $T<T_1$. Hence, we have $\frac{1}{2T\omega\alpha}$ dominates in $f_1(T)$ and $f_1(T)\approx\frac{1}{T\omega\alpha}$. For the same reason, considering (22), if $\eta\to0$, we have $\eta\varphi\to0$ from the definition of φ and $h_{FL}(\tau)\to0$ from its definition. So we can easily derive $\eta\varphi\rho h_{FL}(\tau)\to0$ and $\sqrt{\frac{\eta\varphi\rho h_{FL}(\tau)}{\tau}}\to0$. Then for $T<T_1$, $\frac{1}{2T}\gg\eta\varphi\rho h_{FL}(\tau)$ and $\frac{1}{2T}\gg\sqrt{\frac{\eta\varphi\rho h_{FL}(\tau)}{\tau}}$. Hence, we have $\frac{1}{2\eta\varphi T}$ dominates in $f_2(T)$ and $f_2(T)\approx\frac{1}{T\eta\varphi}$. □

Based on Lemma 3, we have the following proposition.

Proposition 3 (Accelerated convergence of MFL)**.** *If the following conditions are satisfied:*

1) $\eta\beta\leq1$;
2) $T<T_1$;
3) $0<\gamma<\min\{\frac{2(1-\eta\beta)\cos\theta}{\beta\eta p^2},1\}$,

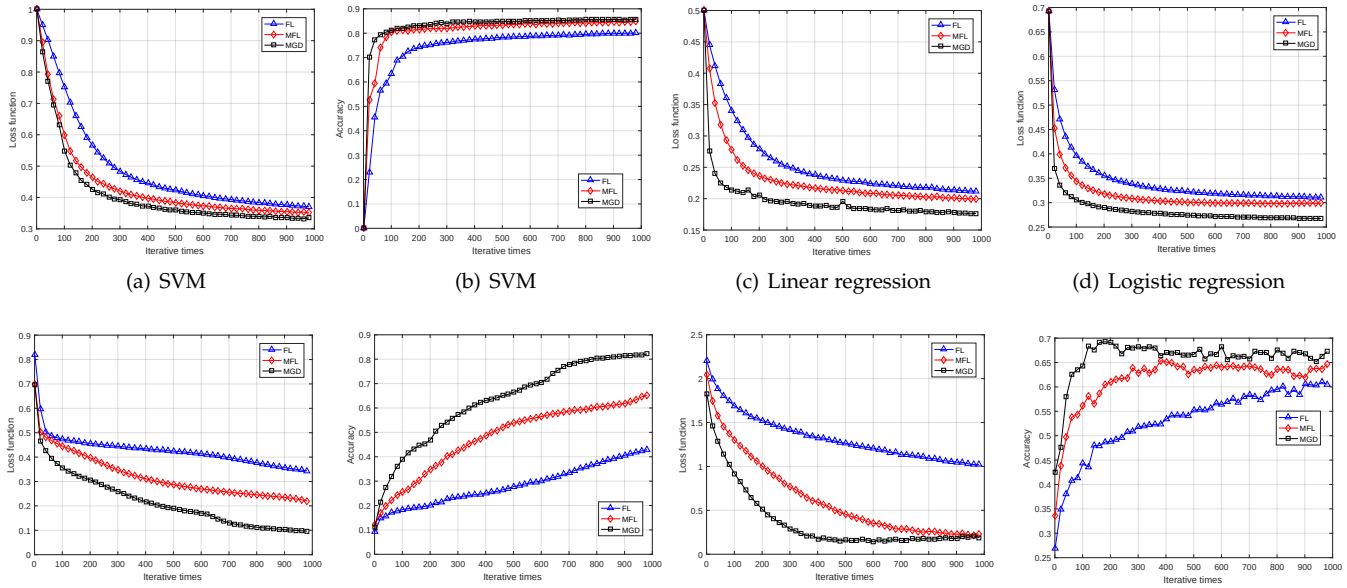
MFL *converges faster than* FL*, i.e.,*

$$f_1(T)<f_2(T).$$

Proof. From condition 1 and condition 2, we have $f_1(T)\approx\frac{1}{T\omega\alpha}$ and $f_2(T)\approx\frac{1}{T\eta\varphi}$. Due to the definition of α and φ, inequality $0<\gamma<\frac{2(1-\beta\eta)\cos\theta}{\beta\eta p^2}$ is equivalent to $\omega\alpha>\eta\varphi$. So if $\omega\alpha>\eta\varphi$, it is obvious that $\frac{1}{T\omega\alpha}<\frac{1}{T\eta\varphi}$, i.e., $f_1(T)<f_2(T)$. However, $0<\gamma<1$ is the condition of MFL convergence. Hence, condition 3 is the range of MFL convergence acceleration after combining with MFL convergence guarantee $0<\gamma<1$. □

7 Simulation and Discussion

In this section, we build and evaluate MFL system based on MNIST and CIFAR-10 datasets. We first describe the simulation environment and the relevant setups of parameters. Secondly, we present and evaluate the comparative simulation results of MFL, FL and MGD under different machine learning models, which include SVM, linear regression, logistic regression and CNN. Finally, the extensive



(a) SVM (b) SVM (c) Linear regression (d) Logistic regression

(e) CNN w.r.t. MNIST (f) CNN w.r.t. MNIST (g) CNN w.r.t. CIFAR-10 (h) CNN w.r.t. CIFAR-10

Fig. 5: Loss function values and testing accuracy under FL, MFL and MGD. (a) and (b) are the loss function and test accuracy curves of SVM, respectively; (c) and (d) are the loss function curves of linear regression and logistic regression, respectively. (e) and (f) are the loss function and test accuracy curves of CNN trained on MNIST, respectively. (g) and (h) are the loss function and test accuracy curves of CNN trained on CIFAR-10, respectively.

TABLE 2: Loss function of three machine learning models

Model	Loss function
SVM	$\frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{2\|\mathcal{D}_i\|}\sum_j \max\{0; 1 - y_j\mathbf{w}^{\mathrm{T}}\mathbf{x}_j\}$
Linear regression	$\frac{1}{2\|\mathcal{D}_i\|}\sum_j \|y_j - \mathbf{w}^{\mathrm{T}}\mathbf{x}_j\|^2$
Logistic regression	$-\frac{1}{\|\mathcal{D}_i\|}\sum_j \|y_j \log\sigma(\mathbf{w},\mathbf{x}_j) + (1 - y_j)\log(1 - \sigma(\mathbf{w},\mathbf{x}_j))\|$ where $\sigma(\mathbf{w},\mathbf{x}_j)$ is given as (23)

experiments are implemented to explore the impacts of γ, τ and non-i.i.d data distribution on MFL convergence performance, and to investigate the communication efficiency of MFL compared with that of FL.

7.1 Simulation Setup

Using the Python, we build a federated network framework where distributed edge nodes coordinate with the central server. In our network, the number of edge nodes can be chosen arbitrarily. SVM, linear regression, logistic regression and CNN are applied to model training. Loss functions of the first three models at node i are presented as in Table 2 [31], and the loss function of CNN is cross-entropy (see [32] for details). Note that $|\mathcal{D}_i|$ is the number of training samples in node i and the loss function of logistic regression is cross-entropy. For logistic regression, model output $\sigma(\mathbf{w},\mathbf{x}_j)$ is sigmoid function for non-linear transform. It is defined by

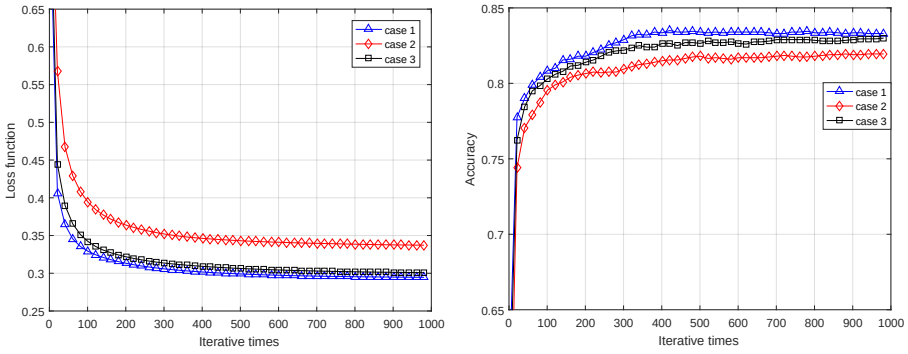
$$\sigma(\mathbf{w},\mathbf{x}_j) \triangleq \frac{1}{1+e^{-\mathbf{w}^{\mathrm{T}}\mathbf{x}_j}}. \tag{23}$$

In our experiments, training and testing samples are randomly allocated to each node, which means that the information of each node is uniform and the data distribution at edge nodes is i.i.d. (Only in Section 7.2.2, non-i.i.d data distribution are used and the rest of experiments use i.i.d data distribution). We use FL and centralized MGD as benchmarks for comparison of MFL. If based on SVM, linear and logistic regression models, the deterministic gradient methods are performed for MFL, FL and centralized MGD. However, if based on CNN model, the stochastic gradient methods are used for MFL, FL and centralized MGD due to the large training data size.

SVM, linear and logistic regression are trained and tested on MNIST dataset [33] which contains 50,000 training handwritten digits and 10,000 testing handwritten digits. In our experiments, we only utilize 5,000 training samples and 5,000 testing samples because of the limited processing capacities of GD and MGD. In this dataset, the j-th sample $\mathbf{x}_j$ is a 784-dimensional input vector which is vectorized from 28×28 pixel matrix and y_j is the scalar label corresponding to $\mathbf{x}_j$. SVM, linear and logistic regression are used to classify whether the digit is even or odd. If the image of $\mathbf{x}_j$ represents an even number, then we set $y_j = 1$. Otherwise, $y_j = -1$. But for logistic regression, we set $y_j = 1$ for the even number and $y_j = 0$ for the odd.

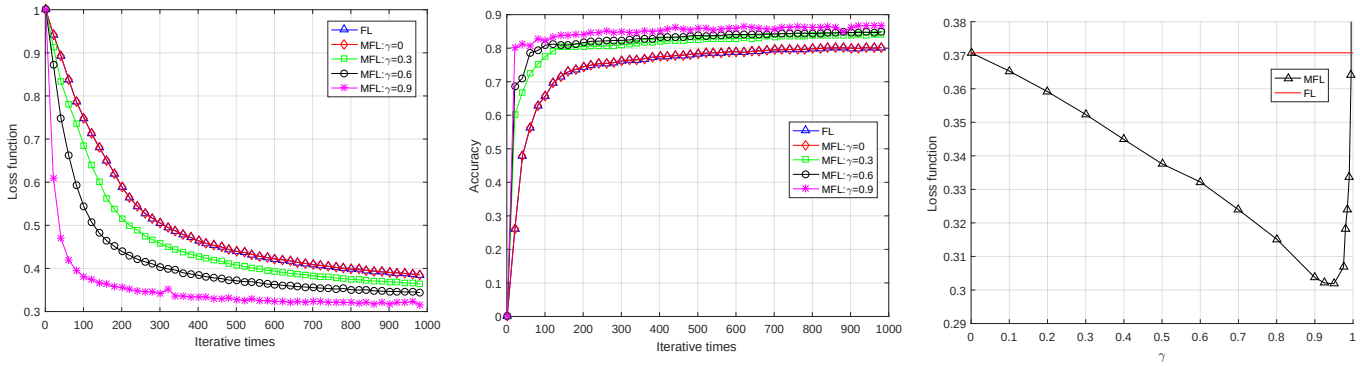
CNN is trained based on MNIST and CIFAR-10 datasets. The CIFAR-10 dataset includes 50,000 color images for training and 10,000 color image for testing, and has 10 different types of objects [34]. We use CNN to perform the classification among the 10 different labels under MNIST and CIFAR-10 datasets, respectively.

For experimental setups, we set 4 edge nodes in FL and MFL and the training models are distributed into all the edge nodes. The same initializations of model parameters are performed and the same data distributions are set for



(a) (b)

Fig. 6: (a) and (b) are loss function and testing accuracy curves under different data distribution cases, respectively.



(a) (b) (c)

Fig. 7: The influence of γ on MFL convergence. (a)Loss function values with iterative times under different γ; (b)Testing accuracy with iterative times under different γ; (c)Loss function values with γ when $T = 1000$.

MFL and FL. Also $\mathbf{d}_i(0) = \mathbf{0}$ is set for node i. We set the learning step size $\eta = 0.002$ which is sufficiently small, SVM parameter $\lambda = 0.3$ and the total number of local iterations $T = 1,000$ for the following simulations.

7.2 Simulation Evaluation

In this subsection, we verify the convergence acceleration of MFL and explore the effects of non-i.i.d data distribution, γ and τ on MFL convergence by simulation evaluation. We further investigate the communication efficiency of MFL compared with that of FL.

7.2.1 Convergence

In our first simulation, the models of SVM, linear regression, logistic regression and CNN are trained and we verify the accelerated convergence of MFL. We set aggregation frequency $\tau = 4$ and momentum attenuation factor $\gamma = 0.5$. MFL, FL and MGD are performed based on the four machine learning models. MGD is implemented based on the global dataset which is obtained by gathering the distributed data on all nodes. The global loss functions of the three solutions are defined based on the same global training and testing data.

The curves of loss function values and accuracy with iterative times are presented in Fig. 5. We can see that the loss function curves for all the learning models are gradually convergent with iterative times. Similarly, the test accuracy curves for SVM and CNN gradually rise until convergence with iterative times. Therefore, convergence of MFL is verified. We also see that the descent speeds of MFL loss function curves on the four learning models are always faster than those of FL while the centralized MGD convergence speeds are fastest. So compared with FL, MFL provides a significant improvement on convergence rate. MGD converges with the fastest speed because MFL and FL suffer the delay in global gradient update for $\tau = 4$. Finally, compared with the results of CNN and SVM, we can conclude that based on CNN model, MFL still shows similar convergence performance compared with what MFL shows in convex model training. So the proposed MFL can perform well in neural networks with non-convex loss functions.

Because linear and logistic regression can not provide the testing accuracy curves, we focus on the SVM model in the following experiments and further explore the impact of MFL parameters on convergence rate.

7.2.2 Effect of non-i.i.d data distribution

In the experiment, we consider three cases to distribute the data samples into different nodes. The three data distribution cases at edge nodes are representative for uniform, totally non-uniform information and the mixture of the

previous two cases, respectively. The specific settings of the three cases are as follows:

- Case 1: For the uniform information distribution, each data sample is randomly allocated to a node. In this case, we think that the data on each node have uniform characteristics. Therefore, this case satisfies i.i.d data distribution as a benchmark.

- Case 2: All data samples at an individual node have the same label (If there are more labels than nodes, each node could have samples with more than one label but not the total number of labels). Because the global dataset has multiple labels, this case will lead to a non-uniform information distribution, which means characteristics brought by each node are not uniform. Thus, this case corresponds to totally non-i.i.d data distribution.

- Case 3: In this case, the first half of N nodes perform random allocation rule of Case 1 to obtain uniform information and the second half of the nodes perform allocation rule of Case 2. This case is a combination of uniform and non-uniform information. We use this case to explore the effect of the mixture of i.i.d and non-i.i.d data distribution.

In the experiment, SVM is used for the training of MFL network under the above data distribution cases. We set aggregation frequency $\tau = 4$ and momentum attenuation factor $\gamma = 5$ for general MFL algorithm.

The experimental results are shown in Fig. 6. The two subfigures show the MFL convergence influence made by different data distributions. We see that the loss function and testing accuracy curves of MFL are always convergent whether data distribution at edge nodes is i.i.d or non-i.i.d, which means even though under non-uniform information distribution, MFL training still achieves expected convergence and shows its robustness. We also see that Case 2 and Case 3 have worse performance than Case 1, because each node in Case 2 and Case 3 has total or partial non-uniform information. Further, Case 2 shows the worst convergence performance. The compared results illustrate that non-i.i.d data distribution still remains MFL convergence but decreases MFL convergence performance.

7.2.3 Effect of γ

We evaluate the impact of γ on the convergence rate of loss function. In this simulation, we still set aggregation frequency $\tau = 4$.

The experimental results are shown in Fig. 7. Subfigure (a) and (b) show that how different values of γ affect the convergence curves of loss function and testing accuracy, respectively. We can see that if $\gamma = 0$, the loss function and accuracy curves of MFL overlap with the corresponding ones of FL because MFL is equivalent to FL for $\gamma = 0$. When γ increases from 0 to 0.9, we can see the convergence rates on both loss function curves and accuracy curves also gradually increase. Subfigure (c) shows the change of final loss function value ($T = 1000$) with $0 < \gamma < 1$. From this subfigure, we can find the final loss function values of MFL are always smaller than FL with $0 < \gamma < 1$. Compared with

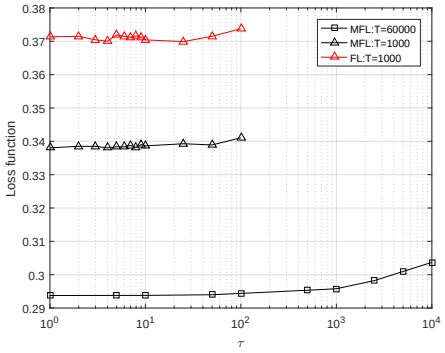


Fig. 8: Loss function values with τ

FL, convergence performance of MFL is improved. This is because $\frac{2(1-\beta\eta)\cos\theta}{\beta\eta p^2} > 1$ and according to Proposition 3, the accelerated convergence range of MFL is $0 < \gamma < 1$. We can see that when $0 < \gamma < 0.95$, the loss function values decrease monotonically with γ so the convergence rate of MFL increases with γ. While $\gamma > 0.95$, the loss function values of MFL start to increase with a gradual deterioration of MFL convergence performance, and in this situation, MFL can not remain convergence. If the γ values are chosen to be close to 1, best around 0.9, MFL reaches the optimal convergence rate.

7.2.4 Effect of τ

Finally, we evaluate the effect of different τ on loss function of MFL. We record the final loss function values with τ based on the three cases of $T = 1,000$ for FL, $T = 1,000$ for MFL and $T = 60,000$ for MFL. We set $\gamma = 0.5$ for MFL. The curves for the three cases are presented in Fig. 8. Comparing FL with MFL for $T = 1,000$, we see that the final loss function values of MFL are smaller than those of FL for any τ. As declared in Proposition 3, under a small magnitude of T and $\eta = 0.002$ which is close to 0, MFL always converges much faster than FL. Further, for $T = 1,000$, the effect of τ on convergence is slight because the curves of FL and MFL are relatively plain. This can be explained by Lemma 3, where $\frac{1}{2\eta\varphi T}$ and $\frac{1}{2\omega\alpha T}$ dominate the convergence upper-bound when the magnitude of T is small. While $T = 60,000$, change of τ affects convergence significantly and the final loss function values gradually increase with τ. As the cases of $T = 1,000$ for MFL and FL, the case of $T = 60,000$ for MFL has a slight effect on convergence if $\tau < 100$. But if $\tau > 100$, MFL convergence performance is getting worse with τ. According to the above analysis of τ, setting an appropriate aggregation frequency will reduce convergence performance slightly with a decline of communication cost (in our cases, $\tau = 100$).

7.2.5 MFL communication efficiency

In the experiment, we evaluate the communication efficiency of MFL compared with FL under different values of γ. Because the momentum and weight are transmitted between the edge nodes and the central server, we can simply assume that the communication size of MFL is twice that of FL due to the additional momentum parameters and the same dimension of momentum and weight. Therefore,

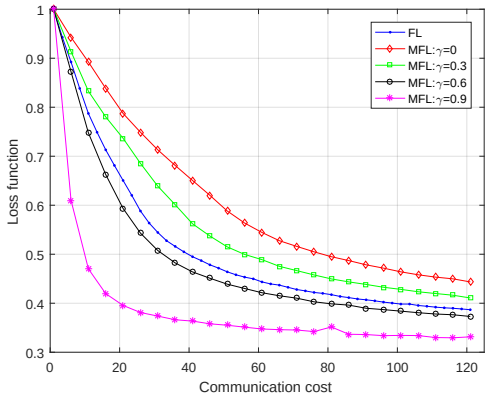


Fig. 9: Comparison of communication efficiency between MFL and FL

we set that communication budget of one global aggregation for MFL is 1, so communication budget of one global aggregation for FL is 0.5. The experiment is based on SVM and we set aggregation frequency $\tau = 4$. The experiment consumes 125 communication budgets for MFL or FL, so MFL performs 125 global aggregations and FL performs 250 global aggregations.

The experimental results are shown in Fig. 9. We see that the communication efficiency is improved with increasing γ, because γ affects the convergence rate of MFL significantly as presented in Section 7.2.3. We can see that MFL with a large value of γ performs better performance than FL and a small γ results to worse performance based on the same communication cost. For example, if $\gamma = 0.6$, MFL has a better convergence with respect to communication cost than FL. Thus, MFL shows higher communication efficiency compared with FL for $\gamma = 0.6$.

8 Conclusion

In this paper, we have proposed MFL which performs MGD in local update step to solve the distributed machine learning problem. Firstly, we have established global convergence properties of MFL and derived an upper bound on MFL convergence rate. This theoretical upper bound shows that the sequence generated by MFL linearly converges to the global optimum point under certain conditions. Then, compared with FL, MFL provides accelerated convergence performance under the given conditions as presented in Proposition 3. Finally, based on MNIST and CIFAR-10 datasets, our simulation results have verified the MFL convergence and confirmed the accelerated convergence of MFL.

Acknowledgments

This work is supported by the National Key Research and Development Program of China under Grant 2018YFA0701603, and the National Natural Science Foundation of China under Grant 61722114.

References

[1] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proc. ICCV*, pages 2722–2730, 2015.

[2] Li Deng, Jinyu Li, Jui-Ting Huang, Kaisheng Yao, Dong Yu, Frank Seide, Michael Seltzer, Geoff Zweig, Xiaodong He, Jason Williams, et al. Recent advances in deep learning for speech research at microsoft. In *Proc. ICASSP*, pages 8604–8608. IEEE, 2013.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. NIPS*, pages 1097–1105, 2012.

[4] Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017.

[5] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[6] Ramanan Subramanian and Faramarz Fekri. Sleep scheduling and lifetime maximization in sensor networks: fundamental limits and optimal solutions. In *Proc. International Conf. Inf. Process. Sensor Netw.*, pages 218–225. IEEE, 2006.

[7] Gerhard P Fettweis. The tactile internet: Applications and challenges. *IEEE Veh. Technol. Mag.*, 9(1):64–70, 2014.

[8] Milan Patel, Brian Naughton, Caroline Chan, Nurit Sprecher, Sadayuki Abeta, Adrian Neal, et al. Mobile-edge computing introductory technical white paper. *White paper, mobile-edge computing (MEC) industry initiative*, pages 1089–7801, 2014.

[9] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutorials*, 19(4):2322–2358, 2017.

[10] Miao Hu, Lei Zhuang, Di Wu, Yipeng Zhou, Xu Chen, and Liang Xiao. Learning driven computation offloading for asymmetrically informed edge computing. *IEEE Trans. Parallel Distrib. Syst.*, 2019.

[11] Xinhou Wang, Kezhi Wang, Song Wu, Sheng Di, Hai Jin, Kun Yang, and Shumao Ou. Dynamic resource scheduling in mobile edge cloud with cloud radio access network. *IEEE Trans. Parallel Distrib. Syst.*, 29(11):2429–2445, 2018.

[12] Nguyen Quoc Viet Hung, Hoyoung Jeung, and Karl Aberer. An evaluation of model-based approaches to sensor data compression. *IEEE Trans. Knowl. Data Eng.*, 25(11):2434–2447, 2012.

[13] Sheng Di, Dingwen Tao, Xin Liang, and Franck Cappello. Efficient lossy compression for scientific data based on pointwise relative error bound. *IEEE Trans. Parallel Distrib. Syst.*, 30(2):331–345, 2018.

[14] Sheng Di and Franck Cappello. Optimization of error-bounded lossy compression for hard-to-compress hpc data. *IEEE Trans. Parallel Distrib. Syst.*, 29(1):129–143, 2017.

[15] Gang Yang, Vincent YF Tan, Chin Keong Ho, See Ho Ting, and Yong Liang Guan. Wireless compressive sensing for energy harvesting sensor nodes. *IEEE Trans. Signal Process.*, 61(18):4491–4505, 2013.

[16] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learn-

ing of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[17] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.*, 37(6):1205–1221, 2019.

[18] 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.

[19] Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. Distributed deep learning on edge-devices: feasibility via adaptive compression. In *16 th IEEE International Symposium on Network Computing and Applications (NCA)*, pages 1–8. IEEE, 2017.

[20] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, pages 1175–1191. ACM, 2017.

[21] Takayuki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. *arXiv preprint arXiv:1804.08333*, 2018.

[22] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[23] Arkadii Semenovich Nemirovsky and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

[24] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Comput. Math. Math. Phys.*, 4(5):1–17, 1964.

[25] Euhanna Ghadimi, Hamid Reza Feyzmahdavian, and Mikael Johansson. Global convergence of the heavy-ball method for convex optimization. In *2015 European Control Conf.*, pages 310–315. IEEE, 2015.

[26] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. *arXiv preprint arXiv:1910.00643*, 2019.

[27] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization. *arXiv preprint arXiv:1905.03817*, 2019.

[28] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *NIPS*, pages 315–323, 2013.

[29] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Netw.*, 12(1):145–151, 1999.

[30] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

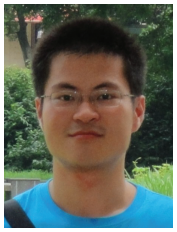
[31] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[32] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

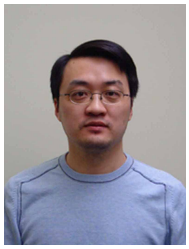
[33] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

Wei Liu received the B.E. degree in electronic information engineering from University of Science and Technology of China, Hefei, China, in 2018. He is currently pursuing the M.E. degree with the Department of Electronic Engineering and Information Science, University of Science and Technology of China. His research interests include distributed machine learning and accelerated computation.



Li Chen received the B.E. in electrical and information engineering from Harbin Institute of Technology, Harbin, China, in 2009 and the Ph.D. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, in 2014. He is currently a faculty member with the Department of Electronic Engineering and Information Science, University of Science and Technology of China. His research interests include wireless IoT communications and wireless optical communications.



Yunfei Chen (S'02-M'06-SM'10) received his B.E. and M.E. degrees in electronics engineering from Shanghai Jiaotong University, Shanghai, P.R.China, in 1998 and 2001, respectively. He received his Ph.D. degree from the University of Alberta in 2006. He is currently working as an Associate Professor at the University of Warwick, U.K. His research interests include wireless communications, cognitive radios, wireless relaying and energy harvesting.

Wenyi Zhang (S'00-M'07-SM'11) is currently a professor with the Department of Electronic Engineering and Information Science, University of Science and Technology of China. He received his bachelor's degree in automation from Tsinghua University in 2001, and his master's and Ph.D. degrees in electrical engineering from University of Notre Dame, in 2003 and 2006, respectively. He was affiliated with the Communication Science Institute, University of Southern California, as a Post-Doctoral Research Associate, and with Qualcomm Incorporated, Corporate Research and Development. His research interest includes wireless communications and networking, information theory, and statistical signal processing. He was an editor for IEEE Communications Letters, and is currently an editor for IEEE Transactions on Wireless Communications.