FedMes: Speeding Up Federated Learning With Multiple Edge Servers

Dong-Jun Han, *Student Member, IEEE*, Minseok Choi, *Member, IEEE*, Jungwuk Park, and Jaekyun Moon, *Fellow, IEEE*

***Abstract*— We consider federated learning (FL) with multiple wireless edge servers having their own local coverage. We focus on speeding up training in this increasingly practical setup. Our key idea is to utilize the clients located in the overlapping coverage areas among adjacent edge servers (ESs); in the model-downloading stage, the clients in the overlapping areas receive multiple models from different ESs, take the average of the received models, and then update the averaged model with their local data. These clients send their updated model to multiple ESs by broadcasting, which acts as bridges for sharing the trained models between servers. Even when some ESs are given biased datasets within their coverage regions, their training processes can be assisted by adjacent servers through the clients in their overlapping regions. As a result, the proposed scheme does not require costly communications with the central cloud server (located at the higher tier of edge servers) for model synchronization, significantly reducing the overall training time compared to the conventional cloud-based FL systems. Extensive experimental results show remarkable performance gains of our scheme compared to existing methods. Our design targets latency-sensitive applications where edge-based FL is essential, e.g., when a number of connected cars/drones must cooperate (via FL) to quickly adapt to dynamically changing environments.**

***Index Terms*— Federated learning, edge server.**

I. Introduction

With the explosive growth in the numbers of smart phones, wearable devices and Internet of Things (IoT) sensors, a large portion of data generated nowadays is collected outside the cloud, especially at the distributed end-devices at the edge. Federated learning (FL) [1]–[9] is a recent paradigm for this setup, which enables training of a machine learning model in a distributed network while significantly resolving privacy concerns of the individual clients. However, training requires repeated downloading and uploading of the models between the parameter server (PS) and clients, presenting significant challenges in terms of 1) the communication bottleneck at the PS and 2) the non-IID (independent, identically distributed) data characteristic across clients [10]–[14].

In FL, the PS can be located at the cloud or at the edge (e.g., small base stations). Most current studies on FL consider the former, with the assumption that millions of clients are within the coverage of the PS at the cloud; at every global round, the clients in the system should repeatedly communicate with the PS (located at the cloud) for downloading and uploading the models. However, an inherent limitation of this cloud-based system is the long distance between the client and the cloud server, which causes significant propagation delay during model downloading/uploading stages in FL [15], [16]. Specifically, it is reported in [15] that the supportable latency of cloud-based systems is larger than 100 milliseconds, while the edge-based systems have supportable latency of less than tens of milliseconds. In order to support latency-sensitive applications (e.g., smart cars) or emergency events (e.g., disaster response by drones) by FL, in this paper, we consider a system where the PS is located at the edge, which is far closer to the clients than the cloud [17].

An issue, however, is that although the edge-based FL system can considerably reduce the latency between the PS and the clients, the coverage of an edge server is generally limited in practical systems (e.g., wireless cellular networks); there are an insufficient number of clients within the coverage of an edge server for training a global model with enough accuracy. Accordingly, the limited coverage of a single edge server could include biased datasets and thus could lead to a biased model after training. Thus in practice, performing FL with the clients in a single edge server would result in a significant performance degradation.

A. Main Contributions

To overcome the above practical challenges, we propose FedMes, a novel FL algorithm highly tailored to the environment with multiple edge servers (ESs). Compared to conventional cloud-based FL systems, FedMes does not require costly communication (i.e., large time delay) with the cloud server for model synchronization. Our key idea is to utilize the clients located in the overlapping areas between the coverage of ESs, which are typical in 5G and beyond systems with dense deployment of ESs [18]. In the model-downloading stage, each ES sends the current model to the clients in its coverage area; in this process, the clients in the overlapped region

Manuscript received March 1, 2021; revised August 28, 2021; accepted August 29, 2021. Date of publication October 6, 2021; date of current version November 22, 2021. This work was supported in part by the National Research Foundation of Korea under Grant 2019R1I1A2A02061135, in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) funds from the Ministry of Science and Information and Communication Technology (MSIT) of Korea under Grant 2020-0-00626, and in part by the National Research Foundation of Korea under Grant NRF-2020R1G1A1101164. *(Corresponding author: Minseok Choi.)*

Dong-Jun Han, Jungwuk Park, and Jaekyun Moon are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejoen 34141, South Korea (e-mail: djhan93@kaist.ac.kr; savertm@kaist.ac.kr; jmoon@kaist.edu).

Minseok Choi is with the Department of Telecommunication Engineering, Jeju National University, Jeju 63243, South Korea (e-mail: ejaqmf@jejunu.ac.kr).

Digital Object Identifier 10.1109/JSAC.2021.3118422

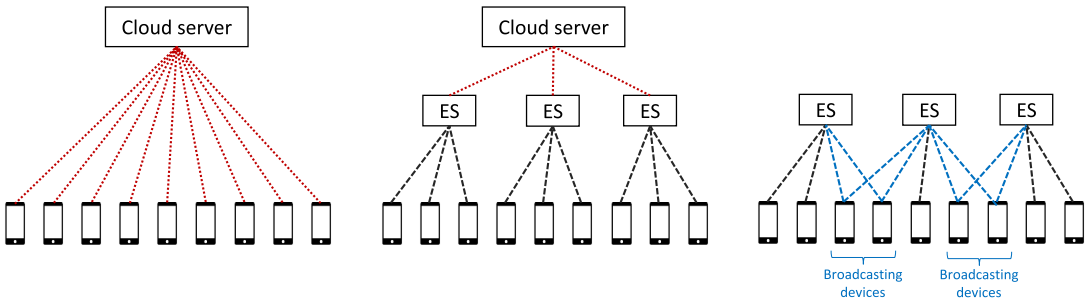


Fig. 1. Basic concept of cloud-based FL, hierarchical FL and the proposed FedMes idea. Compared to existing methods, FedMes does not require costly communication with the central cloud server for model synchronization, significantly reducing the overall training time. In FedMes, the clients in the overlapping areas can act as bridges for sharing the trained models between edge servers (ESs). FedMes targets latency-sensitive applications where edge-based FL is essential, e.g., when a number of cars/drones must cooperate (via FL) to quickly adapt to dynamically changing environments.

receive multiple models from different ESs. These clients in the overlapping area take the average of the received models, and then update the model based on their local data. Then each client sends its updated model to the corresponding ES or ESs, which is aggregated at each ES. A high-level description of FedMes is given in Fig. 1(c).

For example, suppose that client k is located in the non-overlapped region of ES i while client l is in the overlapped region between ES i and ES j. In conventional FL systems, client l participates in the training process of only one of ES i or ES j; on the other hand, in FedMes, client l can act as a bridge for sharing the trained models between both ESs. To be specific, the updated model of client k is averaged only at its associated ES i. In the next step, this averaged model is sent to the clients in its covered area, including client l. After the local model updates at the clients, client l sends its updated model to both ES i and ES j. From this point of view, even when some training samples are only in the coverage of a specific ES, these data can still assist the training process of other servers. Hence, the proposed scheme does not require costly communications with the central cloud server (located at the higher tier of ESs) for model synchronization, significantly reducing the overall training time compared to cloud-based FL systems. Comparing with the scheme which does not consider the overlapping areas, FedMes can provide a significant performance gain especially when the data distributions across coverages of different servers are non-IID, e.g., when a specific server has a biased dataset within its covered area. Our main contributions can be summarized as follows:

- We propose FedMes, an edge-based FL scheme highly tailored to the practical cellular environment with multiple ESs. By utilizing the clients in the overlapped regions between ESs, FedMes enables fast FL.
- We derive the theoretical convergence bound and provide insights on the convergence behavior of FedMes.
- We show via experiments that FedMes provides remarkable performance gain compared to 1) the schemes that require costly communications with the central cloud server for model synchronization and 2) the scheme that does not take the overlapping areas between ESs into account.

B. Related Works

The concept of FL was first proposed in [1]; the authors proposed an algorithm called federated averaging (FedAvg) and confirmed its advantages via experiments on various datasets and various data distribution setups. In [11], [13], [14], the authors propose schemes to reduce the communication overhead between the clients and the server in FL systems; the authors of [13] quantize the model to reduce the communication burden, while the authors of [14] propose federated distillation to achieve the same goal. In [10], the authors propose a data sharing strategy to tackle the non-IID data characteristics across clients. The convergence behavior of FL schemes are studied in [6], [12], and security/privacy issues of FL are addressed in [19]. FL is also being actively studied in communications society, taking wireless channels into account [7]–[9].

Thanks to the recent advent of edge computing, there has been an increased interest in edge-facilitated FL systems [6], [20]–[24]. The authors of [6] focused on optimizing FL framework with a given resource budget in wireless edge networks. The authors of [20] considered resource allocation to minimize energy consumption at the clients in wireless networks. Recently, coded computing schemes are proposed for FL at the edge [24]. However, a single-server setup is considered in [20] and [6], which is totally different from our work leveraging multiple edge servers.

Only a few prior works on FL considered a setup with multiple edge servers [22], [23]. A common idea of these works is to perform hierarchical FL as illustrated in Fig. 1(b). Each ES aggregates the models of clients in its covered area, and the cloud server periodically aggregates the models sent from the ESs. However, the schemes of [22], [23] still require costly communication with the central cloud server for model synchronization, which could significantly slow down the overall training process. Here, if the communication period with the cloud is small, frequent model synchronization between ESs is possible but incurs a large communication time delay. On the other hand, infrequent model synchronization can lead to a bad performance especially with non-IID data distributions across the coverages of edge servers. FedMes overcomes these challenges by enabling fast FL without any

help with the cloud server, i.e., only using the edge servers. Fig. 1 shows the difference between cloud-based FL, hierarchical FL and the proposed FedMes idea. It is shown later in Section V that our method can speed up training compared to the cloud-based FL scheme and the hierarchical FL scheme of [23].

Another line of work focused on fully decentralized FL (or serverless FL) [25]–[28]. The basic idea of decentralized FL approaches is to let the clients to exchange the updated models directly with their neighbors (i.e., clients that are within their communication ranges), without utilizing any servers (neither the cloud server nor the edge server). This is different from our approach leveraging multiple edge servers to enable model synchronization within the cell while allowing the edge servers to share the models by taking advantage of the clients in the overlapped cell regions. The performance of decentralized FL depends on the network topology (i.e., how the clients are connected to each other). When the network is well-connected, decentralized FL can achieve a good convergence rate (i.e., guarantee convergence) but have a large time delay for each round since each client should communicate with a large number of neighbors for model exchange. When the network connection is sparse, the time delay for each round becomes smaller but the convergence is not guaranteed especially in a non-IID data distribution setup. Investigating the advantage of serverless FL compared to the server-based FL schemes (including cloud-based FL, hierarchical FL and the proposed FedMes) is out of scope of the paper.

C. Organizations

The rest of this paper is organized as follows. In Section II, we describe our problem setup with multiple edge servers. The proposed FedMes algorithm is described in Section III and its convergence bound is analyzed in Section IV. We provide experimental results in Section V. Finally, we draw conclusions in Section VI.

II. System Model

A. Background: Federated Learning

Let K be the number of clients in the system. Let n_k be the number of data samples in client k, with $n = \sum_{k=1}^{K} n_k$ being the total number of training samples. We also denote the i-th sample in client k by $x_{k,i}$, for $i \in \{1, 2, \ldots, n_k\}$. Our goal is to solve the following optimization problem

$$\min_{\mathbf{w}} F(\mathbf{w}) = \min_{\mathbf{w}} \sum_{k=1}^{K} \frac{n_k}{n} F_k(\mathbf{w}), \tag{1}$$

where $F_k(\mathbf{w})$ is the local loss function of data samples in client k, written as $F_k(\mathbf{w}) = \frac{1}{n_k} \sum_{i=1}^{n_k} \ell(x_{k,i}; \mathbf{w})$. Now we briefly describe the conventional *cloud-based FedAvg algorithm* in [1], a typical way to solve this problem (see Fig. 1(a)). At step t, each client downloads the current model $\mathbf{w}(t)$ from the PS, which is located at the cloud covering all the clients in the system. Then each client (say, client k) sets $\mathbf{w}_k(t) = \mathbf{w}(t)$ and performs E local updates according to

$$\mathbf{w}_k(t+i+1) = \mathbf{w}_k(t+i) - \eta_{t+i} \nabla F_k(\mathbf{w}_k(t+i), \xi_k(t+i)) \tag{2}$$

for $i = 0, 1, \ldots, E-1$, where η_t is the learning rate at step t and $\xi_k(t)$ is a set of randomly selected data samples from client k at step t. Now each client sends the updated model to the PS, and the PS aggregates the model as $\mathbf{w}(t+E) = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{w}_k(t+E)$. However, full client participation at each aggregation step is impossible in practice and the PS often selects a set $S_{t+E} \subset \{1, 2, \ldots, K\}$, containing the clients that transmit the results to the PS. Thus, we have

$$\mathbf{w}(t+E) = \sum_{k \in S_{t+E}} \frac{n_k}{\sum_{k \in S_{t+E}} n_k} \mathbf{w}_k(t+E). \tag{3}$$

This overall process corresponds to a single global round. This process is repeated until the model achieves the desired performance or some stopping condition is met. According to the algorithm, the model of the k-th client at step $t+1$ is written as

$$\mathbf{w}_k(t+1) = \begin{cases} \mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t)), & \text{if } E \nmid t+1 \\ \sum_{q \in S_{t+1}} \frac{n_q}{N_t} [\mathbf{w}_q(t) - \eta_t \nabla F_q(\mathbf{w}_q(t), \xi_q(t))], & \text{o/w} \end{cases} \tag{4}$$

where $N_t = \sum_{q \in S_{t+1}} n_q$.

B. Problem Setup: Multiple ESs With Overlapping Areas

In contrast with the conventional cloud-based FL systems having a central cloud server covering the whole clients, in this paper we consider L ESs each covering its own local area. We call this local coverage of each edge server a *cell*. Especially with dense deployment of ESs in 5G and beyond networks, there generally exist more than one ES within the range of a specific user that can be reliably communicate with. We call this region in which the client can reliably communicate with multiple ESs *overlapping cell area*.

Let C_i be the set of indices for users located in cell $i \in \{1, 2, \ldots, L\}$. Now define U_i as the set of user indices for the non-overlapped region of cell i, which is the subset of C_i. We also define $V_{i,j}$ as the set of user indices for the overlapping area between cell i and cell j ($i \neq j$ and $V_{i,j} = V_{j,i}$), which is also the subset of C_i. Here, the clients in $V_{i,j}$ can communicate with both ES i and ES j during model download or upload. While we can similarly define overlapped regions with more than two ESs, we consider the case in which the coverage of at most two ESs overlapped for clarity of presentation. Then the coverage of cell i, i.e., C_i can be written as

$$C_i = U_i \cup \Big(\bigcup_{j \in [L] \setminus \{i\}} V_{i,j} \Big) \tag{5}$$

for all $i \in \{1, 2, \ldots, L\}$. The overall coverage of the system can be written as $C = \{1, 2, \ldots, K\} = \cup_{i=1}^{L} U_i \cup (\cup_{i=1}^{L} \cup_{j=i+1}^{L} V_{i,j})$. Note that each ES can communicate with the clients in its covered area only. Our goal is to solve the problem in (1) in this setup, without sharing the learning models at the higher tier of ESs (i.e., the cloud server) during training.

III. Proposed FedMes Algorithm

A. Algorithm Description

Now we describe FedMes, the proposed algorithm tailored to the above setup with multiple edge servers. At the beginning

of step t, each client in cell i downloads the current model $\mathbf{w}^{(i)}(t)$ from ES i and lets $\mathbf{w}_k(t) = \mathbf{w}^{(i)}(t)$ if located in the non-overlapped region, i.e., $k \in U_i$. Client $k \in V_{i,j}$ additionally downloads $\mathbf{w}^{(j)}(t)$ from ES j and combines the two received models according to

$$\mathbf{w}_k(t) = \frac{1}{\sum_{k\in S_t^{(i)}} n_k + \sum_{k\in S_t^{(j)}} n_k} \times \Big(\sum_{k\in S_t^{(i)}} n_k \mathbf{w}^{(i)}(t) + \sum_{k\in S_t^{(j)}} n_k \mathbf{w}^{(j)}(t) \Big), \quad (6)$$

where $S_t^{(i)}$ and $S_t^{(j)}$ are the sets of clients that sent their results to ES i and ES j, respectively, at the previous aggregation step. Here, a larger weight is given to the aggregated model of the ES that utilized more training samples in the previous aggregation step. Hence, client $k \in V_{i,j}$ should receive the value of $\sum_{k\in S_t^{(i)}} n_k$ from ES i and $\sum_{k\in S_t^{(j)}} n_k$ from ES j. If the number of training samples utilized at both ES i and ES j are the same, i.e., $\sum_{k\in S_t^{(i)}} n_k = \sum_{k\in S_t^{(j)}} n_k$, then we can rewrite (6) as

$$\mathbf{w}_k(t) = \frac{1}{2}(\mathbf{w}^{(i)}(t) + \mathbf{w}^{(j)}(t)). \quad (7)$$

We take this assumption for the rest of the paper for ease of presentation. Now every client (say client k) updates the model with its own local data according to equation (2) for E steps to obtain $\mathbf{w}_k(t+E)$. Then, client k sends its updated model $\mathbf{w}_k(t+E)$ to the corresponding ES(s). In particular, the clients in the overlapped region $V_{i,j}$ send their results to both ES i and ES j by *broadcasting*. Here, we note that this does not require additional communication time since only one round of communication is required at each client by the broadcast nature of wireless networks.

Now every ES i collects the locally updated models from the clients in its covered area C_i, to obtain the aggregated result. In this aggregation process, each ES takes the weighted average of the received models, where the weight depends on the location of client k (i.e., whether client k is located in the overlapped region or not). To be specific, let $S_{t+E}^{(i)}$ be the set of indices for clients that transmit the results to ES i. Then, ES i obtains the aggregated result $\mathbf{w}^{(i)}(t+E)$ by taking the weighted average of the received models as

$$\mathbf{w}^{(i)}(t+E) = \sum_{k\in S_{t+E}^{(i)}} \gamma_k^{(i)} \mathbf{w}_k(t+E), \quad (8)$$

where $\gamma_k^{(i)}$ is the normalized parameter that ES i gives weight to client k, depending on its location:

$$\gamma_k^{(i)} = \begin{cases} \alpha_u n_k, & k \in U_i \\ \alpha_v n_k, & k \in V_{i,j} \text{ for any } j \end{cases} \quad \text{and} \quad \sum_{k\in S_{t+E}^{(i)}} \gamma_k^{(i)} = 1. \quad (9)$$

Here, α_u and α_v can be viewed as a design parameter. For example, one can set $\alpha_u < \alpha_v$ to give larger weights to the clients in the overlapped regions that can act as bridges for sharing the trained models between ESs. Note that by setting $\alpha_u = \alpha_v$, the above aggregation rule in (8) at each ES reduces to the conventional FedAvg in [1]. This overall process at the edge servers and the clients is repeated until some stopping condition is met, say at step T. When the overall training is finished, the models are averaged over all ESs to obtain the final global model $\mathbf{w}^f$ as follows:

$$\mathbf{w}^f(T) = \frac{1}{L}\sum_{i=1}^{L} \mathbf{w}^{(i)}(T). \quad (10)$$

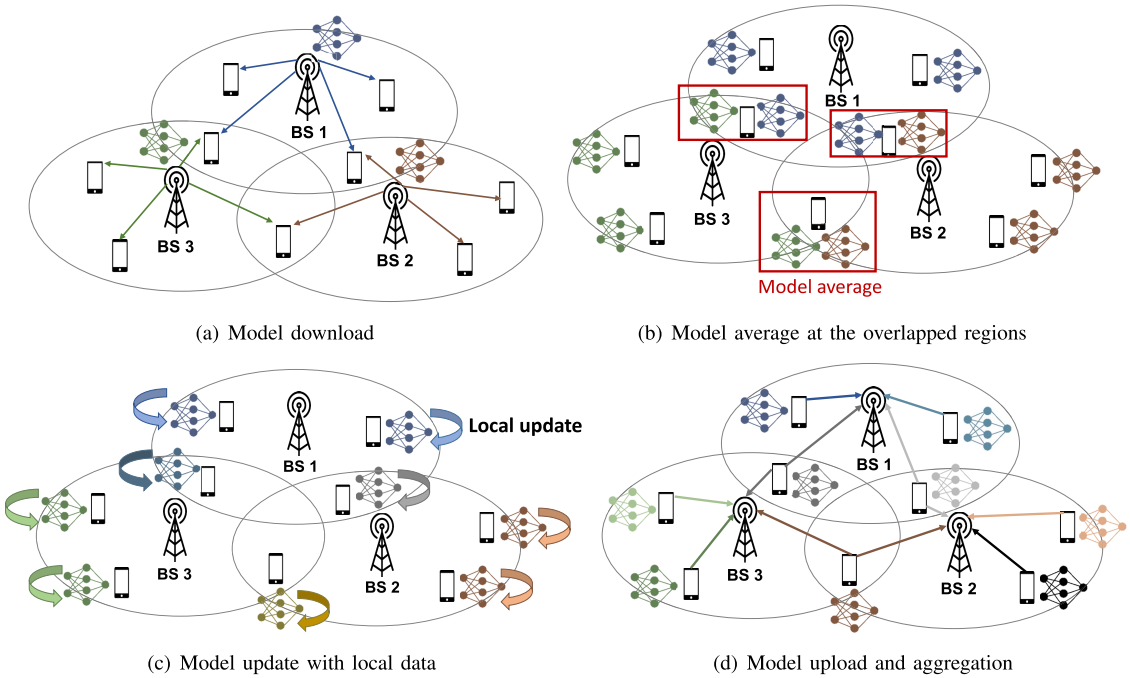
The details of FedMes are summarized in Algorithm 1. Compared to the conventional cloud-based FL systems which require communication with the cloud server (located at the higher tier of ESs) for model synchronization, in FedMes, only the communications between the clients and ESs are required. Fig. 2 shows an example with $L = 3$, $|U_1| = |U_2| = |U_3| = 2$ and $|V_{1,2}| = |V_{2,3}| = |V_{3,1}| = 1$. As can be seen from Figs. 2(b) and 2(c), the unique characteristic of FedMes is that the starting points of models could be different even for the clients in the same cell, due to model average in the overlapped regions. This is totally different from typical FL algorithms in which the clients have the same starting points at the beginning of each global round. From the overall process, it can be also seen that even when some training samples are only in the non-overlapped region of cell i, i.e., U_i, these data

Algorithm 1 Federated Learning With Multiple Edge Servers (FedMes)

Input: Initialized model $\mathbf{w}(0)$, **Output:** Final global model $\mathbf{w}^f(T)$

Set $\mathbf{w}_k(0) = \mathbf{w}(0)$ for all clients $k = 1, 2, \ldots, K$

for each step $t = 0, 1, \ldots, T-1$ **do**
  if $E \nmid t+1$ **then**
    for each client $k = 1, 2, \ldots, K$ in parallel **do**
      $\mathbf{w}_k(t+1) \leftarrow \mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))$ // Local update
    end for
  else
    Each edge server i randomly selects $S_{t+1}^{(i)}$ to receive the updated models from the clients
    for each client $k = 1, 2, \ldots, K$ in parallel **do**
      if $k \in U_i$ for some $i \in \{1, 2, \ldots, L\}$ **then**
        $\mathbf{w}_k(t+1) \leftarrow \sum_{k\in S_{t+1}^{(i)}} \gamma_k^{(i)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))]$ // Edge aggregation
      end if
      if $k \in V_{i,j}$ for $i, j \in \{1, 2, \ldots, L\}$ and $i \neq j$ **then**
        $\mathbf{w}_k(t+1) \leftarrow \frac{1}{2}\Big(\sum_{k\in S_{t+1}^{(i)}} \gamma_k^{(i)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))]$ // Average of two edge aggregations $+ \sum_{k\in S_{t+1}^{(j)}} \gamma_k^{(j)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))] \Big)$
      end if
    end for
  end if
end for
$\mathbf{w}^f(T) \leftarrow \frac{1}{L}\sum_{i=1}^{L} \sum_{k\in S_T^{(i)}} \gamma_k \mathbf{w}_k(T)$ // Averaged model of L edge servers



(a) Model download

(b) Model average at the overlapped regions

(c) Model update with local data

(d) Model upload and aggregation

Fig. 2. Procedure of proposed FedMes algorithm.

can still assist the training process of other cells. We show later in Section V that our scheme with multiple starting points performs well by sufficiently reflecting the data samples that are not in the coverage of a specific ES. By utilizing the weight parameters γ_k in (9), the model of client $k \in C_i$ at step $t+1$ can be written as follows:

$$\mathbf{w}_k(t+1) = \begin{cases} \mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t)), & \text{if } E \nmid t+1 \\ \sum\limits_{k \in S_{t+1}^{(i)}} \gamma_k^{(i)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))], & \\ & \text{if } E \mid t+1 \text{ and } k \in U_i \\ \frac{1}{2}\Big(\sum\limits_{k \in S_{t+1}^{(i)}} \gamma_k^{(i)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))] & \\ + \sum\limits_{k \in S_{t+1}^{(j)}} \gamma_k^{(j)} [\mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t))] \Big), & \\ & \text{if } E \mid t+1 \text{ and } k \in V_{i,j} \end{cases} \tag{11}$$

for $i \in \{1, 2, \ldots, L\}$ and $j \in \{1, 2, \ldots, L\} \setminus \{i\}$.

Note that our FedMes can be easily generalized to cell topologies with multiple ESs (more than two) having overlapping coverage regions, by simply adding equations in (11) for the regions with 3, 4 overlapping ESs or more. A generalized algorithm with more than two overlapping regions is described in Appendix A. We also note that (11) holds regardless of the client's location, which means that (11) is still valid even with moving clients (i.e., mobile devices).

When implementing FedMes in practice, additional information could be required at the ESs depending on the client selection method. If adjacent ESs would like to cooperatively select the clients in their overlapped areas, adjacent ESs should share their client selection policy. However, this is not necessary, because ESs can select their clients independently. On the other hand, from the client's perspective, each device needs to know which ESs are accessible in order to determine whether it should receive the models from multiple ESs. This information can be easily obtained by inquiring connection to ESs with negligible signaling overheads. Especially, the soft handover technique adopted in cellular networks enables even the devices in overlapped areas to obtain these information easily and to communicate with adjacent multiple ESs.

B. Latency Analysis

In this subsection, we compare the latency performance of FedMes with other baseline schemes.

1) FedMes: Assume that the size of the model is M bits. Let c be the number of required CPU cycles to compute 1 bit and f be the number of CPU cycles per second (i.e., computation capacity at each client). We also assume that performing E local updates require computation of $d(E)$ bits at each client. Then, the computation time at each client for E local updates, denoted by t_{comp}, can be written as

$$t_{\text{comp}} = \frac{c \cdot d(E)}{f}. \tag{12}$$

Assume that orthogonal frequency division multiple access (OFDMA) is utilized for uploading clients' local models to their corresponding ESs after E local updates. Let B_u be the uplink bandwidth between the client and the ES. Similarly, we define B_d as the downlink bandwidth that the edge server uses to send the aggregated global model to each client. We also let h_u be the uplink channel gain between the client and the ES, which can be written as $h_u = g_u / d_{\text{edge}}^{\eta}$. Here, g_u, η, d_{edge} denote the fast fading component, pathloss exponent, and the distance between the ES and the client, respectively. We can similarly define the downlink channel gain as $h_d = g_d / d_{\text{edge}}^{\eta}$. Finally, we let p_{client} and p_{edge} be the

transmitter power at the clients and ESs, respectively. Then the communication time for one-round trip between the client and the edge server, t_{edge}, can be written as

$$t_{\text{edge}} = \frac{M}{B_u \log_2(1+\frac{h_u p_{\text{client}}}{B_u N_0})} + \frac{M}{B_d \log_2(1+\frac{h_d p_{\text{edge}}}{B_d N_0})}, \quad (13)$$

where N_0 is the noise spectral density. Here, we note that the uplink communication time $\frac{M}{B_u \log_2(1+\frac{h_u p_{\text{client}}}{B_u N_0})}$ is significantly larger than the downlink communication time $\frac{M}{B_d \log_2(1+\frac{h_d p_{\text{edge}}}{B_d N_0})}$ since the clients generally have much smaller transmit power compared to the server (i.e., $p_{\text{client}} \ll p_{\text{edge}}$). Now the running time of FedMes for a single global round can be written as follows:

$$t_{\text{FedMes}} = t_{\text{comp}} + t_{\text{edge}}. \quad (14)$$

2) Cloud-Based FL: In the cloud-based FL scheme, all clients communicate with the central cloud server for model uploading/downloading. The running time of the cloud-based FL scheme for one global round becomes

$$t_{\text{CloudFL}} = t_{\text{comp}} + t_{\text{cloud}}, \quad (15)$$

where t_{cloud} is the communication time for one-round trip between the client and the cloud server. We can write the channel between the client and the cloud server as $h = g/d_{\text{cloud}}^{\eta}$, where $d_{\text{edge}} \ll d_{\text{cloud}}$ holds. Hence, the uplink communication time of our scheme is significantly smaller than the uplink communication time of the cloud-based FL. Moreover, since the uplink communication time is the dominant factor of the communication time delay, we generally have $t_{\text{edge}} \ll t_{\text{cloud}}$. By comparing (14) and (15), it can be seen that FedMes can significantly improve the latency performance compared to the conventional cloud-based FL scheme.

3) Hierarchical FL: Both the edge servers and the cloud server are utilized in the hierarchical FL scheme of [23]. In the edge aggregation global round, each edge server aggregates the local models sent from the clients in its covered area and distributes the aggregated model to the clients. In the cloud aggregation global round, the cloud server aggregates the updated models from the ESs and distributes across the clients. Suppose that the cloud aggregation is performed periodically at every T_{cloud} global round. Then, the average running time for one global round becomes

$$t_{\text{Hier}} = \frac{1}{T_{\text{cloud}}}[(t_{\text{comp}} + t_{\text{edge}})(T_{\text{cloud}} - 1) + t_{\text{comp}} + t_{\text{cloud}}]. \quad (16)$$

Note that $t_{\text{edge}} \ll t_{\text{cloud}}$ holds. By rewriting the running time of FedMes in (14) as $t_{\text{FedMes}} = \frac{1}{T_{\text{cloud}}}[(t_{\text{comp}}+t_{\text{edge}})(T_{\text{cloud}}-1)+ t_{\text{comp}} + t_{\text{edge}}]$, it can be seen that the latency performance can be significantly improved by using FedMes compared to the hierarchical FL scheme, especially with small T_{cloud}. It can be also seen that the gap decreases with a larger T_{cloud}. These results are confirmed via experiments in Section V.

Remark 1: As stated in the introduction, the latency of decentralized FL (or serverless FL) depends on the network topology (i.e., how the clients are connected to each other). Although analyzing the performance of serverless FL is out of scope of this paper, we provide an example comparing the serverless FL scheme with our FedMes in Appendix B.

IV. Theoretical Results

In this section, we provide the convergence bound on FedMes with the following common assumptions in FL [6], [12], [29], [30] and distributed optimization [31], [32].

Assumption 1: For all $k \in \{1, 2, \ldots, K\}$, $F_k(\mathbf{w})$ is β-smooth with respect to $\mathbf{w}$, i.e., for any $\mathbf{w}$ and $\mathbf{w}'$, $F_k(\mathbf{w}) \leq F_k(\mathbf{w}') + (\mathbf{w} - \mathbf{w}')\nabla F_k(\mathbf{w}') + \frac{\beta}{2}\|\mathbf{w} - \mathbf{w}'\|^2$.

Assumption 2: For all $k \in \{1, 2, \ldots, K\}$, $F_k(\mathbf{w})$ is μ-strongly convex, i.e., for any $\mathbf{w}$ and $\mathbf{w}'$, $F_k(\mathbf{w}) \geq F_k(\mathbf{w}') + (\mathbf{w} - \mathbf{w}')\nabla F_k(\mathbf{w}') + \frac{\mu}{2}\|\mathbf{w} - \mathbf{w}'\|^2$.

Assumption 3: The stochastic gradients in each device is unbiased and its variance is bounded, i.e., $\mathbb{E}_\xi[\nabla F_k(\mathbf{w}, \xi)] = \nabla F_k(\mathbf{w})$ and $\mathbb{E}_\xi[\|\nabla F_k(\mathbf{w}, \xi) - \nabla F_k(\mathbf{w})\|] \leq \sigma_k^2$.

Assumption 4: The expected squared norm of stochastic gradients is uniformly bounded, i.e., $\mathbb{E}_\xi[\|\nabla F_k(\mathbf{w}, \xi)\|^2] \leq G^2$.

All above assumptions hold for *linear/logistic regression models* but Assumption 2 generally does not hold for neural networks with non-convex loss functions. We show later in Section V that the proposed FedMes works well on both neural networks (having non-convex loss functions) and logistic regression models (that surely satisfy all the above assumptions).

For analysis, we assume that all clients in the system participate for each global round. By defining $\mathbf{v}_k(t+1)$ as

$$\mathbf{v}_k(t+1) = \mathbf{w}_k(t) - \eta_t \nabla F_k(\mathbf{w}_k(t), \xi_k(t)), \quad (17)$$

we can rewrite our result in (11) as follows:

$$\mathbf{w}_k(t+1) = \begin{cases} \mathbf{v}_k(t+1), & \text{if } E \nmid t+1 \\ \sum\limits_{k \in C_i} \gamma_k^{(i)} \mathbf{v}_k(t+1), & \text{if } E \mid t+1 \text{ and } k \in U_i \\ \frac{1}{2}\Big(\sum\limits_{k \in C_i} \gamma_k^{(i)} \mathbf{v}_k(t+1) + \sum\limits_{k \in C_j} \gamma_k^{(j)} \mathbf{v}_k(t+1) \Big), & \\ \qquad \text{if } E \mid t+1 \text{ and } k \in V_{i,j} \end{cases} \quad (18)$$

Let $\mathbf{w}^*$ be the optimal solution of $F(\mathbf{w})$ in (1). From the β-smoothness of F, we have

$$F(\mathbf{w}^f(T)) - F(\mathbf{w}^*) \leq (\mathbf{w}^f(T) - \mathbf{w}^*)^T \nabla F(\mathbf{w}^*) + \frac{\beta}{2} \left\|\mathbf{w}^f(T) - \mathbf{w}^*\right\|^2,$$

where $\mathbf{w}^f(T)$ is our final model defined in (10). Now we set $\alpha_u = 2\alpha_v$ and assume that all clients have the same number of data samples. Then by taking expectation at both sides, we have

$$\mathbb{E}\Big[F(\mathbf{w}^f(T))\Big] - F(\mathbf{w}^*) \leq \frac{\beta}{2}\mathbb{E}\Big[\left\|\mathbf{w}^f(T) - \mathbf{w}^*\right\|^2\Big] \quad (19)$$

$$= \frac{\beta}{2}\mathbb{E}\Big[\left\|\bar{\mathbf{v}}(T) - \mathbf{w}^*\right\|^2\Big] \quad (20)$$

where $\bar{\mathbf{v}}(t) = \frac{1}{K}\sum_{k=1}^{K} \mathbf{v}_k(t)$. Here, (20) comes from $\alpha_u = 2\alpha_v$ and assuming that all clients have the same number of data samples. Using same proof technique of [12], we have

the following lemma which shows the upper bound on $\mathbb{E}\Big[\left\|\bar{\mathbf{v}}(T)-\mathbf{w}^*\right\|^2\Big]$ in (20).

Lemma 1: Assume that F_k is β-smooth, μ-strongly convex. If $\eta_T \leq \frac{1}{4\beta}$, we have

$$\begin{aligned}&\mathbb{E}\Big[\left\|\bar{\mathbf{v}}(T+1)-\mathbf{w}^*\right\|^2\Big]\\&\leq (1-\mu\eta_T)\mathbb{E}\Big[\left\|\bar{\mathbf{w}}(T)-\mathbf{w}^*\right\|^2\Big]+\eta_T^2\mathbb{E}\Big[\left\|\bar{\mathbf{g}}(T)-\mathbf{g}(T)\right\|^2\Big]\\&\quad+6\beta\eta_T^2\Gamma+2\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\right\|^2\Big],\end{aligned}$$

where $\bar{\mathbf{w}}(t)=\frac{1}{K}\sum_{k=1}^{K}\mathbf{w}_k(t)$, $\bar{\mathbf{g}}(t)=\frac{1}{K}\sum_{k=1}^{K}\nabla F_k(\mathbf{w}_k(t))$, $\mathbf{g}(t)=\frac{1}{K}\sum_{k=1}^{K}\nabla F_k(\mathbf{w}_k(t),\xi_k(t))$ and $\Gamma=F^*-\frac{1}{K}\sum_{k=1}^{K}F_k^*$. Here, F^* is the minimum value of F and F_k^* is the minimum value of F_k.

Proof: See [12] for the proof. ∎

Now based on Lemma 1, we have the following theorem which provides the convergence bound of FedMes algorithm.

Theorem 1: Suppose Assumptions 1, 2, 3, 4 hold. Assuming $L=3$, consider a cell geometry with $|U_1|=|U_2|=|U_3|=u$ and $|V_{1,2}|=|V_{2,3}|=|V_{3,1}|=v$. By letting $\alpha_u=2\alpha_v$ and selecting the learning rate $\eta_t=\frac{2}{\mu(\gamma+t)}$ for $\gamma=max\{\frac{8\beta}{\mu},E\}$, the convergence bound of FedMes at time step T with full client participation can be written as

$$\begin{aligned}&\mathbb{E}\Big[F(\mathbf{w}^f(T))\Big]-F(\mathbf{w}^*)\\&\leq\frac{\beta}{\gamma+T}\Big(\frac{B}{\mu}+2\beta\left\|\mathbf{w}(0)-\mathbf{w}^*\right\|^2\Big)&(21)\\&\quad+\frac{\beta}{2}Q(T-1)&(22)\end{aligned}$$

where $Q(T)=2\mathbb{E}[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\right\|^2]$ and $B=\frac{1}{K^2}\sum_{k=1}^{K}\sigma_k^2+6\beta\Gamma$.

Proof: See Appendix C. ∎

We can observe two terms from the convergence bound in Theorem 1 above. The first term (21) is in the same form of the bound in [12] with conventional cloud-based FL, and goes to zero as time step T increases. The second term $\frac{\beta}{2}Q(T-1)$ in (22) shows the gap between the model of each client and the averaged model at time step $T-1$. In conventional cloud-based FL scheme where the models of all clients are synchronized at every global round, the second term (22) also goes to zero as T increases, as illustrated in [12]. Similarly, if the clients in the overlapping areas successfully act as bridges for sharing the models between ESs in FedMes, the term in (22) gets close to zero. More specifically, this term goes to zero when the models of all clients converge to the same model. Even if not, the algorithm can approach the optimal solution within a certain error, when the variance of models are bounded. In the next section, we empirically show that FedMes achieves the accuracy of the conventional cloud-based FL scheme at each global round, indicating that the effect of the error term $Q(T)=\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\right\|^2\Big]$ becomes negligible. In the following proposition, we provide a more detailed analysis on this term of (22).

Proposition 1: Assuming $\mathbb{E}[||\mathbf{w}_k(t)||]^2\leq\Omega$ for all k and t, the term $Q(T)$ in (22) can be upper bounded as

$$\begin{aligned}&\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\right\|^2\Big]\\&\leq\frac{4}{(\gamma+T)\mu}\Big(\frac{48K}{L}+24u\Big)(E-1)^2G^2+\beta\Big(\frac{6K}{L}+12u\Big)\Omega.\end{aligned}\tag{23}$$

Proof: See Appendix D. ∎

The first term of (23) goes to zero as T increases. The second term $\beta(\frac{6K}{L}+12u)\Omega$ can be viewed as an error term. Note that this term is an increasing function of u. Since $K=(u+v)L$, i.e., $u=\frac{K}{L}-v$, this error term can be rewritten as a decreasing function of v. This indicates that more devices in the overlapping areas reduces the error term and thus leads to a better convergence.

V. Experimental Results

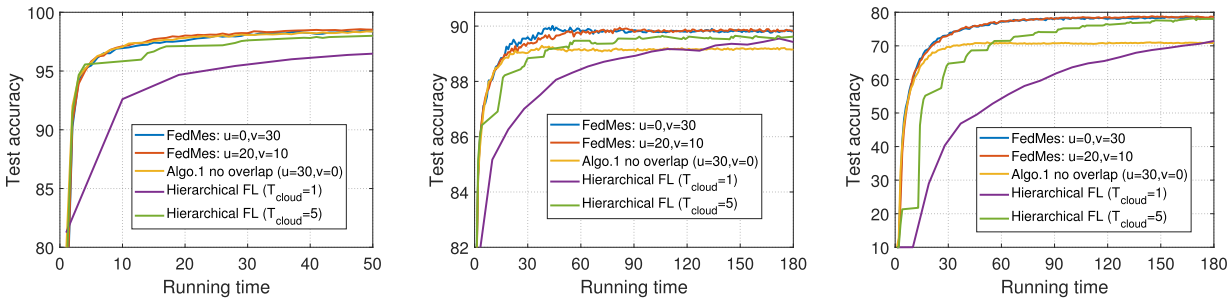
In this section, we provide experimental results for our algorithm using the MNIST [33], FMNIST [34] and CIFAR10 [35] datasets. We split the MNIST and FMNIST datasets into 60,000 training samples and 10,000 test samples, and split the CIFAR10 dataset into 50,000 training samples and 10,000 test samples. For MNIST and FMNIST, we utilized the convolutional neural network (CNN) with 2 convolutional layers and 2 fully connected layers, and 2 convolutional layers and 1 fully connected layer, respectively. For CIFAR10, we utilized VGG-11 for training. These results are provided in Section V-C. In Section V-D, we additionally provide results for training a logistic regression model using MNIST.

A. Comparison Schemes

We compare our FedMes with the following schemes. First, we consider the *hierarchical FL* scheme [23], which utilizes L edge servers and a single cloud server. Here, the cooperation through the clients in the overlapped regions are not considered. In other words, the clients in the overlapped regions communicate with only one ES. After E local updates at each client, each edge server aggregates the models from the clients in its own covered area and then distributes across the clients. After T_{cloud} aggregations at each edge server, the cloud server aggregates the models from the edge servers and distributes across the clients. Here, the case with $T_{\text{cloud}}=1$ can be viewed as a cloud-based FL scheme since communication with the cloud occurs with every E local updates at the clients. Secondly, we consider *Algorithm 1 with no overlap*, where the proposed Algorithm 1 is applied in a setup where the clients in the overlapped regions communicate with only one ES. Each edge server independently performs training without any communication with the cloud, i.e., $T_{\text{cloud}}=\infty$. When training is finished, the model parameters of different cells are averaged to obtain the final global model.

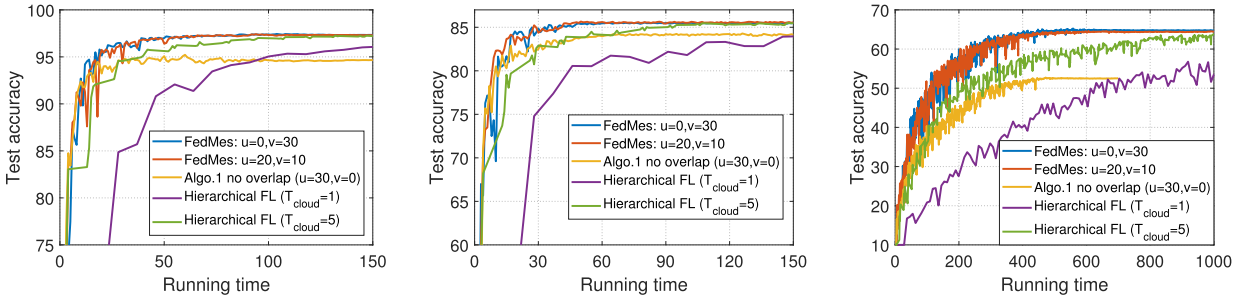
B. Experimental Setup

We consider a setup with $L=3$ edge servers and $K=90$ clients. We also consider a cell geometry with $|U_1|=|U_2|=|U_3|=u$ and $|V_{1,2}|=|V_{2,3}|=|V_{3,1}|=v$, where



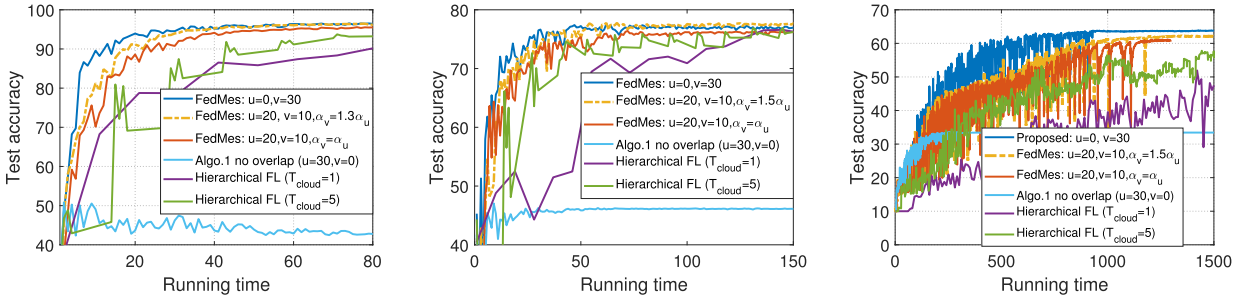
(a) MNIST (b) FMNIST (c) CIFAR10

Fig. 3. Test accuracy versus training time in a (client IID, cell IID) setup.



(a) MNIST (b) FMNIST (c) CIFAR10

Fig. 4. Test accuracy versus training time in a (client non-IID, cell IID) setup.



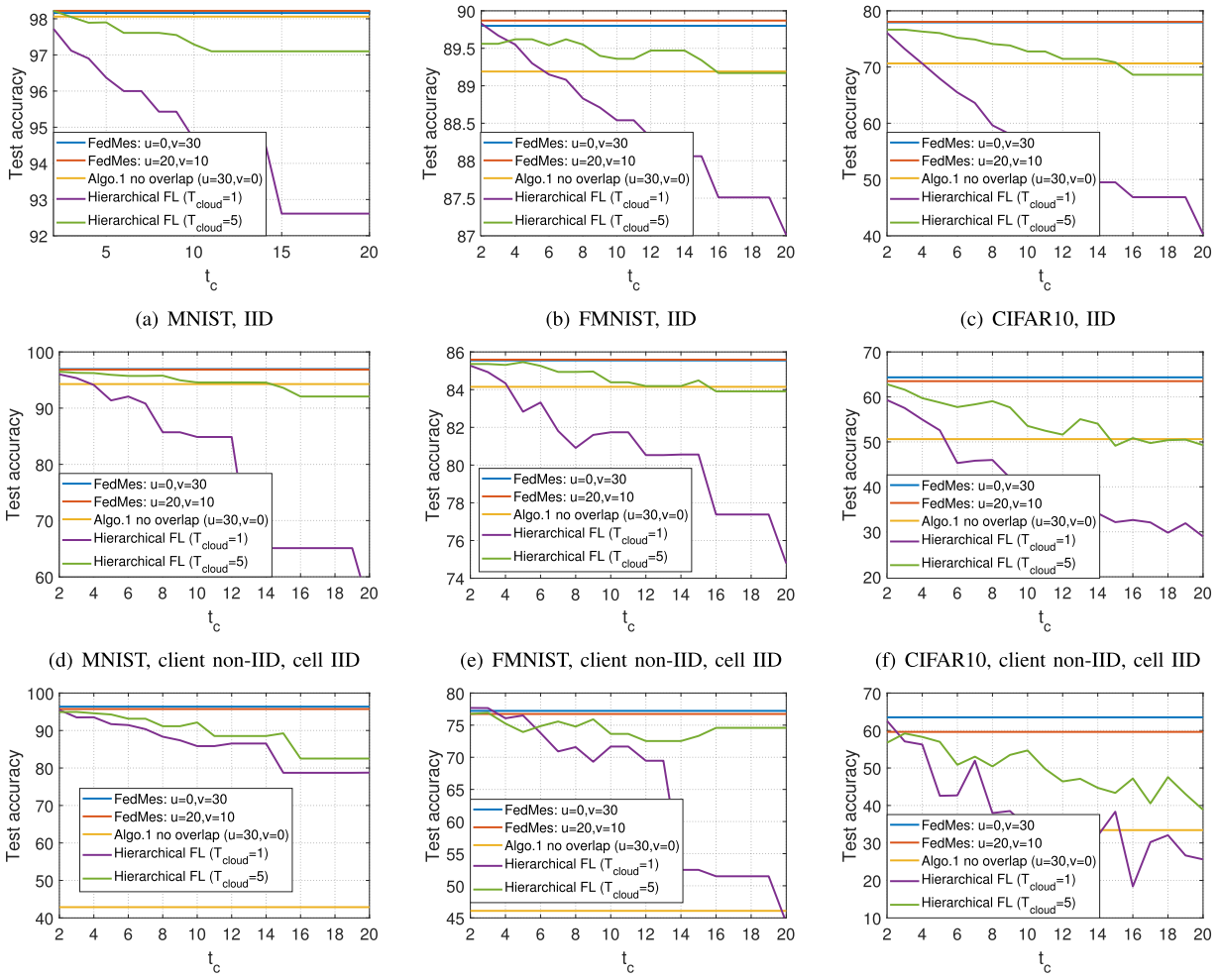
(a) MNIST (b) FMNIST (c) CIFAR10

Fig. 5. Test accuracy versus training time in a (client non-IID, cell non-IID) setup.

$u + v = K/3 = 30$ holds. In the aggregation step at the edge servers, we let each edge server to collect the results of randomly selected $m = 20$ clients in its covered area. Considering $L = 3$ edge servers, a total of 60 clients participate in each aggregation step when there are no overlapping regions ($v = 0$). When $v > 0$, each edge server has access to $u + 2v$ clients. For a fair comparison, we let each edge server to collect the results of $m = 20$ clients, where $\frac{um}{u+2v}$ of them are taken from the non-overlapped region U_i, and $\frac{2vm}{u+2v}$ of them are selected from the two overlapping areas, $\frac{vm}{u+2v}$ of each. Assuming that the adjacent edge servers select the same $\frac{vm}{u+2v}$ clients from the overlapping area by cooperation, a total of $\frac{3(u+v)m}{u+2v} = \frac{60(u+v)}{u+2v}$ clients participates in each aggregation step. Hence, the number of clients participating at each global round is reduced compared to the scheme with no overlap.

We set $t_{\text{edge}}/t_{\text{comp}} = 10$ to capture the communication bottleneck as in [13] and normalize t_{edge} to 1. We define $t_c = t_{\text{cloud}}/t_{\text{edge}}$, where $t_c \gg 1$ holds due to the significant communication delay between the client and the cloud server. We set $t_c = 10$ for experiments as in the setup in [23]. Mini-batch stochastic gradient descent with momentum is utilized for training, with an initial learning rate of 0.01 and a momentum term of 0.9. For all experiments, we set the number of local epochs at each client to 5, and the local mini-batch size to 10.

We consider three data allocation scenarios to test non-IID situations: The first is the (client IID, cell IID) case, where the datasets across K clients and across L cells are IID. Secondly, we consider (client non-IID, cell IID) case where each client randomly selects two classes for data assignment. Hence, the datasets across clients are non-IID but across cells are IID. The last is the (client non-IID, cell non-IID) where the clients in each cell are given with two fixed classes. Hence, the datasets across clients and cells are both non-IID. For the first two scenarios, we only consider the case with $\alpha_u = \alpha_v$ since our scheme with $\alpha_u = \alpha_v$ already shows the optimal performance in these scenarios. For the last non-IID



(a) MNIST, IID (b) FMNIST, IID (c) CIFAR10, IID

(d) MNIST, client non-IID, cell IID (e) FMNIST, client non-IID, cell IID (f) CIFAR10, client non-IID, cell IID

(g) MNIST, client non-IID, cell non-IID (h) FMNIST, client non-IID, cell non-IID (i) CIFAR10, client non-IID, cell non-IID

Fig. 6. Test accuracy versus t_c. The test accuracies are computed at a specific time slot. 1) IID: time 30, 80, 80 for MNIST, FMNIST, CIFAR10. 2) (client non-IID, cell IID): time 40, 80, 400 for MNIST, FMNIST, CIFAR10. 3) client/cell non-IID: time 60, 80, 1000 for MNIST, FMNIST, CIFAR10.

scenario, we consider different (α_u, α_v) scenarios. In addition, we ignored the batch normalization layers of the VGG model during training and testing for the last two non-IID scenarios.

C. Results With Deep Neural Networks

1) Experiments in a (Client IID, Cell IID) Setup: In Fig. 3, we plot the test accuracy versus running time in a (client IID, cell IID) setup. We let $\alpha_u = \alpha_v$ for FedMes. We have the following important observations in this setup. First, the proposed schemes with $v > 0$ have very similar performances. It can be seen that FedMes with $u = 20$, $v = 10$ has almost the same performance as the scheme with all clients in the overlapping areas ($u = 0$, $v = 30$). Comparing FedMes with *Algorithm 1 with no overlap* ($u = 30$, $v = 0$), we have the following results. For the MNIST dataset, Algorithm 1 with no overlap performs almost the same with the proposed schemes, since the dataset is simple enough to achieve the ideal performance by only utilizing the data covered by a specific cell. With the FMNIST dataset, a slight gap between FedMes and Algorithm 1 with no overlap is observed. For a more complicated dataset CIFAR10, we have a larger gap between them, around test accuracy of 8%.

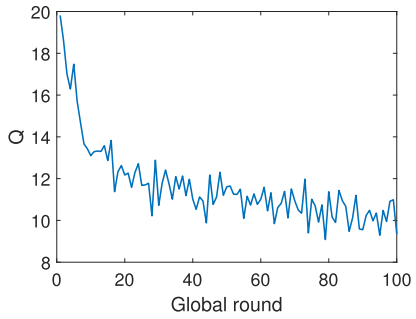
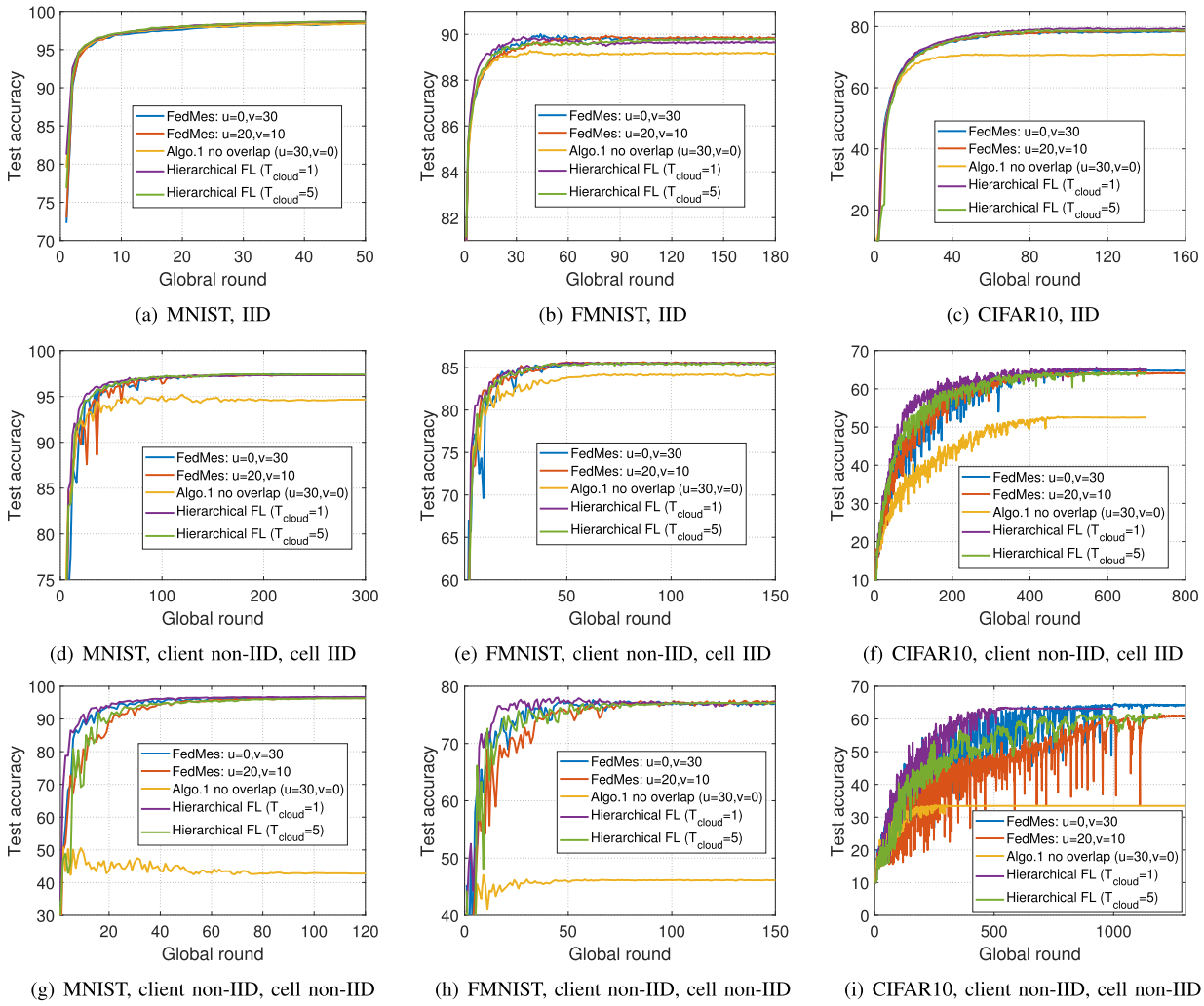


Fig. 7. Error term $Q(T)$ versus global round using MNIST dataset in a (client non-IID, cell non-IID) scenario.

Now we observe the *hierarchical FL* [23] with $T_{\text{cloud}} = 1$, i.e., the cloud-based system where synchronization with the cloud is performed at every global round. For all three datasets, our observation is that FedMes can eventually achieve the best-test accuracy of the cloud-based scheme, which is the ideal performance. However, the proposed schemes do not require costly communication with the central cloud server, significantly reducing the running time to achieve the ideal



(a) MNIST, IID (b) FMNIST, IID (c) CIFAR10, IID

(d) MNIST, client non-IID, cell IID (e) FMNIST, client non-IID, cell IID (f) CIFAR10, client non-IID, cell IID

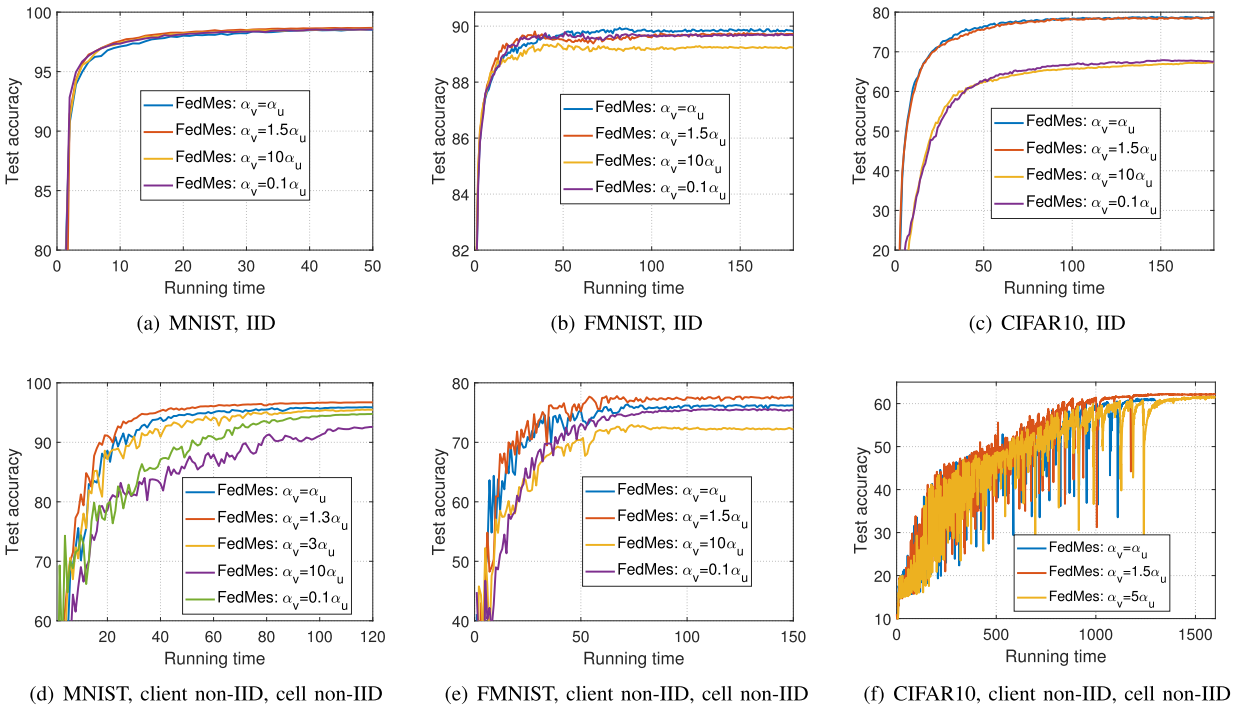
(g) MNIST, client non-IID, cell non-IID (h) FMNIST, client non-IID, cell non-IID (i) CIFAR10, client non-IID, cell non-IID

Fig. 8. Test accuracy versus global round.

test accuracy compared to the cloud-based systems. The hierarchical FL with $T_{\text{cloud}} > 1$ can reduce the period of communication with the cloud, while achieving the ideal performance. Hence, with a properly chosen T_{cloud}, it can be seen that hierarchical FL with $T_{\text{cloud}} > 1$ performs better than the cloud-based scheme with $T_{\text{cloud}} = 1$. However, it can be seen that the hierarchical FL scheme still suffers from communication delay with the cloud. Here, we note that a very large T_{cloud} can reduce the communication time with the cloud in the hierarchical scheme, but as T_{cloud} grows, it gets closer to the comparison scheme *Algorithm 1 with no overlap*. Therefore, significant degradation of the accuracy with complicated datasets is expected when a large T_{cloud} is chosen to reduce the communication time with the cloud. The overall results show remarkable performance gains of FedMes compared to others, especially with on CIFAR10.

2) Experiments in a (Client Non-IID, Cell IID) Setup: Fig. 4 shows the result in a (client non-IID, cell IID) setup. We set $\alpha_u = \alpha_v$ for our schemes. Since the datasets across the clients are non-IID, the performance of each scheme is degraded compared to the case in Fig. 3. We also observe a larger gap between the proposed schemes and Algorithm 1 with no overlap. The trend is consistent with the (client IID, cell IID) case, confirming the advantage of FedMes utilizing the clients in the overlapping cell areas.

3) Experiments in a (Client Non-IID, Cell Non-IID) Setup: Finally, the case with non-IID datasets across both clients and cells is illustrated in Fig. 5. In this non-IID setup, we consider both cases with $\alpha_u = \alpha_v$ and $\alpha_u < \alpha_v$ for FedMes. An interesting observation here is that the test accuracy of hierarchical FL with $T_{\text{cloud}} = 5$ decreases before each aggregation step at the cloud. This is because each edge server is given a biased dataset within its coverage due to the cell non-IIDness, which can degrade the performance before model synchronization at the central cloud. Since *Algorithm 1 with no overlap* does not allow any synchronization at the cloud during training, it has significantly low test accuracy. In FedMes, the clients in the overlapping cell areas enable to share the trained models between edge servers. Therefore, even when some edge servers have biased datasets within their coverage, the model of each edge server can be assisted by various classes that are not within its coverage. Especially in this scenario with non-IID data across cells, it is observed that giving more weights to the clients in the overlapping cell areas (i.e., $\alpha_u < \alpha_v$) can further speed up training than the scheme that gives the same weights to all clients ($\alpha_u = \alpha_v$). The overall results show that FedMes



(a) MNIST, IID (b) FMNIST, IID (c) CIFAR10, IID

(d) MNIST, client non-IID, cell non-IID (e) FMNIST, client non-IID, cell non-IID (f) CIFAR10, client non-IID, cell non-IID

Fig. 9. Effect of varying α_u, α_v on FedMes with $u = 20$, $v = 10$.

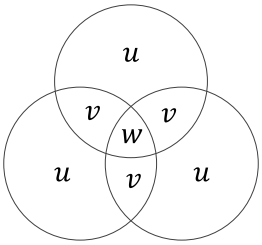


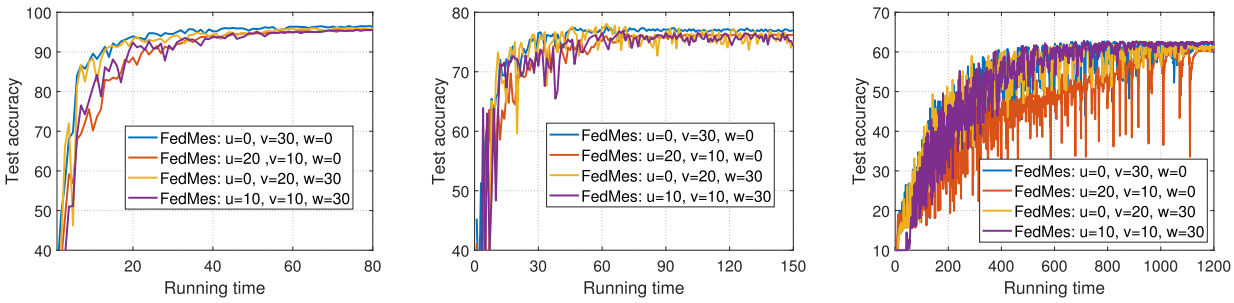
Fig. 10. Number of clients located in each region with $L = 3$ cells.

is still powerful even in this non-IID setup, compared to the schemes that require costly communication with the central cloud server for model synchronization and the scheme that does not take the overlapping areas between edge servers into account.

4) Comparison With Varying t_c: So far, we examined the performance with a fixed $t_c = t_{\text{cloud}}/t_{\text{edge}}$. Now the question is, depending on t_c of the current system, which scheme performs the best and how much performance gain do we have? Fig. 6 provides the result on test accuracy at a specific time as a function of t_c. The hierarchical FL scheme has performance degradation with increasing t_c. Since FedMes does not utilize the cloud, its performance is not affected by t_c. Our observation is that FedMes performs the best even with a small t_c value. Especially in practical regimes where the supportable latency of cloud-based system is much larger than that of edge-based system, FedMes can provide significant advantages compared to other baselines.

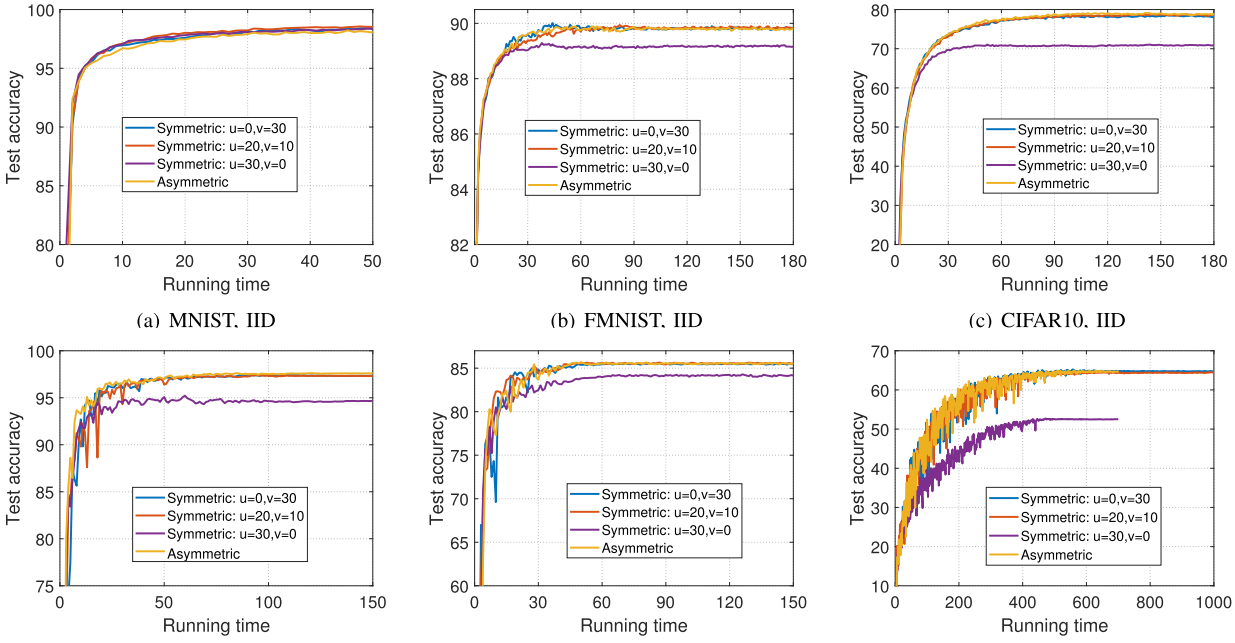
5) Test Accuracy Versus Global Round: Here we first observe the behavior of the error term $Q(T)$ in (22) as a function of global round. As can be seen from Fig. 7, the error term decreases up to a certain constant. To see how this error affects the performance of our scheme, we observe Fig. 8, which provides the accuracy-versus-round plots. The communication time delay is not considered in this plot. It can be seen that FedMes with both ($u = 0$, $v = 30$) and ($u = 20$, $v = 10$) achieve the same final accuracy with the cloud-based system (purple line), except for the case with Fig. 8(i). For CIFAR10 in a (client non-IID, cell non-IID) setup, FedMes with ($u = 0$, $v = 30$) achieves the same final accuracy with the purple line but the case with ($u = 20$, $v = 10$) does not. This indicates that a larger number of clients should be in the overlapping areas to mitigate the effect of the error term, especially when the data distributions across the cells are non-IID and the dataset is relatively complex.

6) Effect of Varying α_u, α_v: We previously showed in Fig. 5 that giving more weights to the clients in the overlapping regions (during aggregation step at the edge servers) can further speed up FedMes, especially in a (client non-IID, cell non-IID) setup. In Fig. 8, which shows the performance of FedMes with $u = 20$, $v = 10$, we provide additional experiments to gain insights on the effects of varying α_u and α_v. We have the following interesting observations. First, if α_v is too small, i.e., if we give too small weights to the clients in the overlapped regions, the effect of clients in the overlapping areas are neglected; the aggregated models of the ESs cannot be shared through the clients in the overlapping areas, degrading the performance of learning. If α_v is too large, i.e., if we give sufficiently large weights to the clients in the overlapped regions, the effect of clients in the non-overlapped regions are neglected which also significantly degrades the performance of the trained model. With an appropriate choice of α_u and α_v in a (client non-IID, cell non-IID) setup, we can speed up training compared to the simple approach that gives the same weights to all clients in the system ($\alpha_u = \alpha_v$). In particular, for MNIST, $\alpha_v = 1.3\alpha_u$ gives the best performance, while



(a) MNIST (b) FMNIST (c) CIFAR10

Fig. 11. Experiments with multiple overlapping ESs in a (client non-IID, cell non-IID) setup.



(a) MNIST, IID (b) FMNIST, IID (c) CIFAR10, IID

(d) MNIST, client non-IID, cell IID (e) FMNIST, client non-IID, cell IID (f) CIFAR10, client non-IID, cell IID

Fig. 12. Performance of FedMes in an asymmetric cell topology.

$\alpha_v = 1.5\alpha_u$ does it for FMNIST and CIFAR10. Compared to the result in a non-IID setup, giving more weights to the clients in the overlapped region does not provide additional performance gain in an IID setup.

7) Experiments With Multiple Overlapping ESs: In the previous plots, we considered a setup where each client is located in the overlapping region of no more than two ESs. Here, we assume that the clients can be in regions with more than two overlapping ESs. Again, we focus on $L = 3$ ESs and define u, v as above. We also introduce w, the number of clients in the overlapped region among all $L = 3$ cells (see Fig. 10). The number of clients in the system can be written as $K = 3(u + v) + w$. Fig. 11 shows the performance of FedMes depending on u, v and w, in a (client non-IID, cell non-IID) setup. Here, we let each ES to give the same weights to all clients regardless of the location. The overall results show that having more clients in the overlapped regions can speed up training. On CIFAR10, when $u > 0$, the case with $w > 0$ can achieve a target accuracy much faster.

8) Experiments in an Asymmetric Cell Topology: In Fig. 12, we provide additional experimental results in an asymmetric cell topology. The number of edge servers are $L = 3$, and we set the number of clients in each region as in Fig. 13. The results are consistent with the plots in the main manuscript, confirming the advantage of FedMes in practical asymmetric cell topology.

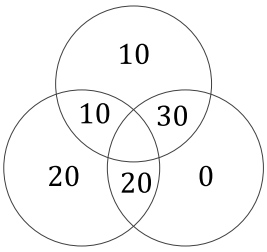
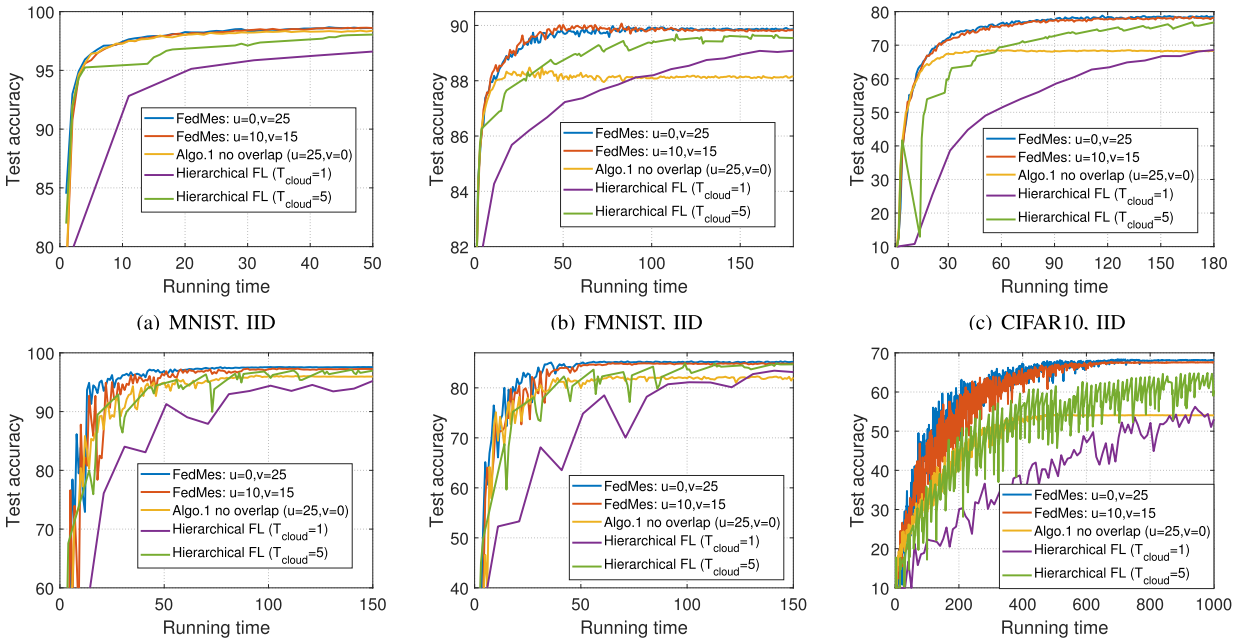


Fig. 13. Asymmetric cell topology for Fig. 12.

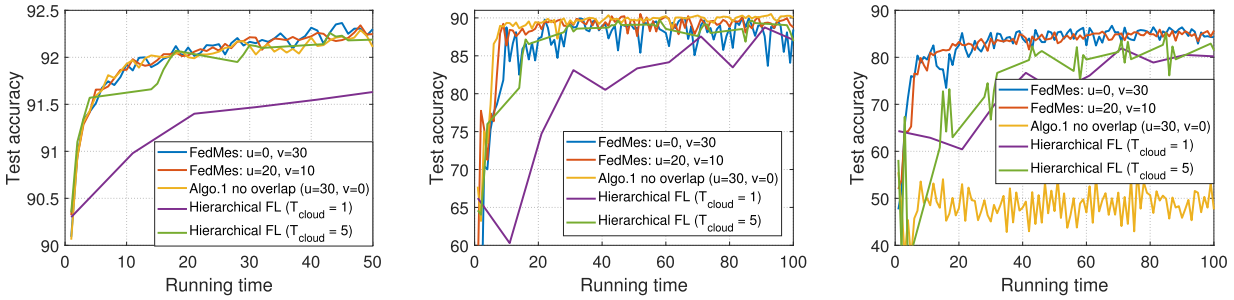
9) Experiments With $L = 4$ Cells: Finally, to confirm the applicability of FedMes further, we consider a scenario with $L = 4$ cells having $K = 100$ clients in the system. We assume $|U_i| = u$ for $i \in \{1, 2, 3, 4\}$ and $|V_{1,2}| = |V_{2,3}| = |V_{3,4}| = |V_{4,1}| = v$, $|V_{1,3}| = |V_{2,4}| = 0$. Fig. 14 shows the test



(a) MNIST, IID (b) FMNIST, IID (c) CIFAR10, IID

(d) MNIST, client non-IID, cell IID (e) FMNIST, client non-IID, cell IID (f) CIFAR10, client non-IID, cell IID

Fig. 14. Performance of FedMes with $L = 4$ cells.



(a) IID (b) Client non-IID, cell IID (c) Client non-IID, cell non-IID

Fig. 15. Test accuracy versus training time for training a logistic regression model using MNIST.

accuracy as a function of running time. The results confirms the advantage of FedMes in practical scenarios with more than three cells.

D. Results With a Logistic Regression Model

In this subsection, we provide experimental results for a logistic regression model which satisfies the convexity assumption of Assumption 2. We focus on the multinomial logistic regression problem using MNIST with 10 classes. Each 2D image with size 28×28 is converted to a vector of size 784 to be the input of the model. Other setups are exactly the same as described in Section IV-B.

Fig. 15 shows the test accuracy versus running time for different data distribution scenarios. As in the previous plots, it can be seen that the cloud-based FL with $T_{\text{cloud}} = 1$ does not perform well due to the large communication time delay between the clients and the cloud server. Other schemes have similar performances in both IID and (client non-IID, cell IID) setups. However, in a (client non-IID, cell non-IID) setup, FedMes again shows significant performance gain compared to others, confirming the advantage of utilizing multiple edge servers and the clients in the overlapping cell areas.

VI. Conclusion

We proposed FedMes, a FL algorithm specifically geared to the practical cellular environment with multiple edge servers. By utilizing the clients located in the overlapping cell areas that act as bridges for sharing the models between edge servers, FedMes enables to speed up training without utilizing the central cloud server. We derived the theoretical convergence bound of FedMes and provided insights on the convergence behavior. Extensive experiments verified the advantage of FedMes compared to existing methods on various datasets with different data distribution setups. Our solution enables to support latency-sensitive applications by speeding up FL in real-world wireless networks 1) having large communication time delay between the clients and the cloud server and 2) having biased datasets within the coverage of each edge server. Extending our result to federated distillation or ES-specific personalization are interesting topics for future research.

APPENDIX A
GENERALIZATION TO THE CASE WITH MORE THAN TWO OVERLAPPING ESS

Our idea can be generalized to overlapping of more than two ESs following the procedure of Algorithm 1. Suppose a specific client k is located in the overlapping area of three ESs: ESs i, j and l. Then, we can simply rewrite the aggregation process of (6) as

$$\begin{aligned}&\mathbf{w}_k(t)\\&=\frac{1}{\sum_{k\in S_t^{(i)}} n_k+\sum_{k\in S_t^{(j)}} n_k+\sum_{k\in S_t^{(l)}} n_k}\\&\quad\times\Big(\sum_{k\in S_t^{(i)}} n_k\mathbf{w}^{(i)}(t)+\sum_{k\in S_t^{(j)}} n_k\mathbf{w}^{(j)}(t)+\sum_{k\in S_t^{(l)}} n_k\mathbf{w}^{(l)}(t)\Big).\end{aligned}\tag{24}$$

If we further assume that the number of training samples utilized at ESs i, j and l are the same, we can rewrite equation (7) in the main manuscript as

$$\mathbf{w}_k(t)=\frac{1}{3}(\mathbf{w}^{(i)}(t)+\mathbf{w}^{(j)}(t)+\mathbf{w}^{(l)}(t)).\tag{25}$$

Similarly, considering N-overlapping ESs, the above two equations can be simply rewritten as

$$\mathbf{w}_k(t)=\frac{1}{\sum_{i\in\mathcal{N}}\sum_{k\in S_t^{(i)}} n_k}\sum_{i\in\mathcal{N}}\sum_{k\in S_t^{(i)}} n_k\mathbf{w}^{(i)}(t)\tag{26}$$

and

$$\mathbf{w}_k(t)=\frac{1}{N}\sum_{i\in\mathcal{N}}\mathbf{w}^{(i)}(t),\tag{27}$$

respectively, where $\mathcal{N}$ is the set of ES indices that simultaneously covers client k with $|\mathcal{N}|=N$. Now client k performs local update and broadcasts the updated model to all connected ESs. Finally, each ES aggregates the models in its coverage as in the current algorithm.

APPENDIX B
COMPARISON WITH SERVERLESS FL

Consider a serverless FL scheme in Fig. 16(a) where all devices in a specific cell can communicate with each other for model exchange. It can be seen that this scheme achieves exactly the same accuracy with our FedMes (in Fig. 16(b)) at each global round. However, the serverless FL scheme can require a significantly larger time delay in this scenario since each device should communicate with a significant number of neighbors at each global round.

APPENDIX C
PROOF OF THEOREM 1

We first have

$$\begin{aligned}&\mathbb{E}\Big[\,\|\bar{\mathbf{g}}(T)-\mathbf{g}(T)\|^2\,\Big]\\&=\frac{1}{K^2}\sum_{k=1}^{K}\mathbb{E}[\|\nabla F_k(\mathbf{w}_k(T),\xi_k(T))-\nabla F_k(\mathbf{w}_k(T))\|^2]\\&\le\frac{1}{K^2}\sum_{k=1}^{K}\sigma_k^2\end{aligned}\tag{28}$$

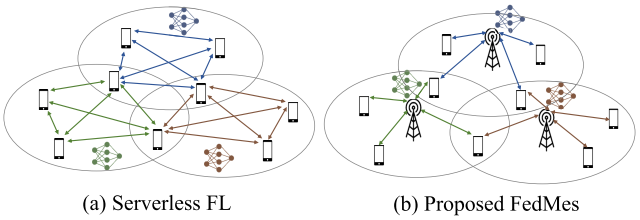


Fig. 16. Comparison between serverless FL and the proposed idea.

where (28) comes from Assumption 3. By combining the result of Lemma 1 and equation (28), we can write

$$\begin{aligned}&\mathbb{E}\Big[\,\|\bar{\mathbf{v}}(T+1)-\mathbf{w}^*\|^2\,\Big]\\&\quad\le(1-\mu\eta_T)\mathbb{E}\Big[\,\|\bar{\mathbf{w}}(T)-\mathbf{w}^*\|^2\,\Big]\\&\qquad+\eta_T^2B+Q(T)\end{aligned}$$

where $Q(T)=2\mathbb{E}[\sum_{k=1}^{K}\frac{1}{K}\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\|^2]$ and $B=\frac{1}{K^2}\sum_{k=1}^{K}\sigma_k^2+6\beta\Gamma$. Now by defining Δ_T as

$$\Delta_T=\mathbb{E}\Big[\,\|\bar{\mathbf{w}}(T)-\mathbf{w}^*\|^2\,\Big],\tag{29}$$

we have

$$\begin{aligned}\Delta_{T+1}&=\mathbb{E}\Big[\,\|\bar{\mathbf{v}}(T+1)-\mathbf{w}^*\|^2\,\Big]&(30)\\&\le(1-\mu\eta_T)\Delta_T+\eta_t^2B+Q(T).&(31)\end{aligned}$$

Here, (30) comes from (18) and by letting $\alpha_u=2\alpha_v$. Equation (31) directly comes from the result of Lemma 1.

Let $\eta_T=\frac{\alpha}{T+\gamma}$ for some $\alpha>\frac{1}{\mu}$ and $\gamma>0$ such that $\eta_1\le\min\{\frac{1}{\mu},\frac{1}{4\beta}\}=\frac{1}{4\beta}$ and $\eta_T\le2\eta_{T+E}$. Following the same procedure of Theorem 1 in [12], we have

$$\Delta_{T+1}\le\frac{x}{T+\gamma+1}+Q(T)\tag{32}$$

where $x=\max\{\frac{\alpha^2B}{\alpha\mu-1},(\gamma+1)\Delta_0\}$. Now by choosing $\alpha=\frac{2}{\mu}$ and $\gamma=\max\{\frac{8\beta}{\mu}-1,E\}$, we can finally write

$$\begin{aligned}&\mathbb{E}\Big[F(\mathbf{w}^f(T))\Big]-F(\mathbf{w}^*)\\&\le\frac{\beta}{2}\mathbb{E}\Big[\,\big\|\mathbf{w}^f(T))-\mathbf{w}^*\big\|^2\,\Big]\\&=\frac{\beta}{2}\Delta_T\\&\le\frac{2\beta}{\gamma+T}\Big(\frac{B}{\mu}+2\beta\Delta_0\Big)+\frac{\beta}{2}Q(T-1)\\&=\frac{2\beta}{\gamma+T}\Big(\frac{B}{\mu}+2\beta\|\mathbf{w}(0)-\mathbf{w}^*\|\Big)+\frac{\beta}{2}Q(T-1)\end{aligned}$$

which completes the proof.

APPENDIX D
PROOF OF PROPOSITION 1

For any α, we can write

$$\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(T)-\mathbf{w}_k(T)\right\|^2\Big] \tag{33}$$

$$=\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|(\mathbf{w}_k(T)-\alpha)-(\bar{\mathbf{w}}(T)-\alpha)\right\|^2\Big] \tag{34}$$

$$\leq\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\mathbf{w}_k(T)-\alpha\right\|^2\Big] \tag{35}$$

$$=\frac{1}{K}\mathbb{E}\Big[\sum_{k\in U_1\cup U_2\cup U_3\cup V_{12}\cup V_{23}\cup V_{31}}\left\|\mathbf{w}_k(T)-\alpha\right\|^2\Big] \tag{36}$$

where (35) holds because $\mathbb{E}[||x-\mathbb{E}[x]||^2]\leq\mathbb{E}[||x||^2]$ is satisfied for an arbitrary x. Let T_0 be the time step for aggregation at the ESs satisfying $T_0\leq T$ and $T-T_0\leq E-1$. Based on T_0, define $\bar{\mathbf{w}}^{(U_1)}(T_0)=\frac{1}{u+v}\Big(\sum_{k\in U_1}\mathbf{w}_k(T_0)+\frac{1}{2}\sum_{k\in V_{12}}\mathbf{w}_k(T_0)+\frac{1}{2}\sum_{k\in V_{31}}\mathbf{w}_k(T_0)\Big)$ and $\bar{\mathbf{w}}^{(C\setminus U_1)}(T_0)=\frac{1}{u+v}\sum_{k=1}^{K}\mathbf{w}_k(T_0)-\bar{\mathbf{w}}_k^{(U_1)}(T_0)$. Here, note that $|U_1|=|U_2|=|U_3|=u$ and $|V_{1,2}|=|V_{2,3}|=|V_{3,1}|=v$ hold.

Now by setting α as

$$\alpha=\frac{1}{u+v}\sum_{k=1}^{K}\mathbf{w}_k(T_0), \tag{37}$$

for $k\in U_1$, we can write

$$\mathbb{E}\Big[\sum_{k\in U_1}\left\|\mathbf{w}_k(T)-\alpha\right\|^2\Big]$$

$$=9\mathbb{E}\Big[\sum_{k\in U_1}||\frac{1}{3}\Big(\mathbf{w}_k(T)-\frac{1}{(u+v)}\sum_{k=1}^{K}\mathbf{w}_k(T_0)\Big)||^2\Big]$$

$$=9\mathbb{E}\Big[\sum_{k\in U_1}||\frac{1}{3}\Big(\mathbf{w}_k(T)-\bar{\mathbf{w}}^{(U_1)}(T_0)\Big)-\frac{1}{3}\bar{\mathbf{w}}^{(C\setminus U_1)}(T_0)||^2\Big]$$

$$\underset{(a)}{\leq}9\mathbb{E}\Big[\sum_{k\in U_1}\Big\{\frac{1}{3}||\mathbf{w}_k(t)-\bar{\mathbf{w}}^{(U_1)}(T_0)||^2+\frac{1}{6(u+v)}\Big(2\sum_{k\in U_2\cup U_3}||\mathbf{w}_k(T_0)||^2+\sum_{k\in V_{12}}||\mathbf{w}_k(T_0)||^2+\sum_{k\in V_{31}}||\mathbf{w}_k(T_0)||^2+2\sum_{k\in V_{23}}||\mathbf{w}_k(T_0)||^2\Big)\Big\}\Big]$$

$$\underset{(b)}{\leq}9\mathbb{E}\Big[\sum_{k\in U_1}\frac{1}{3}||\mathbf{w}_k(T)-\bar{\mathbf{w}}^{(U_1)}(T_0)||^2\Big]+6u\Omega$$

$$\underset{(c)}{\leq}3u\cdot4\eta_T^2(E-1)^2G^2+6u\Omega$$

where (a) comes from convexity of $||\cdot||^2$, (b) comes from $\mathbb{E}[||\mathbf{w}_k(t)||^2]\leq\Omega$, (c) comes from the proof of Lemma 3 of [12]. We can obtain the same results for $k\in U_2$ and $k\in U_3$.

Similarly, let $\bar{\mathbf{w}}^{(V_{12})}(T_0)=\frac{1}{u+v}\Big(\sum_{k\in U_1}\mathbf{w}_k(T_0)+\sum_{k\in U_2}\mathbf{w}_k(T_0)+\sum_{k\in V_{12}}\mathbf{w}_k(T_0)+\frac{1}{2}\sum_{k\in V_{23}}\mathbf{w}_k(T_0)+\frac{1}{2}\sum_{k\in V_{31}}\mathbf{w}_k(T_0)\Big)$ and $\bar{\mathbf{w}}^{(C\setminus V_{12})}(T_0)=\frac{1}{u+v}\sum_{k=1}^{K}\mathbf{w}_k(T_0)-\bar{\mathbf{w}}_k^{(V_{12})}(T_0)$. Now for $k\in V_{12}$, we can write

$$\mathbb{E}\Big[\sum_{k\in V_{12}}\left\|\mathbf{w}_k(T)-\alpha\right\|^2\Big]$$

$$=4\mathbb{E}\Big[\sum_{k\in V_{12}}||\frac{1}{2}\Big(\mathbf{w}_k(T)-\frac{1}{(u+v)}\sum_{k=1}^{K}\mathbf{w}_k(T_0)\Big)||^2\Big]$$

$$=4\mathbb{E}\Big[\sum_{k\in V_{12}}||\frac{1}{2}\Big(\mathbf{w}_k(T)-\bar{\mathbf{w}}^{(V_{12})}(T_0)\Big)-\frac{1}{2}\bar{\mathbf{w}}^{(C\setminus V_{12})}(T_0)||^2\Big]$$

$$\underset{(d)}{\leq}4\mathbb{E}\Big[\sum_{k\in U_1}\Big\{\frac{1}{2}||\mathbf{w}_k(t)-\bar{\mathbf{w}}^{(V_{12})}(T_0)||^2+\frac{1}{4(u+v)}\Big(2\sum_{k\in U_3}||\mathbf{w}_k(T_0)||^2+\sum_{k\in V_{23}}||\mathbf{w}_k(T_0)||^2+\sum_{k\in V_{31}}||\mathbf{w}_k(T_0)||^2\Big)\Big\}\Big]$$

$$\underset{(e)}{\leq}4\mathbb{E}\Big[\sum_{k\in U_1}\frac{1}{2}||\mathbf{w}_k(T)-\bar{\mathbf{w}}^{(V_{12})}(T_0)||^2\Big]+2v\Omega$$

$$\underset{(f)}{\leq}2v\cdot4\eta_T^2(E-1)^2G^2+2v\Omega$$

where (d) comes from convexity of $||\cdot||^2$, (e) comes from $\mathbb{E}[||\mathbf{w}_k(t)||^2]\leq\Omega$, (f) comes from the proof of Lemma 3 of [12]. We can obtain the same results for $k\in V_{23}$ and $k\in V_{31}$.

By inserting the above results to (36), our objective function can be upper bounded as

$$\mathbb{E}\Big[\sum_{k=1}^{K}\frac{1}{K}\left\|\bar{\mathbf{w}}(t)-\mathbf{w}_k(t)\right\|^2\Big]$$

$$\leq(\frac{24K}{L}+12u)\eta_T^2(E-1)^2G^2+(\frac{6K}{L}+12u)\Omega$$

$$\underset{(g)}{\leq}\frac{4}{(\gamma+T)\mu}(\frac{48K}{L}+24u)(E-1)^2G^2+(\frac{6K}{L}+12u)\Omega$$

where (g) holds by setting $\eta_T^2=\frac{4}{(\gamma+T)^2\mu^2}\leq\frac{4}{(\gamma+T)\mu}$, which completes the proof.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Stat.*, vol. 54, Fort Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.

[2] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *Proc. NIPS Workshop Private Multi-Party Mach. Learn.*, 2016. [Online]. Available: https://arxiv.org/abs/1610.05492

[3] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*. [Online]. Available: https://arxiv.org/abs/1610.02527

[4] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," 2019, *arXiv:1902.01046*. [Online]. Available: https://arxiv.org/abs/1902.01046

[5] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," 2019, *arXiv:1908.07873*. [Online]. Available: https://arxiv.org/abs/1908.07873

[6] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 1205–1221, Jun. 2019.

[7] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, Jan. 2020.

[8] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.
[9] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.
[10] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*. [Online]. Available: https://arxiv.org/abs/1806.00582
[11] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient federated learning from non-i.I.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2019.
[12] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in *Proc. Int. Conf. Learn. Represent.*, 2020. [Online]. Available: https://arxiv.org/abs/1907.02189
[13] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2021–2031.
[14] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data," 2018, *arXiv:1811.11479*. [Online]. Available: https://arxiv.org/abs/1811.11479
[15] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.
[16] T. T. Nguyen, V. N. Ha, L. B. Le, and R. Schober, "Joint data compression and computation offloading in hierarchical fog-cloud systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 293–309, Jan. 2019.
[17] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing'a key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.
[18] D.-J. Han, J.-Y. Sohn, and J. Moon, "Hierarchical broadcast coding: Expediting distributed learning at the wireless edge," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2266–2281, Apr. 2021.
[19] J. So, B. Guler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2168–2181, Jul. 2021.
[20] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2019, pp. 1387–1395.
[21] W. Y. B. Lim *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.
[22] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning ACROSS heterogeneous cellular networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 8866–8870.
[23] L. Liu, J. Zhang, S. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.
[24] S. Prakash, S. Dhakal, M. Akdeniz, A. S. Avestimehr, and N. Himayat, "Coded computing for federated learning at the edge," 2020, *arXiv:2007.03273*. [Online]. Available: https://arxiv.org/abs/2007.03273
[25] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "BrainTorrent: A Peer-to-Peer environment for decentralized federated learning," 2019, *arXiv:1905.06731*. [Online]. Available: https://arxiv.org/abs/1905.06731
[26] A. Lalitha, S. Shekhar, T. Javidi, and F. Koushanfar, "Fully decentralized federated learning," in *Proc. 3rd Workshop Bayesian Deep NIPS Workshop*, 2018, pp. 1–9.
[27] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. Stich, "A unified theory of decentralized sgd with changing topology and local updates," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5381–5393.
[28] J. Wang, A. K. Sahu, Z. Yang, G. Joshi, and S. Kar, "Matcha: Speeding up decentralized sgd via matching decomposition sampling," in *Proc. 6th Indian Control Conf. (ICC)*, Dec. 2019, pp. 299–300.
[29] H. T. Nguyen, V. Sehwag, S. Hosseinalipour, C. G. Brinton, M. Chiang, and H. Vincent Poor, "Fast-convergent federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 201–218, Jan. 2020.
[30] M. M. Amiri, D. Gunduz, S. R. Kulkarni, and H. V. Poor, "Convergence of update aware device scheduling for federated learning at the wireless edge," *IEEE Trans. Wireless Commun.*, vol. 20, no. 6, pp. 3643–3658, Jun. 2021.
[31] Y. Zhang, J. C. Duchi, and M. J. Wainwright, "Communication-efficient algorithms for statistical optimization," *J. Mach. Learn. Res.*, vol. 14, pp. 3321–3363, Nov. 2013.
[32] S. U. Stich, "Local SGD converges fast and communicates little," 2018, *arXiv:1805.09767*. [Online]. Available: http://arxiv.org/abs/1805.09767
[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[34] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*. [Online]. Available: https://arxiv.org/abs/1708.07747
[35] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

Dong-Jun Han (Student Member, IEEE) received the B.S. degree in mathematics and electrical engineering and the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejoen, South Korea, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree. His research interests include distributed machine learning and information theory.

Minseok Choi (Member, IEEE) received the B.S., M.S., and Ph.D. degrees from the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2011, 2013, and 2018, respectively. He was a Visiting Postdoctoral Researcher in electrical and computer engineering with the University of Southern California (USC), Los Angeles, CA, USA, and a Research Professor in electrical engineering with Korea University, Seoul, South Korea. He has been an Assistant Professor with Jeju National University, Jeju, South Korea, since 2020. His research interests include wireless caching networks, stochastic network optimization, and machine learning in wireless networks.

Jungwuk Park received the B.S. degree in electrical and electronic engineering from Yonsei University and the M.S. degree in electrical engineering from Korea Advanced Institute of Science and Technology (KAIST), in 2019 and 2021, respectively, where he is currently pursuing the Ph.D. degree. His research interests include various fields of machine learning, including distributed/federated learning.

Jaekyun Moon (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering at Carnegie Mellon University, Pittsburgh, PA, USA. From 1990 to 2009, he was with a Faculty Member with the School of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, USA. He consulted as the Chief Scientist of DSPG, Inc., from 2004 to 2007. He also worked as the Chief Technology Officer at Link-A-Media Devices Corporation. He is currently a Professor of electrical engineering at KAIST. His research interests include channel characterization, signal processing and coding for data storage, and digital communication. He received the McKnight Land-Grant Professorship from the University of Minnesota. He was a recipient of the IBM faculty development awards, the IBM partnership awards, and the National Storage Industry Consortium (NSIC) Technical Achievement Award for the invention of the maximum transition run (MTR) code, a widely used error-control/modulation code in commercial storage systems. He served as the Program Chair for the 1997 IEEE Magnetic Recording Conference and the Past Chair of the Signal Processing for Storage Technical Committee, IEEE Communications Society. He served as a Guest Editor for the 2001 IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS Special Issue on Signal Processing for High Density Recording and an Editor for IEEE TRANSACTIONS ON MAGNETICS in the area of signal processing and coding from 2001 to 2006.