Efficient Multi-Layer Stochastic Gradient Descent Algorithm for Federated Learning in E-health

Chong Yu[1], Shuaiqi Shen[1], Shiqiang Wang[2], Kuan Zhang[1], Hai Zhao[3]
1. Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, USA
2. IBM T. J. Watson Research Center, Yorktown Heights, New York, USA
3. Department of Computer Science and Engineering, Northeastern University, Shenyang, China
Email:{cyu6@huskers.unl.edu, sshen@huskers.unl.edu, wangshiq@us.ibm.com, kuan.zhang@unl.edu, zhaoh@mail.neu.edu.cn}

***Abstract*—E-health systems consist of intelligent devices, medical institutions, edge nodes, and cloud servers to improve healthcare service quality and efficiency. In e-health systems, patients' data are cooperatively collected by their wearable devices and the hospital they have visited, i.e., vertically distributed data. The data on wearable devices share the same feature set but are different in sample spaces, i.e., horizontally partitioned data. Meanwhile, hospitals target various user groups resulting in high data diversity, i.e., non-identically distributed data. These three characteristics cause that existing federated learning frameworks cannot efficiently train models on medical data. Furthermore, model training in e-health is time-sensitive because some diseases mutate very quickly and spread easily, which requires fast convergence of machine learning algorithms. In this paper, we address the problem of how to efficiently and rapidly train global models on e-health data. Specifically, we propose a multi-layer federated learning framework to cope with data that are vertically, horizontally, and non-identically distributed. Moreover, we develop a Multi-Layer Stochastic Gradient Descent (MLSGD) algorithm towards the proposed framework to learn the optimal global model. To improve training efficiency, partial models learned by devices are aggregated on edge nodes before exchanging intermediate results with hospitals. The weight of local models is proportional to local data size when performing global aggregation to balance the impact of local models on the global model. We also prove the convergence of the MLSGD algorithm from a theoretical perspective. The experimental results from the real-world dataset MIMIC-III validate that the proposed algorithm converges fast and achieves desired accuracy.**

***Index Terms*—E-health, multi-layer federated learning, training efficiency, convergence analysis**

I. Introduction

E-health systems allow smart devices, body sensors, medical instruments, and health institutions to cooperatively provide health services with communication and computing technologies [1]. Compared with the traditional "off-line" health system, e-health not only gathers information from clinical cases but also collects data with various wearable devices, such as bracelets and portable blood glucose meters. Due to communication constraints, computational bottlenecks, and privacy concerns, it is challenging to upload all data generated by distributed nodes to a cloud server for centralized processing [2]. Fortunately, edge computing emerges as a decentralized computing paradigm that offloads computation tasks to edge nodes for fully utilizing computing resources and saving communication cost. Learning on edge, such as federated learning, can also enhance privacy in the applications of e-health [3]. Federated learning enables multiple users that keep local data to jointly train machine learning algorithms over the entire dataset without sharing the raw data.

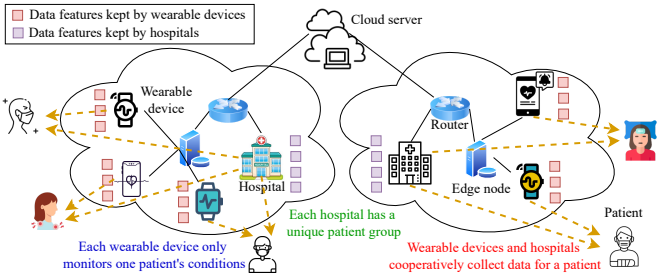

Fig. 1. E-health system architecture

Federated learning can be categorized into three types according to different data distribution patterns. Firstly, horizontal federated learning can deal with the horizontal data partitioning scenarios where each client shares the same feature subset but is different in samples [4]. Utilizing horizontal federated learning, each client independently learns a local model based on its own data, and all clients transmit their model parameters to the cloud server for aggregating a global model [5]. Secondly, vertically federated learning can handle the vertical data partitioning scenarios where each client records different subsets of features for the same sample space [6]. In this circumstance, multiple clients collaboratively learn a local model by communicating intermediate results between clients [7]. Thirdly, cross-silo federated learning [8] aims to cope with several data distribution cases including horizontal, vertical data partitioning, or both. In this setting, clients with the same features conduct horizontal federated learning, and those with the same samples conduct vertically federated learning.

However, existing works still face challenges when applied in e-health systems. Consider an e-health system where edge nodes, hospitals, and the cloud server are interconnected via routers, and wearable devices are managed by edge nodes, as illustrated in Fig. 1. Data distribution in e-health has three characteristics: i) Patients' wearable devices and the hospital they have visited cooperatively collect and store data, i.e., data are vertically distributed across wearable devices and hospitals. This is because patients not only conduct self-monitoring but

also visit hospitals for further physical examinations, e.g., X-Ray or MRI. ii) All wearable devices collect the same health information such as heart rate, and each wearable device only monitors one patient's conditions, i.e., data are horizontally partitioned across wearable devices of different patients. iii) Both the size and distribution of data vary heavily between different hospitals because each hospital is responsible for data collection for its unique patient group, i.e., data are non-identically distributed across different hospitals.

On one hand, data distribution pattern is more complex in e-health so that existing frameworks cannot effectively train models on medical data. Since many existing federated learning [5], [6] consider either horizontally or vertically distributed data, they cannot be directly applied to datasets with both horizontal and vertical data partitioning. Meanwhile, although cross-silo federated learning [9] is effective in training horizontally and vertically partitioned data simultaneously, it does not consider the non-identically distributed feature of medical data. It equally treats all local models when performing aggregation. However, the local model obtained by clients with a larger number of samples should have a higher weight to balance the impact of local models on the global model when data are non-identically distributed.

On the other hand, model training in e-health is time-sensitive and requires higher training efficiency. Several diseases such as COVID-19 mutate very quickly and spread easily. Rapid and accurate model training through existing cases is conducive to the diagnosis and control of such diseases. This requires that training algorithms should converge quickly while effectively processing e-health data.

In this paper, we aim to achieve efficient and rapid global model learning when training e-health data. The main contributions of this paper are summarized as follows.

- We propose a multi-layer federated learning framework with one intermediate result exchange and two aggregation phases for e-health systems to cope with both vertically and horizontally partitioned data while considering non-identically distributed data.
- Based on the proposed framework, we develop an efficient Multi-Layer Stochastic Gradient Descent (MLSGD) algorithm to train the optimal global model. For enhancing training efficiency, the local partial model aggregation is performed on edge nodes before exchanging intermediate results with hospitals. Furthermore, to balance the impact of local models on the global model, the weight of local models is proportional to local data size when performing global aggregation. From a theoretical perspective, we prove that the proposed algorithm converges.
- We conduct extensive experiments to validate the performance of the proposed MLSGD algorithm using real-world health datasets. The results confirm that our proposed algorithm can effectively train medical data and rapidly converge compared with gradient descent algorithm based on conventional machine learning frameworks. In addition, the influence of local partial model and global model aggregation intervals are evaluated.

II. Related Works

Existing works study federated learning frameworks and algorithms for adapting federated learning to various Internet of Things application scenarios.

Horizontal federated learning is one of the most frequently used frameworks, which enables the server to generate a global model by aggregating all local models without sharing local data [10]. Based on horizontal federated learning, Choudhury et al. [11] proposed a framework with two-level privacy protection that can learn a global model from distributed health data kept locally at different clients. Moreover, Wu et al. [12] proposed a cloud-edge based federated learning framework for in-home health monitoring and a generative convolutional autoencoder to mitigate the influence of non-independent-and-identically-distributed data on model training. These works are effective when utilized in simple cases where clients hold the full feature set of samples but are not applicable to complex e-health systems because they do not have the corresponding strategy to cope with the vertical data partitioning.

Vertical federated learning was proposed to coordinate multiple clients with partial data features to jointly train a model [6]. To enable clients to independently conduct stochastic gradient algorithms, Chen et al. [13] proposed an asynchronous learning strategy for vertical federated learning. Although these works are mature in dealing with vertically partitioned data, they cannot handle the additional horizontally distributed data. For the scenario where horizontally and vertically distributed data co-exist, Das et al. [9] proposed a federated learning framework for multi-tier networks. However, it still faces challenges when utilized in e-health systems because of the following reasons. On one hand, it considers a symmetric network structure so that each silo performs the same operation, while device and hospital sides are asymmetric in e-health. On the other hand, cross-silo federated learning ignores the non-identically distributed feature of data and equally treats all local models when performing global aggregation, resulting in a low convergence rate.

In contrast to existing works, we focus on designing a special federated learning framework for e-health systems with a three-tier asymmetric network structure, i.e., server, hospital or edge, and device tiers. Based on this framework, we intend to propose an effective and efficient federated learning algorithm for training medical data.

III. Multi-Layer federated learning

A. System model

We consider an e-health system consisting of one cloud server and M units, where each unit includes $K_m, m = 1, ..., M$ wearable devices and a hospital. A patient's wearable device and the hospital where the patient has visited cooperatively collect the patient's data. Specifically, a portion of data is collected by patients using wearable devices, such as heart rate monitors and sleep trackers in long-term, while the rest of the data is collected by hospitals according to patients' visits and lab results. We assume that

a patient only corresponds to one hospital. A dataset with K samples, $\{\mathbf{X}^{(i)}, y^{(i)}\}_{i=1}^{K}$, is maintained by M units. Each unit m holds a subset D_m of the data with K_m samples: $\{\mathbf{X}^{(m,i)}, y^{(m,i)}\}_{i=1}^{K_m}$, where $\mathbf{X}^{(m,i)}$ denotes the feature vector of the i-th sample, and $y^{(m,i)}$ represents the target value. The dataset is vertically partitioned between the wearable devices and the hospital within a unit. For a single sample $\mathbf{X}^{(m,i)} := [(\mathbf{X}_1^{(m,i)})^{\mathrm{T}}, (\mathbf{X}_2^{(m,i)})^{\mathrm{T}}]^{\mathrm{T}}$, the wearable device i maintains features $\mathbf{X}_1^{(m,i)}$, and the hospital holds $\mathbf{X}_2^{(m,i)}$. Since a device only collects information for one user, the data on the device is only related to one sample. A hospital is responsible for all visiting patients, so that the data on the hospital is relevant to all K_m samples, i.e., the hospital in unit m keeps $\{\mathbf{X}_2^{(m,i)}\}_{i=1}^{K_m}$. Noting that target value $y^{(m,i)}$ is kept by both wearable devices and hospitals.

B. Multi-layer federated learning framework

As shown in Fig. 2, the proposed multi-layer federated learning framework contains three phases: i) vertical federated learning, ii) local horizontal federated learning, and iii) global horizontal federated learning. The details of each phase are given as follows.

Vertical federated learning: Due to separated feature collection, neither wearable devices nor hospitals have the entire dataset. Transmitting data collected by devices to the hospitals for centralized model training is infeasible because raw data transmission may divulge private information of patients. To solve this issue, we apply vertical federated learning to achieve model training on such distributed dataset. Wearable devices and hospitals calculate intermediate results and send the results to each other. Utilizing the intermediate results, they can update their model by computing partial derivatives, while keeping the raw data private. The details of intermediate results are given in Section IV.

Local horizontal federated learning: Allowing all devices to conduct model training with the hospital separately may degrade training efficiency because the hospital needs to learn a unique model for each device. Furthermore, the number of samples at each device in the considered e-health scenario is limited, leading to overfitting. To enhance training efficiency, we can utilize horizontal federated learning among devices so that the hospital only needs to train one model that corresponds to the aggregated model on the device side. Meanwhile, overfitting can be avoided by applying horizontal federated learning because the sample size increases. In addition, involving all devices in the model aggregation process may result in high communication cost and long latency. To solve these issues, a subset of devices is randomly selected to participate in the local horizontal federated learning process [14].

Global horizontal federated learning: As the datasets held by units are determined by the type of hospitals and the number of patients, sample size and data distribution may vary heavily between different units. The non-identically distributed datasets may cause that the local model fits well for their data but is not applicable to the entire dataset [10]. Our goal is to find a universal model that can obtain higher accuracy when applying to entire data. To realize this aim, global horizontal federated learning is applied among units.

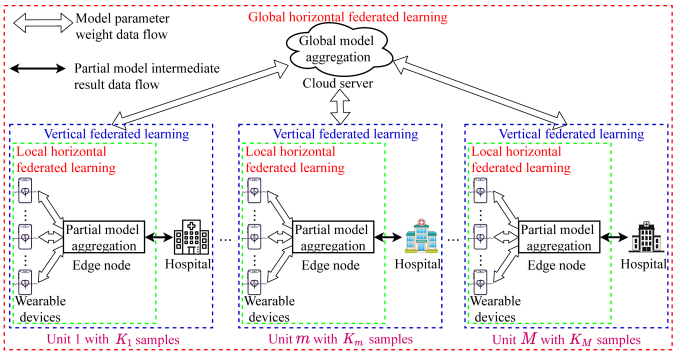

Fig. 2. Multi-layer federated learning framework

C. Problem Formulation

Since each wearable device monitors user conditions to generate one sample, the sample size equals the number of devices in each unit. The loss function for unit m is

$$F_m(\boldsymbol{\theta}_m) = \frac{1}{K_m}\sum_{i=1}^{K_m} f(\boldsymbol{\theta}_m; \mathbf{X}^{(m,i)}, y^{(m,i)}) + \sum_{j=1}^{2} r(\boldsymbol{\theta}_{j,m}), \tag{1}$$

where $r(\cdot)$ is a regularizer, and $\boldsymbol{\theta}_m := [(\boldsymbol{\theta}_{1,m})^{\mathrm{T}}, (\boldsymbol{\theta}_{2,m})^{\mathrm{T}}]^{\mathrm{T}}$ concatenates the partial model learned by wearable devices and hospitals to constitute the local model of unit m. The device side partial model $\boldsymbol{\theta}_{1,m}$ is obtained by aggregating $\boldsymbol{\theta}_{1,m,n}$ that is trained on the n-th device in the m-th unit, i.e.,

$$\boldsymbol{\theta}_{1,m} = \frac{1}{|\mathcal{A}_m|}\sum_{n\in\mathcal{A}_m} \boldsymbol{\theta}_{1,m,n}, \tag{2}$$

where $\mathcal{A}_m$ denotes a randomly selected device subset in unit m for device side partial model training. The cardinality of the set is $|\mathcal{A}_m|$, which is assumed to be proportional to K_m, i.e., $|\mathcal{A}_m| = \alpha K_m$, where K_m is not only the number of devices but also the number of data samples. The global model can be defined as

$$\tilde{\boldsymbol{\theta}} := [(\tilde{\boldsymbol{\theta}}_1)^{\mathrm{T}}, (\tilde{\boldsymbol{\theta}}_2)^{\mathrm{T}}]^{\mathrm{T}},$$
$$\text{with } \tilde{\boldsymbol{\theta}}_1 = \frac{1}{K}\sum_{m=1}^{M} K_m\boldsymbol{\theta}_{1,m}\ ,\ \tilde{\boldsymbol{\theta}}_2 = \frac{1}{K}\sum_{m=1}^{M} K_m\boldsymbol{\theta}_{2,m}, \tag{3}$$

where $K = \sum_{m=1}^{M} K_m$ is the total number of samples. Noting that the global model parameter $\tilde{\boldsymbol{\theta}}$ is only calculated and observable to nodes every P iterations, but we define it for all t to facilitate the analysis later. The global loss function is

$$F(\tilde{\boldsymbol{\theta}}) = \frac{1}{K}\sum_{m=1}^{M} K_m F_m(\tilde{\boldsymbol{\theta}}), \tag{4}$$

which measures how well the model fits the entire datasets.

IV. Multi-Layer Stochastic Gradient Descent Algorithm

In this section, we propose a Multi-Layer Stochastic Gradient Descent (MLSGD) algorithm based on the multi-layer federated learning framework, as shown in Algorithm 1.

Algorithm 1: MLSGD Algorithm

Input: η, Q, P

Output: Global model $\tilde{\boldsymbol{\theta}}^t := [(\tilde{\boldsymbol{\theta}}_1^t)^{\mathrm{T}}, (\tilde{\boldsymbol{\theta}}_2^t)^{\mathrm{T}}]^{\mathrm{T}}$.

Initialize $\boldsymbol{\theta}_{1,m,n}^0 = \tilde{\boldsymbol{\theta}}_1^0$, $\boldsymbol{\theta}_{2,m}^0 = \tilde{\boldsymbol{\theta}}_2^0$, $\forall\ m, n$;
for $t = 0, \ldots, T$ **do**
  if $t\ (mod\ P) = 0$ **then**
    Server computes $\tilde{\boldsymbol{\theta}}_1^t = \frac{1}{K}\sum_{m=1}^{M} K_m \boldsymbol{\theta}_{1,m}^t$, $\tilde{\boldsymbol{\theta}}_2^t = \frac{1}{K}\sum_{m=1}^{M} K_m \boldsymbol{\theta}_{2,m}^t$;
    for $m = 1, \ldots, M$ **do**
      for $n = 1, \ldots, K_m$ **do**
        $\boldsymbol{\theta}_{1,m,n}^t = \tilde{\boldsymbol{\theta}}_1^t$;
      end
      $\boldsymbol{\theta}_{2,m}^t = \tilde{\boldsymbol{\theta}}_2^t$;
    end
  end
  if $t\ (mod\ Q) = 0$ **then**
    for $m = 1, \ldots, M$ **do**
      Randomly selected Q subsets $\{\mathcal{A}_m^\tau\}_{\tau=t}^{t+Q-1}$ with the mini-batches $\{D_{\mathcal{A}_m^\tau}\}_{\tau=t}^{t+Q-1}$;
      Compute $\boldsymbol{\theta}_{1,m}^t = \frac{1}{|\mathcal{A}_m^t|}\sum_{n\in\mathcal{A}_m^t} \boldsymbol{\theta}_{1,m,n}^t$;
      for $n = 1, \ldots, K_m$ **do**
        $\boldsymbol{\theta}_{1,m,n}^t = \boldsymbol{\theta}_{1,m}^t$;
        if $n \in \{\mathcal{A}_m^\tau\}_{\tau=t}^{t+Q-1}$ **then**
          Send $\{\boldsymbol{\zeta}_{1,m,n}^t\}_{\tau=t}^{t+Q-1}$ to edge nodes
        end
      end
      Edge nodes and hospitals exchange intermediate results $\{\mathcal{Z}_1^\tau\}_{\tau=t}^{t+Q-1}$ and $\{\mathcal{Z}_2^\tau\}_{\tau=t}^{t+Q-1}$;
      Edge nodes transmit $\{\mathcal{Z}_2^\tau\}_{\tau=t}^{t+Q-1}$ to devices;
      for $n \in \{\mathcal{A}_m^\tau\}_{\tau=t}^{t+Q-1}$ **do**
        Extract the information corresponding to its own samples from $\{\mathcal{Z}_2^\tau\}_{\tau=t}^{t+Q-1}$;
      end
    end
  end
  for $m = 1, \ldots, M$ **do**
    for $n \in \mathcal{A}_m^t$ **do**
      $\boldsymbol{\theta}_{1,m,n}^{t+1} = \boldsymbol{\theta}_{1,m,n}^t - \eta\nabla_{(1)}F_m(\boldsymbol{\theta}_{1,m,n}^t, \boldsymbol{\theta}_{2,m}^{t_0}; D_{m,n})$;
    end
    $\boldsymbol{\theta}_{2,m}^{t+1} = \boldsymbol{\theta}_{2,m}^t - \eta\nabla_{(2)}F_m(\boldsymbol{\theta}_{1,m}^{t_0}, \boldsymbol{\theta}_{2,m}^t; D_{\mathcal{A}_m^t})$;
  end
end

In initialization stage, the server generates a model containing two parts, i.e., $\tilde{\boldsymbol{\theta}}_1^0$ and $\tilde{\boldsymbol{\theta}}_2^0$. The part $\tilde{\boldsymbol{\theta}}_1^0$, which is related to device collected data, is sent to devices in the system, and the part $\tilde{\boldsymbol{\theta}}_2^0$, which is related to hospital collected data, is transferred to all hospitals, and neither devices nor hospitals have the entire model. The cloud server conducts global model aggregation, i.e, it aggregates local models $[(\boldsymbol{\theta}_{1,m})^{\mathrm{T}}, (\boldsymbol{\theta}_{2,m})^{\mathrm{T}}]^{\mathrm{T}}$ to calculate the global model $[(\tilde{\boldsymbol{\theta}}_1)^{\mathrm{T}}, (\tilde{\boldsymbol{\theta}}_2)^{\mathrm{T}}]^{\mathrm{T}}$ based on (3) every P iterations, where P is a positive integer. Once the global model is generated, the two elements in the aggregated model are transmitted back and replace the original models on devices and hospitals, respectively.

In iteration 0, and every Q-th iteration thereafter, two operations are conducted. On one hand, devices within a unit perform local partial model aggregation. Specifically, since each device has a small sample size, learning models on such a dataset may cause the overfitting problem. To solve the issue, the device side local partial models in each unit, i.e., $\boldsymbol{\theta}_{1,m,n}^t, n \in \mathcal{A}_m^t$, are collected and aggregated by the edge node based on (2) every Q iterations, where $\frac{P}{Q}$ is a positive integer. Since each unit may contain thousands of devices, involving all devices in local partial model aggregation may degrade model training efficiency. The edge node and hospital within unit m agree on Q device subsets $\{\mathcal{A}_m^\tau\}_{\tau=t}^{t+Q-1}$, and the corresponding mini-batches are $\{D_{\mathcal{A}_m^\tau}\}_{\tau=t}^{t+Q-1}$. The mini-batch $D_{\mathcal{A}_m^\tau}$ consists of $D_{m,n},\ \forall n \in \mathcal{A}_m^\tau$, where $D_{m,n} = \{\mathbf{X}_1^{(m,n)}, \mathbf{X}_2^{(m,n)}; y^{(m,n)}\}$ is the sample on the n-th device on m unit. Only the device in the subset $\mathcal{A}_m^\tau$ can participate in local aggregation. Then the edge node transmits the local aggregation results to all devices within the same unit.

On the other hand, devices and their corresponding hospitals exchange intermediate results every Q iterations. Since both devices and hospitals only have the partial of the entire model, they cannot calculate partial derivatives without sharing the intermediate results. The intermediate result on devices for one sample is $\boldsymbol{\zeta}_{1,m,n}^t = f_1(\boldsymbol{\theta}_{1,m}^{t_0}; \mathbf{X}_1^{(m,n)})$, where $f_1(\cdot)$ denotes the machine learning model such as LSTM on devices, and t_0 is the last iteration that satisfies $t_0\ (mod\ Q) = 0$. Similarly, the intermediate result on hospitals for one sample is $\boldsymbol{\zeta}_{2,m,n}^t = f_2(\boldsymbol{\theta}_{2,m}^{t_0}; \mathbf{X}_2^{(m,n)})$, where $f_2(\cdot)$ represents the machine learning model on hospitals. Generally, $f_1(\cdot)$ and $f_2(\cdot)$ are the same. In every Q round, devices send $\{\boldsymbol{\zeta}_{1,m,n}^\tau\}_{\tau=t}^{t+Q-1}$ to edge nodes. Then edge nodes stack all $\{\boldsymbol{\zeta}_{1,m,n}^\tau\}_{\tau=t}^{t+Q-1}$ to form Q intermediate result sets $\{\mathcal{Z}_1^\tau\}_{\tau=t}^{t+Q-1} := \{\boldsymbol{\zeta}_{1,m,n}^\tau\}_{n\in\mathcal{A}_m^\tau}, t \le \tau \le t+Q-1$ and communicate these sets to the corresponding hospital. Conversely, hospitals transfer $\{\mathcal{Z}_2^\tau\}_{\tau=t}^{t+Q-1} := \{\boldsymbol{\zeta}_{2,m,n}^\tau\}_{n\in\mathcal{A}_m^\tau}, t \le \tau \le t+Q-1$ to the edge node within the same unit. After that, edge nodes forward $\{\mathcal{Z}_2^\tau\}_{\tau=t}^{t+Q-1}$ to device $n, n \in \{\mathcal{A}_m^\tau\}_{\tau=t}^{t+Q-1}$ and devices extract the information corresponding to their own samples. Once receiving the required intermediate result, both devices and hospitals updated their local models by utilizing gradient descent. We use η to represent the learning rate. The partial derivatives of the F_m with respect to $\boldsymbol{\theta}_{1,m,n}^t$ is $\nabla_{(1)}F_m(\boldsymbol{\theta}_{1,m,n}^t, \boldsymbol{\zeta}_{2,m,n}^t; D_{m,n})$. For ease of understanding, we can rewrite the partial derivatives to a function of model parameters, i.e., $\nabla_{(1)}F_m(\boldsymbol{\theta}_{1,m,n}^t, \boldsymbol{\theta}_{2,m}^{t_0}; D_{m,n})$. The partial model trained by devices is updated by

$$\boldsymbol{\theta}_{1,m,n}^{t+1} = \boldsymbol{\theta}_{1,m,n}^t - \eta\nabla_{(1)}F_m(\boldsymbol{\theta}_{1,m,n}^t, \boldsymbol{\theta}_{2,m}^{t_0}; D_{m,n}). \quad (5)$$

The partial derivatives of the F_m with respect to $\boldsymbol{\theta}_{2,m}^t$

is $\nabla_{(2)}F_m(\boldsymbol{\zeta}^t_{1,m,n}, \boldsymbol{\theta}^t_{2,m}; D_{\mathcal{A}^t_m})$ obtained from a mini-batch $D_{\mathcal{A}^t_m}$. Similarly, the partial derivative is rewritten as $\nabla_{(2)}F_m(\boldsymbol{\theta}^{t_0}_{2,m}, \boldsymbol{\theta}^t_{2,m}; D_{m,n})$. The partial model learned by hospital is updated by

$$\boldsymbol{\theta}^{t+1}_{2,m} = \boldsymbol{\theta}^t_{2,m} - \eta\nabla_{(2)}F_m(\boldsymbol{\theta}^{t_0}_{1,m}, \boldsymbol{\theta}^t_{2,m}; D_{\mathcal{A}^t_m}). \tag{6}$$

Noting that $t_0 \leq t$ is the last iteration that edge nodes and hospitals exchange intermediate results.

V. Convergence Analysis

We present convergence results of the proposed MLSGD algorithm in this section. The following commonly-used assumptions on loss functions are given for facilitating analysis.

Assumption 1. *The gradient $\nabla F_m(\boldsymbol{\theta})$ is ρ-Lipschitz, i.e.,*

$$||\nabla F_m(\boldsymbol{\theta}_1) - \nabla F_m(\boldsymbol{\theta}_2)|| \leq \rho||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2||, \tag{7}$$

and $\nabla_{(i)}F_m(\boldsymbol{\theta})$ is ρ_i-Lipschitz continuous, i.e.,

$$||\nabla_{(i)}F_m(\boldsymbol{\theta}_1) - \nabla_{(i)}F_m(\boldsymbol{\theta}_2)|| \leq \rho_i||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2||. \tag{8}$$

Assumption 2. *The mini-batch stochastic gradient descent is unbiased, i.e.,*

$$\mathbb{E}[\nabla_{(i)}F_m(\boldsymbol{\theta}; D)] = \nabla_{(i)}F_m(\boldsymbol{\theta}), \tag{9}$$

and the variance of stochastic gradients is bounded, i.e.,

$$\mathbb{E}[||\nabla_{(i)}F_m(\boldsymbol{\theta}; D) - \nabla_{(i)}F_m(\boldsymbol{\theta})||^2] \leq \delta_i^2. \tag{10}$$

Assumption 3. *The expected Euclidean norm of $\nabla_{(i)}F_m(\boldsymbol{\theta}; D)$ is uniformly bounded, i.e.,*

$$\mathbb{E}[||\nabla_{(i)}F_m(\boldsymbol{\theta}; D)||^2] \leq \omega^2. \tag{11}$$

According to Algorithm 1, we can compute the global model by

$$\tilde{\boldsymbol{\theta}}^{t+1} = \tilde{\boldsymbol{\theta}}^t - \eta\boldsymbol{G}^t, \tag{12}$$

where $\boldsymbol{G}^t$ denotes the gradient for the global model, and it equals the mean of gradients for all units, i.e.,

$$\begin{aligned}\boldsymbol{G}^t :&= [(\boldsymbol{G}^t{}_{(1)})^{\mathrm{T}}, (\boldsymbol{G}^t{}_{(2)})^{\mathrm{T}}]^{\mathrm{T}} \\ &= \begin{bmatrix} \frac{1}{K}\sum_{m=1}^{M} K_m \boldsymbol{G}^t{}_{(1,m)} \\ \frac{1}{K}\sum_{m=1}^{M} K_m \nabla_{(2)}F_m(\boldsymbol{\theta}^{t_0}_{1,m}, \boldsymbol{\theta}^t_{2,m}; D_{\mathcal{A}^t_m}) \end{bmatrix}, \\ \text{with } \boldsymbol{G}^t{}_{(1,m)} &= \frac{1}{|\mathcal{A}^t_m|}\sum_{n\in\mathcal{A}^t_m} \nabla_{(1)}F_m(\boldsymbol{\theta}^t_{1,m,n}, \boldsymbol{\theta}^{t_0}_{2,m}; D_{m,n}).\end{aligned} \tag{13}$$

Due to the space limitation, we only present the main result of convergence analysis in Theorem 1. The details of convergence analysis can be found in [15]. To simplify the expression, we define that

$$\begin{aligned} C_1 &= K\alpha\eta^2\omega^2\left(2Q^2 + 3\left(P-Q\right)^2 + 2MP^2\right) \\ &\quad + M\eta^2P^2\left(\delta_1^2 + K\alpha\delta_2^2\right), \\ C_2 &= K\eta^2\omega^2\left(\left(P-Q\right)^2 + 2MP^2 + P^2\right) \\ &\quad + M\eta^2P^2\left(\frac{\delta_1^2}{\alpha} + K\delta_2^2\right), \\ C_3 &= \rho\eta M\left(2\omega^2 + \frac{\delta_1^2}{K\alpha} + \delta_2^2\right) + \left(8M+8\right)\omega^2. \end{aligned} \tag{14}$$

Theorem 1. *From Assumptions 1, 2, and 3, we have*

$$\begin{aligned} &\mathbb{E}\left[\frac{1}{T}\sum_{t=0}^{T-1}\left|\left|\nabla F(\tilde{\boldsymbol{\theta}}^t)\right|\right|^2\right] \\ &\leq \frac{2}{T\eta}\left(F(\tilde{\boldsymbol{\theta}}^0) - F^*\right) + \frac{2\rho_1^2}{K\alpha}C_1 + \frac{2\rho_2^2}{K}C_2 + C_3. \end{aligned} \tag{15}$$

where F^* is the lower bound of loss function.

VI. Performance Evaluation

A. Experimental Settings

We evaluate the proposed MLSGD algorithm with MIMIC-III health dataset. The dataset is preprocessed as in [16] to create 14,681 training samples and 3,236 test samples, where each sample has 76 features. We horizontally split the data among $M = 10$ units to follow the non-identical distribution and vertically split the data among devices and hospitals with each party having 36 features. The LSTM model is adopted to conduct the in-hospital mortality prediction task. For comparison, we utilize three baselines: 1) Centralized model training; 2) Devices communicate their data to the corresponding hospital and hospitals perform horizontal federated learning; 3) Devices in a random choosing subset independently conduct vertical federated learning with the corresponding hospital to obtain local models. After that, local models are transmitted to a server for aggregation. Finally, aggregated results are sent back to devices and hospitals for the next model update. For all experiments, the learning rate is $\eta = 0.01$ and the device selection rate for baseline 3 and the proposed algorithm is $\alpha = 0.02$. To measure training time, we consider that devices communicate with edge nodes and hospitals via mobile Internet, where download and upload speeds are 110 Mbps and 14 Mbps [17], respectively. Meanwhile, edge nodes, hospitals, and the cloud server communicate via fixed broadband with 204 Mbps download speed and 74 Mbps upload speed [17]. Take the proposed algorithm as an example to illustrate how to calculate the training time. For each global aggregation, systems conduct once global aggregation, $\frac{P}{Q}$ times local aggregation, $\frac{P}{Q}$ times intermediate result exchange, and P times local computation. The training time for each global aggregation is $t = t_g + \frac{P}{Q}(t_l + t_e) + P \times t_c$, where t_g, t_l, and t_e denote communication time for global aggregation, local aggregation, and intermediate result exchange, respectively. The local computation time of each iteration t_c is obtained from experiments.

B. Experimental results

We validate the convergence of the proposed MLSGD algorithm. Fig. 3 shows how Area Under the Curve of the Receiver Operating Characteristics curve (AUC of ROC) changes as the number of iterations varies. The proposed MLSGD algorithm and baseline 2 converge when iteration reaches 1500, while baselines 1 and 3 do not converge. The results indicate that our proposed algorithm can converge within limited iterations and outperform baselines 1 and 3. In addition, we utilize the change of AUC of ROC with time to evaluate the training

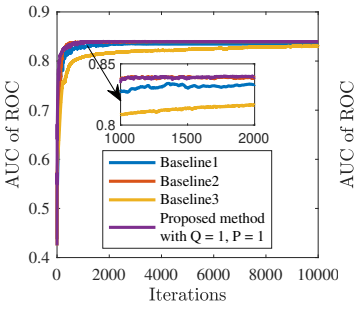

Fig. 3. AUC of ROC vs iterations.

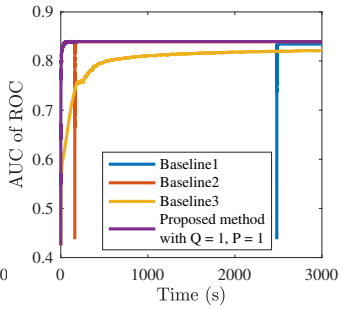

Fig. 4. AUC of ROC vs time.

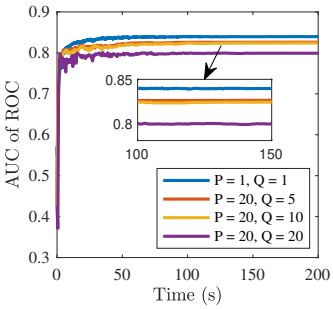

Fig. 5. AUC of ROC vs Q.

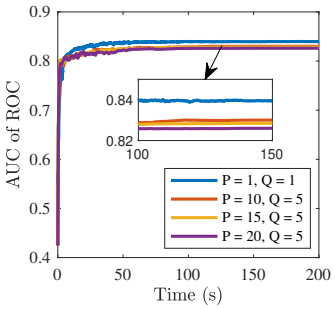

Fig. 6. AUC of ROC vs P.

efficiency of our proposed MLSGD algorithm, as shown in Fig. 4. Baseline 1 starts model training when time is about $2500\,\mathrm{s}$ because it requires transmitting entire raw data to a cloud server before conducting model learning. Similarly, the AUC of ROC of baseline 2 starts to increase when time is about $160\,\mathrm{s}$ because devices should first communicate their data to the corresponding hospital. Moreover, since baseline 3 and the proposed MLSGD algorithm only transmit model parameters which is much smaller than raw data, they start model training quicker than baselines 1 and 2. In addition, the proposed MLSGD algorithm converges when time is about $100\,\mathrm{s}$, while baseline 3 converges when time is about $3000\,\mathrm{s}$. The results show that our proposed MLSGD algorithm is more training efficiency than baselines.

The effect of local partial model aggregation interval (or intermediate result exchange interval) Q on the convergence of our proposed MLSGD algorithm is shown in Fig. 5. As Q increases, the convergence value of AUC of ROC decreases. The result reveals that choosing a small Q, which means more frequent local aggregation and intermediate result exchange, can contribute to a more accurate global model when P is fixed. Fig. 6 shows how the global model aggregation interval P influences the convergence of our proposed MLSGD algorithm. A larger P leads to a lower convergence value of AUC of ROC. This demonstrates that the accuracy of the global model degrades when the frequency of global aggregation decreases, i.e., as P grows. Based on these results, we can select a small Q and P to train a model with higher accuracy when ignoring communication resource restrictions.

VII. Conclusion

In this paper, we have proposed a novel multi-layer federated learning framework that enables the implementation of distributed learning in e-health systems where data are both vertically and horizontally partitioned and follow the non-identical distribution. Based on the multi-layer federated learning framework, we have developed an efficient MLSGD algorithm to minimize the global loss function for finding the optimal global model. We have analyzed the theoretical convergence result of the proposed algorithm. The experiments validate that the MLSGD algorithm can achieve rapid convergence while guaranteeing accuracy. In the future, we will investigate how to choose the optimal subset of devices to further accelerate training, as well as how to achieve the tradeoff between communication overhead and accuracy considering device computation capabilities and resource constraints.

Acknowledgment

This work was supported by Department of Agriculture and National Institute of Food and Agriculture under Grants No. 2021-67021-34417 and 2021-67021-34412.

References

[1] W. Yao, K. Zhang, and C. Yu *et al.*, "Exploiting ensemble learning for edge-assisted anomaly detection scheme in e-healthcare system," in *2021 GLOBECOM*, Madrid, Spain, Dec. 2021, pp. 1–7.

[2] O. Choudhury, A. Gkoulalas-Divanis, and T. Salonidis *et al.*, "Edge computing for internet of things: a survey, e-healthcare case study and future direction," *arXiv preprint arXiv:1910.02578*, 2019.

[3] J. Xu, B. Glicksberg, and C. Su *et al.*, "Federated learning for healthcare informatics," *J. Healthc. Inform. Res.*, vol. 5, no. 1, pp. 1–19, 2021.

[4] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1310–1322, 2020.

[5] Q. Li, X. Wei, and H. Lin *et al.*, "Inspecting the running process of horizontal federated learning via visual analytics," *IEEE Trans. Vis. Comput. Graphics.*, to appear.

[6] Q. Yang, Y. Liu, and T. Chen *et al.*, "Federated machine learning: concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[7] S. Abdulrahman, H. Tout, and H. Ould-Slimane *et al.*, "A survey on federated learning: the journey from centralized to distributed on-site learning and beyond," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5476–5497, 2021.

[8] P. Kairouz, H. McMahan, and B. Avent *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[9] A. Das, S. Wang, and S. Patterson, "Cross-silo federated learning for multi-tier networks with vertical and horizontal data partitioning," *arXiv preprint arXiv:2108.08930*, 2021.

[10] S. Shen, C. Yu, and K. Zhang *et al.*, "Communication-efficient federated learning for connected vehicles with constrained resources," in *2021 IWCMC*, Harbin, China, Jun. 2021, pp. 1636–1641.

[11] O. Choudhury, A. Gkoulalas-Divanis, and T. Salonidis *et al.*, "Differential privacy-enabled federated learning for sensitive health data," *arXiv preprint arXiv:1910.02578*, 2019.

[12] Q. Wu, X. Chen, and Z. Zhou *et al.*, "Fedhome: cloud-edge based personalized federated learning for in-home health monitoring," *IEEE Trans. Mob. Comput.*, to appear.

[13] T. Chen, X. Jin, and Y. Sun *et al.*, "Vafl: a method of vertical asynchronous federated learning," *arXiv preprint arXiv:2007.06081*, 2020.

[14] H. McMahan, E. Moore, and D. Ramage *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *2107 AISTATS*, Fort Lauderdale, USA, Apr. 2017, pp. 1273–1282.

[15] Convergence analysis of multi-layer stochastic gradient descent algorithm for federated learning in e-health. [Online]. Available: https://drive.google.com/file/d/1H7HkJeNEoGZzq61p_bdlZpbyVGl6Vyui/view

[16] H. Harutyunyan, H. Khachatrian, and D. Kale *et al.*, "Multitask learning and benchmarking with clinical time series data," *Sci. Data*, vol. 6, no. 1, pp. 1–18, 2019.

[17] United States's mobile and fixed broadband internet speeds. [Online]. Available: https://www.speedtest.net/global-index/united-states#mobile