# Resource Allocation Based on Digital Twin-Enabled Federated Learning Framework in Heterogeneous Cellular Network

Yejun He, *Senior Member, IEEE*, Mengna Yang, Zhou He, and Mohsen Guizani, *Fellow, IEEE*

***Abstract*—Federated learning (FL) allows user devices (UDs) to upload local model parameters to participate in a global model training, which protects UDs' data privacy. Nevertheless, FL still faces challenges such as core network congestion, UDs' limited resources and less efficient mapping between devices and cyber systems. Therefore, in this article, we integrate the digital twin (DT) and the mobile edge computing (MEC) technologies into a hierarchical FL framework in the heterogeneous cellular network scenario. When the UDs are not in the service range of the small base stations (SBSs), the framework allows macro base stations to assist UDs' local computation, thus reducing the transmission delay. It also protects the user privacy and allows more users to join in the training in order to improve the FL accuracy. In addition, we propose a deep reinforcement learning-based scheme to solve the joint optimization problem of dynamic UDs-stations association and resource allocation, thereby minimizing the energy consumption within a limited time delay. Simulation results show that our proposed scheme not only effectively reduces the task transmission failure rate and energy consumption compared with the baseline scheme, but also saves the communication cost through the DT network.**

***Index Terms*—Digital twin (DT), federated learning, heterogeneous cellular network, mobile edge computing (MEC), resource allocation.**

## I. Introduction

With the rapid development of 5th generation (5G) mobile communications and intelligent terminal equipment, traditional cloud computing has been unable to meet user's requirements for ultra-high transmission rates and low latency. In order to meet the user's demand for low latency and high speed, mobile edge computing (MEC) [1] has emerged, and the distributed advantages of MEC are beneficial to the mobility of devices. In addition, the application of artificial intelligence (AI) in 5G cellular is very promising due to the challenges such as network complexity, model deficit, and algorithm deficit [2]. Especially in the two scenarios of enhanced mobile broadband (eMBB) and ultra reliable low latency communication (URLLC) for 5G cellular networks, the two transmission technologies of orthogonal frequency-division multiplexing (OFDM) and orthogonal frequency division multiple access (OFDMA) are still applicable, which can reduce co-channel interference.

In recent years, smart devices have become increasingly prevalent and they are usually used as main computing resources. Therefore, various types of devices inevitably lead to data heterogeneity. In addition, in order to protect data privacy, users are reluctant to share a large amount of data, and traditional centralized machine learning (ML) with a large amount of data for model training becomes inappropriate. Thus, the emergence of FL could mitigate some of the above challenges [3].

FL is a classic example of distributed learning. It only needs to upload the local model parameters of the user devices (UDs), and does not need to share the entire dataset with other users, thereby protecting the privacy of user's data from being leaked. However, since long-distance communication between UDs and the cloud server can cause longer delays and core network congestion problems, most studies focus on reducing the communication cost in FL [4], [5], [6], [7], [8], [9]. The authors in [4] proposed structured and sketched updates to reduce the uplink communication cost and improve the communication efficiency between the client and the cloud server. Wang et al. [5] proposed an orthogonal approach, i.e., communication mitigation federated learning (CMFL) algorithm, which reduces the communication overhead by reducing irrelevant local updates uploaded to the cloud. Sattler et al. [6] proposed a sparse ternary compression method to reduce the massive communication overhead between the client and the cloud. Moreover, since the existing FL is only considered utilizing first-order gradient descent, the authors of [7] also introduced a momentum FL, which used momentum gradient descent in the local update of FL to accelerate the convergence of FL. Furthermore, Chen et al. [8] proposed a probabilistic user selection and resource allocation scheme to minimize the learning cost of FL. Xu et al. [9] developed an optimization method based on decay and consensus on the basis

Manuscript received 6 April 2022; revised 7 August 2022; accepted 6 September 2022. Date of publication 12 September 2022; date of current version 16 January 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62071306, and in part by Shenzhen Science and Technology Program under Grants JCYJ20200109113601723, GJHZ20180418190529516, JSGG20210802154203011, and JSGG20210420091805014. The review of this article was coordinated by Dr. Phone Lin. *(Corresponding author: Yejun He.)*

Yejun He and Mengna Yang are with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: heyejun@126.com; 2019115452@qq.com).

Zhou He is with the School of Mechanical Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: zhe12@umd.edu).

Mohsen Guizani is with the Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Masdar City, UAE (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/TVT.2022.3205778

of the variation-aware periodic averaging method, which reduces the communication overhead and improves the convergence performance.

Due to the emergence of MEC, existing studies are carried out based on the device-edge two-layer framework or the device-edge-cloud three-layer framework to migrate tasks from the core network to the edge network. Based on the device-to-edge two-layer architecture, it can significantly reduce the communication delay and alleviate the core network congestion problem by sinking the aggregation server to the edge network. Dinh et al. [10] leveraged the Pareto efficiency model to balance the training time and energy consumption between clients and edge servers. Chen et al. [11] proposed a large-scale matching algorithm with incomplete preference lists to assign tasks to devices with higher willingness and reduce task execution time. Ren et al. [12] proposed the problem of CPU acceleration and jointly optimized batch size and communication resources, thereby reducing the communication time delay between the client and the edge server. Feng et al. [13] deployed the FL framework in the MEC system and optimized the model compression, sample selection and user selection strategies through a joint optimization algorithm. Besides, the authors in [14] solved the time delay and energy consumption during training by fine-grained offloading based on actor-critic FL in a multi-MEC environment. The device-edge-cloud based three-layer architecture that includes client, edge server, and cloud. In this architecture, the model parameter aggregation includes edge aggregation and cloud aggregation. Since lower deployment cost and easier deployment of edge server than the cloud servers, and the FL model of the three-layer architecture can reduce the core network burden and communication time delay, it can aggregate more user models to participant FL training. For example, Wu et al. [15] proposed a three-layer personalized FL framework that enables UDs not only to enjoy the benefits from the global model, but also to create their own personalized models to meet the needs of their resources and applications. Zhu et al. [16] used federated reinforcement learning (FRL) to make the agent and the server to jointly make decisions without sharing each other's model, which can enhance the privacy of the three-layer model. Meanwhile, the authors of [17] and [18] considered three-layer FL models for dynamic edge association and resource allocation to minimize time delay and energy consumption. Generally, the three-layer model framework can not only reduce the core network congestion problem, but also support more users to participate in FL training.

However, with the communication delay and operation data growing, it is difficult for the MEC server to collect and analyze the operation data of the UDs for online optimization. Therefore, Glaessgen et al. [19] proposed a digital twin (DT) model that connects physical entities and network systems. DT is the mapping between physical entities and their virtual twins. The virtual model can obtain the state information of physical entities by sensor data [20], and create virtual objects in digital space by software definition network [21]. Physical objects and virtual objects can real-time transmission, thereby predicting, estimating and analyzing the dynamic changes of physical entities to assist the network decision-making module to make decisions. The concept of DT was first introduced in 2002 by Michael Grives in a presentation titled "The Conceptual Ideal of PLM". And NASA has put the concept of DT into practical use for the first time. They had developed two identical spacecraft in the Apollo program to simulate and reflect the state of space in flight training. Subsequently, multiple DTs collaborate to form a DT network, which is applied in scenarios such as manufacturing, healthcare and intelligent transportation system [22]. Although DT has some advantages, DT modeling that needs to synchronize a large amount of data encounters challenges because of the emphasis on data privacy and security.

Given the advantages of DT and FL, the combination of DT and FL brings light to the above challenges. DT does not have to synchronize all the original data because of FL can learn a model from user data to synchronize to DT, which can not only reduce the communication burden but also reduce the risk of data leakage. At the same time, DT can reduce the influence of unreliable communication between the UDs and the server, and directly analyze the DT network to obtain the operating status information of the physical entities, and then make optimization decisions by wireless transmission to the network decision module, thereby assisting the FL model train. For example, Sun et al. [23] considered the difference between the true value and its numerical representation in the DT mapping and explored the impact of this difference on the unloading decision. Lu et al. integrated DT and FL into the MEC network and proposed the digital twin edge networks (DITENs), which can reduce the overhead of data transmission, improve communication efficiency, and protect data privacy [24]. Similarly, the authors of [25] applied DT and AI to the vehicular edge network, and adopted a distributed multiagent learning scheme for task offloading and resource allocation.

In view of the above observations, our motivation is to utilize the DT network to assist in building a digital model, and to utilize edge server to solve the straggler effect problem of UDs in FL process, thereby reducing the system cost of the UDs. We propose a communication efficient and DT-enabled FL framework based on a heterogeneous cellular network. The main contributions of this work are summarized as follows.

- Based on the heterogeneous cellular network, we adopt the OFDMA to avoid the interference between UDs. In order to reduce the transmission delay, improve the user quality of service (QoS) and protect the user privacy, we develop a framework that integrates the DT and MEC technologies into a three-layer FL. The proposed framework allows more users to participate in the FL process.
- We develop a communication-efficient FL process based on dynamic UDs-stations association and user cross-cell mobility, further reducing the UDs' energy consumption with limited power.
- We transform the resource allocation joint optimization problem into a Markov decision process (MDP), and propose a deep reinforcement learning-based scheme to solve the energy consumption minimization problem.

The rest of this paper is organized as follows. The system model is presented in Section II. Section III introduces the objective function and the solution to the joint optimization of

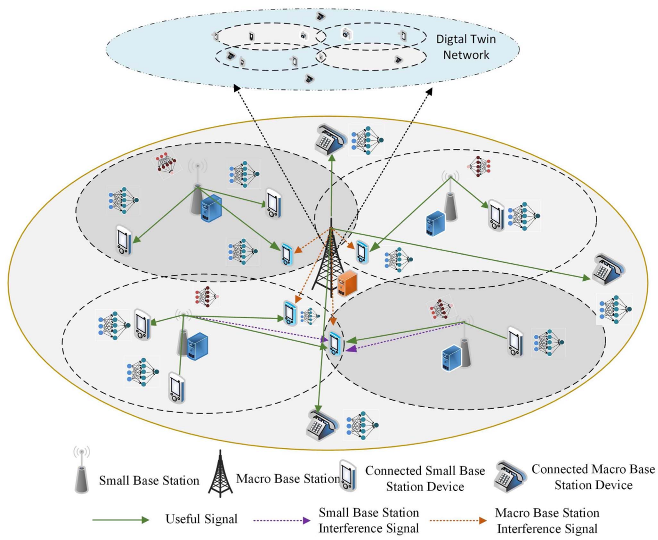


Fig. 1. The system model of DT-enabled FL Framework.

resource allocation. Section IV discusses simulation results and conclusion is given in Section V.

## II. System Model

As shown in Fig. 1, we consider a heterogeneous cellular network with a DT system model. There are one macro base station (MBS) with a server, *S* small base stations (SBSs) with an aggregation server and *K* UDs, which can be denoted as *M*, $\{B_1, B_2, B_3, \ldots, B_j, \ldots, B_S\}$, $\{U_1, U_2, U_3, \ldots, U_k \ldots, U_K\}$, respectively. Each base station is equipped with an edge server that not only has a computing capability, but also has the ability to aggregate parameters. Besides, each UD has computationally intensive tasks and has some computing capability. During one iteration of FL, all UDs first receive a global model from the MBS server, and then dynamically select a model training method according to its own state. Specifically, there are two methods: if the UDs are in the service range of the SBSs, they first perform local training, upload its trained model parameters to the SBS server, and the SBSs server will aggregate the parameters of all UDs and then upload to the MBS server; Otherwise, UDs directly upload parameters to the MBS server. Since UDs upload the model parameters to MBS, in order to avoid long time delays, the MBS server can assist local training through the DT network that deployed on the MBS. As mentioned above, the virtual object of UDs $DT_k$ can be expressed as $DT_k = \{\mathfrak{F}_{\omega_k}, D_k + \Delta\zeta\}$, where $\mathfrak{F}_\omega$ is the model of UD *k*, and the $D_k$ and $\Delta\zeta$ are the size of UDs task and the deviation between the actual value and DT mapping value of the UDs, respectively. It is to be noted that there is a deviation between the actual value and the mapped value of UDs in the DT network.

### A. Hierarchical Federated Learning Model

From the perspective of time delay and energy consumption, the MBS assisted computing can alleviate the communication burden, reduce the straggler effect, and improve the training performance of the FL model. FL provides a learning model for DT networks. At the same time, DT can reduce the impact of unreliable communications between the UDs and the server, and obtain the state information of physical entities by analyzing the DT network, thereby assisting the MBS server make decisions. The virtual object and physical entity cooperatively train the shared model and can improve the reliability and security of the system. These advantages motivate us to adopt a DT-enabled hierarchical model. A FL iteration is divided into three parts, which are local training of UDs, aggregation of the SBSs server, and aggregation of the MBS server.

Let $\mathcal{D} = \{\mathfrak{x}_k, \mathfrak{y}_k\}$ denote the training dataset of UDs. Where $|\mathcal{D}|$ represents the total number of training samples, $\mathfrak{x}_k$ is the input sample and $\mathfrak{y}_k$ is the corresponding sample label. The process of the FL training is as follows.

*1) Local Training of UDs:* In the *n*-th iteration, after receiving the initialized global model $\omega$, each UD uses its own data samples $\mathcal{D}_k$ to update the local parameter model $\omega_k^n$ to find the best parameters $\tilde{\omega}$ to minimize the local model loss function $\mathfrak{F}(\omega_k^n)$.

$$\mathfrak{F}(\omega_k^n) = \frac{1}{|\mathcal{D}_k|} * \sum_{x_k, y_k \subseteq \mathcal{D}_k} \mathfrak{f}\left(\omega_k^n, \mathfrak{x}_k, \mathfrak{y}_k\right) \tag{1}$$

And the remaining dataset samples are updated locally. We have

$$\omega_k^n = \omega_k^{n-1} - \mu_n \nabla \mathfrak{F}\left(\omega_k^{n-1}\right) \tag{2}$$

where $\mu_n$ is the gradient descent step size in the *n*-th iteration, $(\mathfrak{x}_k, \mathfrak{y}_k)$ is data samples of the UD *k* and $\mathfrak{f}(\cdot)$ quantifies the difference between the estimated and the actual values.

*2) Aggregation of SBSs Server:* In the *n*-th iteration, the edge server attached to the SBS $S_j$ aggregates the local model parameters uploaded within its own service scope to obtain a partial global parameter model $\omega_j^n$.

$$\omega_j^n = \frac{1}{|\mathcal{D}_{s_j}|} \sum_{k \in s_j} \mathcal{D}_k \cdot \omega_k^n \tag{3}$$

where $s_j$ is the set of UDs that select SBS $S_j$.

*3) Aggregation of MBS Server:* In the *n*-th iteration, the server deployed in the MBS aggregates the model parameters generated by its computation and from the SBSs ($\omega_j^n$) and some local model parameters ($\omega_k^n$) to obtain a new global model $\omega^n$.

$$\omega^n = \frac{1}{\mathcal{D}} \left( \sum_{k \in s_k} (1 - v_k)\mathcal{D}_k \omega_k^n + v_k \mathcal{D}_k \omega_m^n + \sum_{k \in s_j} \mathcal{D}_k \omega_j^n \right) \tag{4}$$

where $|\mathcal{D}|$ is the total dataset size of UDs, and $s_k$ is the set of UDs that selects the MBS.

*Remark 1 (The convergence performance of (4))* In our design, the MBS server can assist in reducing the straggler effect and enable more UDs to participate in the FL training. This is similar to adding a new client with strong computing power to the original FL (ignoring the data offloading), and its convergence performance is equivalent to the centralized gradient descent algorithm. In the global aggregation phase, after the parameters are broadcast, we have $\omega_k^n = \omega_j^n = \omega^n$. Specifically, according to the linearity of the gradient operator and (4), we can obtain

$$\begin{aligned}
\omega^n &= \frac{1}{\mathcal{D}}\left(\sum_{k\in s_k}(1-v_k)\mathcal{D}_k\omega_k^n + v_k\mathcal{D}_k\omega_m^n + \sum_{k\in s_j}\mathcal{D}_k\omega_j^n\right)\\
&= \frac{1}{\mathcal{D}}\Bigg(\sum_{k\in s_k}(1-v_k)\mathcal{D}_k\left(\omega_k^{n-1} - \mu_n\nabla\mathfrak{F}_k(\omega_k^{n-1})\right)\\
&\quad + v_k\mathcal{D}_k\left(\omega_m^{n-1} - \mu_n\nabla\mathfrak{F}_j(\omega_m^{n-1})\right)\\
&\quad + \sum_{k\in s_j}\mathcal{D}_k\left(\omega_j^{n-1} - \nu_n\nabla\mathfrak{F}_j\left(\omega_j^{n-1}\right)\right)\Bigg)\\
&= \frac{1}{\mathcal{D}}\Bigg(\sum_{k\in s_k}\left((1-v_k)\mathcal{D}_k\omega^{n-1} + v_k\mathcal{D}_k\omega^{n-1}\right)\\
&\quad + \sum_{k\in s_j}\mathcal{D}_k\omega^{n-1} - \mu_n\Bigg(\sum_{k\in s_k}\Big((1-v_k)\mathcal{D}_k\nabla\mathfrak{F}_k\left(\omega^{n-1}\right)\\
&\quad + v_k\mathcal{D}_k\nabla\mathfrak{F}_m(\omega^{n-1})\Big) + \sum_{k\in s_j}\mathcal{D}_k\nabla\mathfrak{F}_j\left(\omega^{n-1}\right)\Bigg)\Bigg)\\
&= \frac{\sum_{k\in s_k}\mathcal{D}_k\omega^{n-1} + \sum_{k\in s_j}\mathcal{D}_k\omega^{n-1}}{\mathcal{D}}\\
&\quad - \frac{\mu_n}{\mathcal{D}}\Bigg(\Bigg(\sum_{k\in s_k}\left((1-v_k)\mathcal{D}_k\nabla\mathfrak{F}_k\left(\omega^{n-1}\right)\right)\\
&\quad + v_k\mathcal{D}_k\nabla\mathfrak{F}_m(\omega^{n-1})\Bigg) + \sum_{k\in s_j}\mathcal{D}_k\nabla\mathfrak{F}_j(\omega^{n-1})\Bigg)\\
&= \omega^{n-1} - \nu_n\nabla\mathfrak{F}\left(\omega^{n-1}\right)
\end{aligned} \tag{5}$$

## B. Communication Model

In this paper, we consider an OFDMA for UDs to avoid co-channel interference. Given the total bandwidth $B$, the channel needs to transmit model parameters among UDs, SBSs and MBS to achieve a system model convergence.

*1) UDs-SBSs-MBS Communication Model:* In this part, after the UDs finish one round of local training and upload their local model parameters to the SBSs server, the server will aggregate the model parameters of all receiving UDs, and then the SBSs server will transmit the partial global model to the MBS server. It is to be noted that UDs transmit local model parameters to the SBSs, they can receive the interference signal from the MBS or the other SBSs, as shown in Fig. 2. Let the received signal to interference plus noise ratio (SINR) represent the channel status and is given by

$$SINR_k^{u,s} = \frac{p_{user}\cdot h_k^{u,s}}{\sum_{\bar{s}}p_{in}h_k^{u,\bar{s}} + p_{on}h_k^{u,m} + \varsigma^2}, k\in s_j \tag{6}$$

and define $\xi_k$ as the bandwidth allocation ratio for UD $k$. Thus the corresponding transmission rate [21] is expressed as

$$r_k^{u,s} = \xi_k Blog_2\left(1 + SINR_k^{u,s}\right), k\in s_j \tag{7}$$

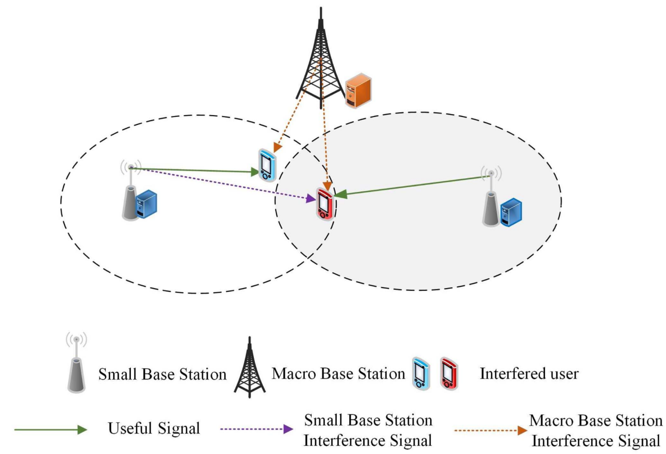


Fig. 2. The user with interference signal.

where $p_{user}$, $p_{in}$ and $p_{on}$ represent the transmit power of UDs, the power of interference signal from other SBSs and the power of interference signal from MBS, respectively. $h_k^{u,s}$ is the channel gain between UD $k$ and its connected SBS, $h_k^{u,\bar{s}}$ is the channel gain between UD $k$ and other SBSs, and $h_k^{u,m}$ is the channel gain between UD $k$ and MBS. $\varsigma^2$ represents the background noise power.

Similarly, the transmission rate from SBSs to MBS is given by

$$r_j^{s,m} = \xi_j Blog_2\left(1 + \frac{p_e h_j^{s,m}}{\varsigma^2}\right) \tag{8}$$

where $p_e$ represents the transmit power of SBSs, $h_j^{s,m}$ is the channel gain between SBS and MBS, and $\xi_j = \sum_{k\in s_j}\xi_k$.

*2) UDs-MBS Communication Model:* When UDs are outside the service scope of SBSs, they will upload the trained local model parameters to the MBS for aggregation, and MBS server with the DT network assists the UDs to train the local model. The transmission rate from UDs to the MBS is expressed as

$$r_k^{u,m} = \xi_k Blog_2\left(1 + \frac{p_{user}h_k^{u,m}}{\varsigma^2}\right), k\in s_k \tag{9}$$

where $h_k^{u,m}$ represents the channel gain between the UDs and the MBS.

## C. Computation Model

In our system framework, the computation model including UDs local computing delay and energy consumption, the MBS server computing delay and energy consumption, UDs transmission delay and energy consumption, and the MBS server maintains DT system energy consumption. Since the MBS and the SBSs servers have strong computing capability and transmission rate, we ignore the time delay and the energy consumption when the model parameters aggregate and the UDs download the global model.

*1) UDs-SBSs-MBS Computation Model:* For the UD $k$ with the task $D_k$ in the service range of SBSs, we define the CPU cycle frequency of UD $k$ as $f_k$. And let $\chi_k$ represent the number of CPU cycles that UD *k* needs to process one unit of data. Thus, the local computation time delay of the UD *k* is

$$t_k^{us-cmp} = \frac{\chi_k |D_k|}{f_k}, k \in s_j \tag{10}$$

and the corresponding energy consumption is given by

$$e_k^{us-cmp} = \frac{\tau}{2} \chi_k |D_k| f_k^2, k \in s_j \tag{11}$$

where $\tau$ is effective capacitance coefficient, $\tau = 2 \cdot 10^{-28}$[18].

In the transmission phase, the UD *k* transmits the local parameter model to SBSs server, and SBSs server transmits the partial aggregated model parameters to the MBS server. The two transmission time delay can be characterized by

$$t_{kj}^{us-tran} = \frac{|o_{kj}|}{r_k^{u,s}}, k \in s_j \tag{12}$$

and

$$t_j^{sm-tran} = \frac{|o_j|}{r_k^{u,s}} \tag{13}$$

where $|o_{kj}|$ is the size of transmitted model parameters from UD *k* to SBS *j*, and $|o_j|$ represents the size of transmitted model parameters from SBSs *j* to MBS server. Numerically, $|o_{kj}|$ is equal to $|o_j|$, and the corresponding transmission energy consumption can be expressed as

$$e_{kj}^{us-tran} = p_{user} \cdot t_k^{us-cmp}, k \in s_j \tag{14}$$

and

$$e_j^{sm-tran} = p_e \cdot t_j^{sm-cmp} \tag{15}$$

We then derive the total time delay and the total energy consumption in this transmission mode respectively as

$$T^{us} = max\left\{max\left\{t_k^{us-cmp} + t_{kj}^{us-tran}\right\} + t_j^{sm-tran}\right\}, \\ k \in s_j \tag{16}$$

$$E^{us} = \sum_{k \in s_j} \left(e_k^{us-cmp} + e_{kj}^{us-tran} + e_j^{sm-tran}\right) \tag{17}$$

*2) UDs-MBS Computation Model:* In this way, when the UDs are not in the service of SBSs, besides UDs will train the model locally, the MBS server can assist training the local model through the DT network. Correspondingly, the computation time delay and the energy consumption generated by UDs' local computation are given by

$$t_k^{um-cmp} = \frac{\chi_k \upsilon_k |D_k|}{f_k}, k \in s_k \tag{18}$$

$$e_k^{um-cmp} = \tau \chi_k \upsilon_k |D_k| f_k^2, k \in s_k \tag{19}$$

where $\upsilon_k$ represents the proportion of local computing tasks.

We express the computation time delay and its computation energy consumption trained by MBS server with the DT network as well as the energy consumption of the MBS to maintain the virtual object respectively as

$$t_k^{m-cmp} = \frac{\chi_k (1 - \upsilon_k) |D_k + \Delta\zeta|}{f_m}, k \in s_k \tag{20}$$

$$e_k^{m-cmp} = p_m \cdot t_k^{m-cmp}, k \in s_k \tag{21}$$

$$e_k^{dt} = p_m \cdot max\left\{t_k^{um-cmp}, t_k^{m-cmp}\right\}, k \in s_k \tag{22}$$

where $p_m$ and $f_m$ are the computing power and the computing resource of MBS server, respectively.

Meanwhile, the UDs' corresponding transmission time delay and energy consumption in this way are expressed as

$$t_k^{um-tran} = \frac{\upsilon_k |o_k|}{r_k^{u,m}}, k \in s_k \tag{23}$$

$$e_k^{um-tran} = p_{user} \cdot t_k^{um-tran}, k \in s_k \tag{24}$$

where $|o_k|$ is equal to the $|o_{kj}|$.

The total time delay and total energy consumption in this transmission mode respectively are expressed as

$$T^{um} = max\left\{max\left\{t_k^{um-cmp}, t_k^{m-cmp}\right\} + t_k^{um-tran}\right\}, \\ k \in s_k \tag{25}$$

$$E^{um} = \sum_{k \in s_k} e_k^{um-cmp} + e_k^{m-cmp} + e_k^{dt} + e_k^{um-tran} \tag{26}$$

Overall, the total time delay and total energy consumption of the system are given by

$$T = max\{T^{us}, T^{um}\} \tag{27}$$

$$E = E^{us} + E^{um} \tag{28}$$

## III. Problem Formulation and the Solution

### A. Problem Formulation

According to the aforementioned models, we now formulate an optimization problem to minimize the system energy consumption in one global iteration of the FL process. The minimization problem is formulated as

$$\begin{aligned} & min_{\xi_1,\xi_2,\xi_3,\ldots,\xi_k,f} E \\ s.t.\ & \\ & C1: 0 < T \le T_l \\ & C2: f_k^{min} \le f_k \le f_k^{max}, k \in K \\ & C3: 0 < \xi_k < 1, \sum_k \xi_k = 1, k \in K \\ & C4: 0 \le \upsilon_k \le 1, k \in s_k \\ & C5: t_k^{us-cmp}, t_k^{um-cmp} \le T_{max}^{cmp}, k \in K \\ & C6: t_k^{us-tran}, t_k^{um-tran} \le T_{max}^{tran}, k \in K \end{aligned} \tag{29}$$

where C1 defines the training time threshold to ensure the comprehensive performance of the system model, C2 and C3 respectively represent the computation capacity constraints and the uplink communication resource constraints, C4 means the proportion of task data size that trains in UDs-MBS model, C5 represents the maximum local computation delay of UDs, and C6 indicates the maximum transmission time delay constraints to ensure the QoS of the communication between UDs and SBSs.

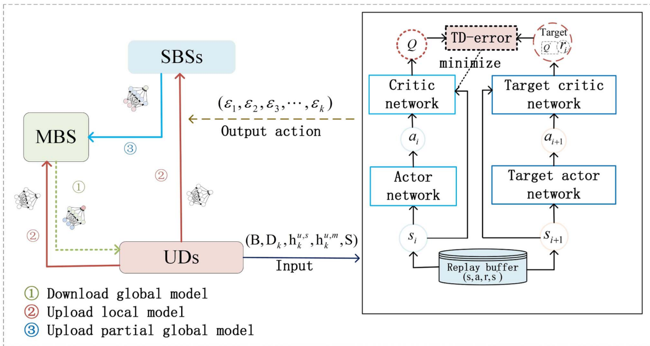


Fig. 3. Framework of DDPG-based dynamic UDs-stations association and resource allocation.

## B. Problem Transformation and Solution

In different FL iterations, with the dynamic changes in the size of the data collected by the UDs, available computing resources, transmission channels state, and UDs-stations association, the objective is to properly allocate resources to minimize the energy consumption under limited time delay. The solution of the objective function is a stochastic optimization problem that needs to determine the resource allocation at each time slot. Therefore, traditional methods are difficult to deal with the constrained dynamic optimization problem. The deep deterministic policy gradient (DDPG) [27] algorithm has a low optimization computational complexity and can be used to solve a mixed-integer nonlinear programming problem. In this case, we utilize the DDPG algorithm to solve the optimization problem of resource allocation. The framework is shown in Fig. 3.

The Markov decision process (MDP) is the optimal decision process of the stochastic dynamic system based on Markov process theory. That is, according to each observed state, select an action from the available action set to make a decision, randomly select the next state, and obtain its state transition probability. The decision-maker makes new decisions based on the newly observed state. The mathematical description of MDP is: $\{s, a, r, p\}$, where $s$ represents the state of the agent, $a$ represents the action that the agent can take in each state, $r$ represents the obtained reward when entering the next state and $p$ is the state transition probability of the agent taking action $a$ in the current state $s$. We transform the problem of solving the objective function into MDP problem, and its modeling process is as follows.

*1) State Modeling:* We utilize DT network to build a digital model. After collecting the network environment information, the DT network updates the collected state information, such as the association decision and channel status between UDs and the server. Finally, the DT network transmits the collected state information to the DDPG agent module. Therefore, at each time slot $t$, the state space can be described as $s_t = \{D_k, c_k, h_k^{u,s}, h_k^{u,m}, B, S\}$, where the $D_k$ is the task data size of UDs $k$, $c_k$ computing density, $h_k^{u,s}$ represents the channel gain from UDs and SBSs, $h_k^{u,m}$ is the channel gain from UDs and MBS, $B$ is the total bandwidth and the $S$ is the decision of UDs-SBS association.

*2) Action Modeling:* The agent makes reasonable resource allocation decisions based on the observed state, and the action can be described as $a_t = \{\xi_1, \xi_2, \ldots, \xi_k\}$.

*3) Reward Modeling:* Affected by action $a_t$ to the environment, the environment will return the new state $s_{t+1}$ and the reward $r_t$ at the next moment. When the constraint of the transmission time delay is satisfied, we set the reward as the total energy consumption of UDs. Otherwise, we add a reasonable penalty so that the agent can choose right action. The $r_t$ is given by

$$r_t = \begin{cases} -\sum_k^K \left(e_k^{cmp} + e_k^{tran}\right), & if\ t_k^{tran} \leq T_{max}^{tran} \\ -\sum_k^K \left(e_k^{cmp} + e_k^{tran}\right) - c. & \text{otherwise} \end{cases} \tag{30}$$

where $k \in K$, $c$ is a constant, $e_k^{cmp}$ is the computation of energy consumption, $e_k^{tran}$ represents the transmission energy consumption of UDs and $t_k^{tran} \leq T_{max}^{tran}$ represents the transmission delay $t_k^{tran}$ of the model should not exceed the limit of the maximum transmission delay $T_{max}^{tran}$, respectively.

In this work, the action $a_t$ of the agent is continuous and $a_t \in [0, 1]$, so we use the DDPG to solve the MDP problem. In the proposed DDPG-based resource allocation algorithm, the goal is to choose a resource allocation action with the optimal policy $o^*$ to minimize the system total energy consumption. Specifically, the algorithm can be described as follows. After initializing the actor network $\pi(s; \phi_a)$ and the critic network $Q(s, a; \phi_c)$, first of all, the agent selects action $a_t = \pi(s; \phi_a) + \vartheta_t$ according to the current policy $\pi(s; \phi_a)$ and the exploration noise $\vartheta_t \sim N(0, \sigma)$. Second, the agent performs resource allocation action $a_t$ and obtains the reward $r_t$ and new state $s_{t+1}$. Third, the agent samples a mini-batch of $L$ transitions $(s_i, a_i, r_i, s_{i+1})$ from the replay buffer $(s_t, a_t, r_t, s_{t+1})$, so the loss function can be calculated as $L_{loss} = \frac{1}{L}\sum_i (y_i - Q(s_i, a_i; \phi_c))^2$, where $y_i = r_i + \gamma Q'(s_{i+1}, \pi'(s_{i+1}; \phi'_a); \phi'_c)$. Finally, the target actor and critic networks are updated by $\phi'_a$: $\tau\phi_a + (1-\tau)\phi'_a$; $\phi'_c$: $\tau\phi_c + (1-\tau)\phi'_c$, respectively. The pseudo-code for the algorithm implementation is shown as Algorithm 1.

## IV. Simulation Results and Discussion

In our system model, we consider one MBS with an edge server, and four SBSs with an aggregation server and ten UDs with computation capabilities. The coverage radius of MBS and SBSs are 250 m and 125 m, respectively. The task data size trained by UDs is uniformly distributed within [40,80] Mbits. The minimum and maximum UDs computing resources are set as 0.8 and 1 GHz, and the computing density follows a uniform distribution within [30, 100] cycles/bit. In addition, we set the total system bandwidth for model uploading as $B = 10$ MHz and the background noise power as $10^{-8}$W.

The simulation results are performed with Python 3.7, parl 1.3.1 and paddlepaddle 1.6.3. The algorithm parameters are as follows: Actor-Critic network has two fully connected hidden layers, each layer has 64 neurons, and the activation function adopts ReLU function. And we set the input dimension of the Actor network to $9 \times K$. "9" denotes the sum of $4 \times K$ and $5 \times K$, where the $4 \times K$ denotes the channel gain with between UDs and SBSs and $5 \times K$ represents other five state dimension. In

**Algorithm 1:** DDPG-Based Resource Allocation Algorithm in Dynamically DT-Enabled FL Framework.

Define the structure of the actor network $\pi(s;\phi_a)$ and critic network $Q(s,a;\phi_c)$, and randomly initialize them with weights $\phi_a$ and $\phi_c$;
Initialize target actor network $\pi'(s;\phi'_a)$: $\phi'_a \leftarrow \phi_a$;
Initialize target critic network $Q'(s,a;\phi'_c)$: $\phi'_c \leftarrow \phi_c$;
Initialize replay buffer *R* and some network hyperparameters.
**for** *episode = 1,2,...E* **do**
  Initialize a random process $\vartheta$ for action exploration, and obtain initial state $s_1$ from DT network.
  **for** *step t = 1,2,...,T* **do**
    Select action $a_t = \pi(s;\phi_a)+\vartheta_t$, according to the current policy $\pi(s;\phi_a)$ and exploration noise $\vartheta_t \sim N(0,\sigma)$;
    Execute resource allocation action $a_t$;
    Obtain reward received $r_t$ and new state $s_{t+1}$ from DT network;
    Store transition $(s_t,a_t,r_t,s_{t+1})$ that the agent interacts with the environment in *R*;
    Sample a mini-batch of *L* transitions $(s_i,a_i,r_i,s_{i+1})$ from *R*;
    Update critic network by minimizing the loss:
      $y_i = r_i + \gamma Q'(s_{i+1},\pi'(s_{i+1};\phi'_a);\phi'_c)$;
      $L_{loss} = \frac{1}{L}\sum_i (y_i - Q(s_i,a_i;\phi_c))^2$;
    Update the actor network policy using the sampled gradient;
    Update the target actor network and critic network:
      $\phi'_a \leftarrow \tau\phi_a + (1-\tau)\phi'_a$;$\phi'_c \leftarrow \tau\phi_c + (1-\tau)\phi'_c$.
  **end**
**end**

TABLE I
SIMULATION PARAMETERS

| Parameters | Value |
|---|---|
| The transmit power of UDs, $p_{user}$ | 0.2 W |
| The power of interference signal from other SBSs, $p_{in}$ | 40 mW |
| The power of interference signal from MBS, $p_{on}$ | 50 mW |
| The transmit power of SBSs, $p_e$ | 2 W |
| The size of uploading model parameter | 4500 KB |
| The computing power of MBS, $p_m$ | 5 W |
| The computing resource of MBS server, $f_m$ | [10,50] GHz |
| The maximum computation delay of UDs, $T_{max}^{cmp}$ | 0.1 s |
| Actor learning rate | 0.0001 |
| Critic learning rate | 0.0001 |
| Discount Factor, $\gamma$ | 0.97 |
| Soft update coefficient, $\tau$ | 0.001 |
| Batch size | 128 |

addition, the output layer of the Actor network adopts a softmax function, and the output layer of the Critic network is a linear neuron. The values of the network hyperparameters and other parameters are listed in Table I.

Additionally, to the best of our knowledge, there is no existing work for the communication scenarios we consider. Therefore, we cannot compare our results with other works. We illustrate the effectiveness of our scheme by comparing with the following four baseline schemes.

1) *Baseline 1:* In this scheme, we do not collect data by the DT network, i.e., consider the time delay caused by unreliable remote communications between the UDs and the server. We utilize the same parameter settings and the DDPG algorithm as the proposed scheme to solve the problem.
2) *Baseline 2:* In this scheme, there is a DT network for data collection in the data collection stage, thus the unreliable communication transmission between UDs and the server is reduced. And the traditional random allocation scheme, i.e., randomly divide the resource that need to be allocated, is used in the optimization of resource allocation. The poor channel condition between the UDs and the server may affect the FL training process. Since this scheme cannot reasonably allocate resources according to the channel conditions, the effect of resource allocation is not obvious. Other parameter settings are the same as the proposed scheme.
3) *Baseline 3:* In this scheme, there is a DT network for data collection at the data collection stage, thus the unreliable communication transmission between UDs and the server is reduced. In addition, the average resource allocation scheme, i.e., evenly distribute the resource that need to be allocated, is used in resource allocation. Since this scheme cannot reasonably allocate resource according to the channel conditions, the effect of this scheme is not obvious. Other parameter settings are the same as the proposed scheme.
4) *Baseline 4:* Inspired by [26], we use a multi-DQN network to dynamically allocate resource discretely, i.e., continuous action discretization, in this scheme. However, DQN may lead the agent to skip better actions during exploration, which can also explain why the DDPG algorithm that favors continuous action is chosen to solve our problem. The simulation parameter settings are the same as the proposed scheme.

For convenience of observation, the value of our objective function represents the total energy consumption of UDs, and it is approximately equal to the negative value of the cumulative reward of the agent. Fig. 4 shows the relationship between the total energy consumption of UDs and the training episodes with different schemes. It can be seen from the figure that the performance of the "Proposed scheme" is better than that of the "Baseline 1". This is because the DT network helps collect digital information, which can reduce the time delay of unreliable communications between the UDs and the server. Moreover, we can also know the convergence effect of the model from the figure.

In addition to comparing with "Baseline 1," we also compare our method with "Baseline 2," "Baseline 3" and "Baseline 4". As shown in Fig. 5 and Fig. 6, we compare the energy consumption with different bandwidth and different UDs transmission tolerance delay, and the "Proposed scheme" outperforms other baseline schemes. In Fig. 5, we show that the different methods have different energy consumption with the different system total bandwidth. Appropriately

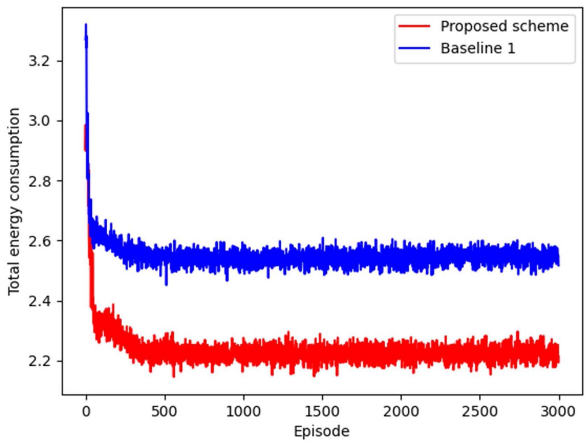


Fig. 4. Convergence diagram of different situations.

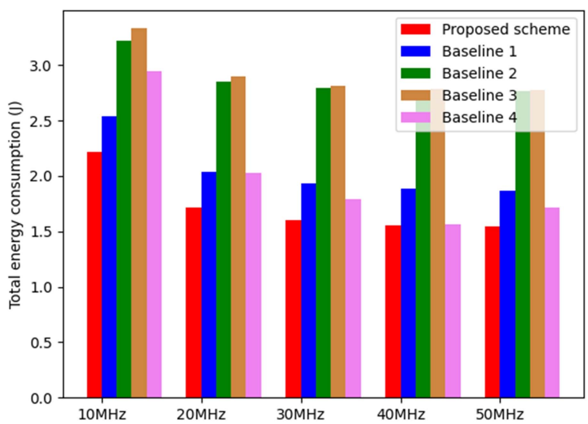


Fig. 5. Energy consumption of different system total bandwidth.

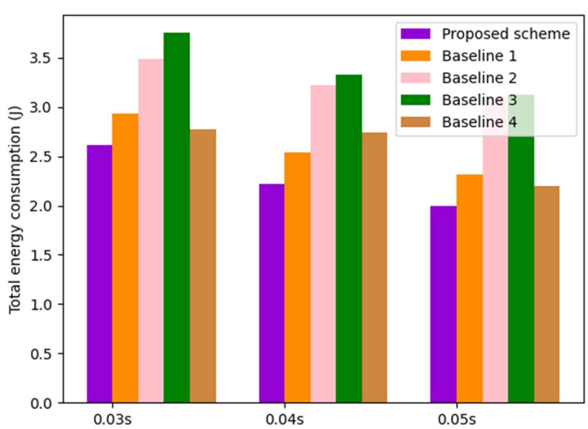


Fig. 6. Energy consumption of different transmission tolerant delay.

increasing the bandwidth for model parameters uploading can reduce the energy consumption of UDs. In Fig. 6, we can find that UDs with longer transmission tolerant delay can generate less system energy consumption, because the longer the tolerant delay has, the higher the task completion ratio and the less the penalty.

In Fig. 7, we verify the usability of the model by different number of UDs. The total energy consumption of each scheme grows as the number of UDs increases, because the resource allocated

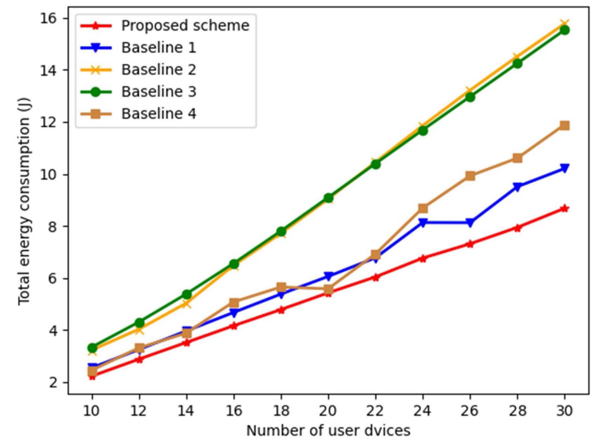


Fig. 7. Energy consumption of different user devices.

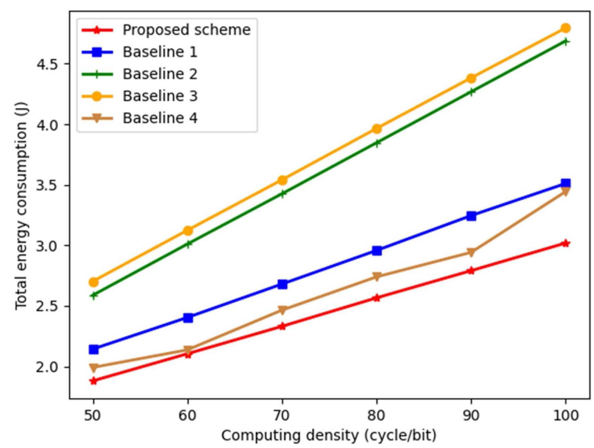


Fig. 8. Energy consumption of different computing density.

to each UD gradually decreasing with the increase of UDs under the total bandwidth $B = 10\,\text{MHz}$. Nevertheless, because of the efficient resource allocation strategy and the assistance of the DT network, our scheme still has better performance than the baseline in more UDs. Specifically, our scheme with the use of the MBS server with the DT network assists the UDs to train a local model, thus significantly reduces the local train energy consumption. In addition, our scheme can achieve a flexible resource allocation to reduce the transmission energy consumption of UDs. Therefore, compared with "Baseline 1," "Baseline 2," "Baseline 3" and "Baseline 4," our scheme achieves an energy consumption average reduction by $12.44\%$, $38.51\%$, $39.36\%$ and $16.48\%$, respectively.

As shown in Fig. 8 and Fig. 9, we can see that the total energy consumption increases with the increase of the computing density and the data size of the computational task of the UDs. With the limited resource, excessive computing density or UDs task data size can cause UDs computing burden and communication congestion. The energy consumption of each scheme increases with the increase of the computing density and the data size of the computational task of the UDs. This is because the resources of the total system have an upper limit, and the increase of the computing density and the data size of the computational task of the UDs can reduce the average acquired resource of UDs,

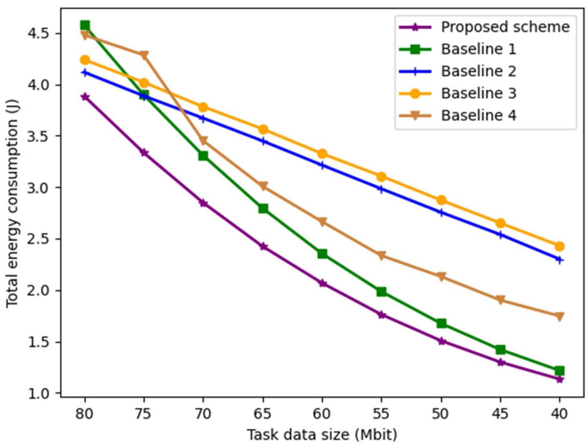


Fig. 9. Energy consumption of different user devices task data size.

which affects the final performance. In any case, our method still shows better performance because of its accurate resource allocation. Specifically, under the total bandwidth $B = 10$ MHz and ten UDs, our scheme achieves an energy consumption average reduction by 13.17%, 32.16%, 34.32% and 6.1%, respectively in Fig. 8 compared with "Baseline 1," "Baseline 2," "Baseline 3" and "Baseline 4". In addition, compared to the baseline methods, the average reduction of the energy consumption in our scheme in Fig. 9 is 11.73%, 32.60%, 35.01% and 23.9%, respectively.

## V. Conclusion

In this paper, we first considered a hierarchical FL framework in a heterogeneous cellular network scenario, and integrated DT and MEC techniques into the proposed framework. When the UDs are not in the service range of SBSs, the framework can achieve macro base stations with the DT network assisting the UDs local computation to reduce the transmission time delay. Second, a DDPG-based scheme was proposed to solve the problem of dynamic UDs-stations association and resource allocation, thereby minimizing the energy consumption within a limited delay. Finally, our method obtained better performance after being compared with the baseline methods.

## References

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.
[2] R. Shafin, L. Liu, V. Chandrasekhar, H. Chen, J. Reed, and J. C. Zhang, "Artificial intelligence-enabled cellular networks: A critical path to Beyond-5G and 6G," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 212–217, Apr. 2020.
[3] J. Konecny et al., "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:161002527*.
[4] J. Konen et al., "Federated learning: Strategies for improving communication efficiency," 2017, *arXiv:161005492*.
[5] L. Wang, W. Wang, and B. Li, "CMFL: Mitigating communication overhead for federated learning," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 954–964.
[6] F. Sattler, S. Wiedemann, K.-R. Mller, 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. 2020.
[7] W. Liu et al., "Accelerating federated learning via momentum gradient descent," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 8, pp. 1754–1766, Aug. 2020.
[8] M. Chen et al., "Convergence time minimization of federated learning over wireless networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2021, pp. 1–6.
[9] X. Xu et al., "The gradient convergence bound of federated multi-agent reinforcement learning with efficient communication," 2021, *arXiv:210313026*.
[10] C. T. Dinh et al., "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Trans. Netw.*, vol. 29, no. 1, pp. 398–409, Feb. 2021.
[11] D. Chen et al., "Matching-theory-Based low-latency scheme for multitask federated learning in MEC networks," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11415–11426, Jul. 15, 2021.
[12] J. Ren, G. Yu, and G. Ding, "Accelerating DNN training in wireless federated edge learning systems," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 219–232, Jan. 2021.
[13] C. Feng, Z. Zhao, Y. Wang, T. Q. S. Quek, and M. Peng, "On the design of federated learning in the mobile edge computing systems," *IEEE Trans. Commun.*, vol. 69, no. 9, pp. 5902–5916, Sep. 2021.
[14] K.-H. Liu, Y.-H. Hsu, W.-N. Lin, and W. Liao, "Fine-grained offloading for multi-access edge computing with actor-critic federated learning," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–6.
[15] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 35–44, 2020.
[16] T. Zhu, W. Zhou, D. Ye, Z. Cheng, and J. Li, "Resource allocation in IoT edge computing via concurrent federated reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1414–1426, Jan. 2022.
[17] W. Zhang et al., "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021.
[18] S. Luo, X. Chen, Q. Wu, Z. Zhou, and S. Yu, "HFEL: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6535–6548, Oct. 2020.
[19] E. Glaessgen and D. Stargel, "The Digit. Twin paradigm for future NASA and U.S. Air Force Vehicles," Feb. 21, 2020. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2012-1818
[20] Y. Dai, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning for stochastic computation offloading in digital twin networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 4968–4977, Jul. 2021.
[21] W. Sun, S. Lei, L. Wang, Z. Liu, and Y. Zhang, "Adaptive federated learning and digital twin for industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5605–5614, Aug. 2021.
[22] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13789–13804, Sep. 15, 2021.
[23] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12240–12251, Oct. 2020.
[24] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning for digital twin edge networks in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5709–5718, Aug. 2021.
[25] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multi-agent deep reinforcement learning for vehicular edge computing and networks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1405–1413, Feb. 2022.
[26] X. Wang, Y. Zhang, R. Shen, Y. Xu, and F.-C. Zheng, "DRL-Based energy-efficient resource allocation frameworks for uplink NOMA systems," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7279–7294, Aug. 2020.
[27] T. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, San Juan, Puerto Rico, 2016, pp. 1–14.

**Yejun He** (Senior Member, IEEE) received the Ph.D. degree in information and communication engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2005. From 2005 to 2006, he was a Research Associate with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Hong Kong. From 2006 to 2007, he was a Research Associate with the Department of Electronic Engineering, Faculty of Engineering, The Chinese University of Hong Kong, Hong Kong. In 2012, he was a Visiting Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. From 2013 to 2015, he was an Advanced Visiting Scholar (Visiting Professor) with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. Since 2011, He has been a Full Professor with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China, where he is the Director of Guangdong Engineering Research Center of Base Station Antennas and Propagation, and the Director of Shenzhen Key Laboratory of Antennas and Propagation, Shenzhen, China. He was selected as Pengcheng Scholar Distinguished Professor, Shenzhen, China, and Minjiang Scholar Chair Professor of Fujian Province. He has authored or coauthored more than 230 research papers, books (chapters) and holds about 20 patents. His research interests include wireless communications, antennas and radio frequency. He was also the recipient of the Shenzhen Overseas High-Caliber Personnel Level B (”Peacock Plan Award” B) and Shenzhen High-Level Professional Talent (Local Leading Talent). He was the recipient of the Shenzhen Science and Technology Progress Award and the Guangdong Provincial Science and Technology Progress Award in 2017 and 2018, respectively. Dr. He is currently the Chair of IEEE Antennas and Propagation Society-Shenzhen Chapter and obtained 2012 IEEE APS Outstanding Chapter Award. He was a Reviewer for various journals, such as the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, IEEE WIRELESS COMMUNICATIONS, IEEE COMMUNICATIONS LETTERS, *International Journal of Communication Systems*, *Wireless Communications and Mobile Computing*, and *Wireless Personal Communications*. He was also the Technical Program Committee Member or the Session Chair of various conferences, including the IEEE Global Telecommunications Conference, IEEE International Conference on Communications, IEEE Wireless Communication Networking Conference, and IEEE Vehicular Technology Conference. He was the TPC Chair of IEEE ComComAp 2021, General Chair of IEEE ComComAp 2019, TPC Co-Chair of WOCC 2022/2019/2015, and Organizing Committee Vice Chair of the International Conference on Communications and Mobile Computing (CMC 2010). He acted as the Publicity Chair of several international conferences, such as the IEEE PIMRC 2012. He is the Principal Investigator for more than 30 current or finished research projects, including the National Natural Science Foundation of China, Science and Technology Program of Guangdong Province, and Science and Technology Program of Shenzhen City. Dr. He is a Fellow of IET, a Senior Member of the China Institute of Communications, and a Senior Member of the China Institute of Electronics. He is an Associate Editor for IEEE NETWORK, IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, *IEEE Antennas and Propagation Magazine*, *International Journal of Communication Systems*, *China Communications*, and *Wireless Communications and Mobile Computing*.

**Mengna Yang** is currently working toward the M.S. degree in information and communication engineering with the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China. Her research interests include wireless communications, mobile edge computing, and federated learning.

**Zhou He** is currently working toward the Ph.D. degree in mechanical engineering with the University of Maryland, College Park, MD, USA. His research interests include wireless communications, antennas, and reliability of lelectronic products.

**Mohsen Guizani** (Fellow, IEEE) received the B.S. (with distinction), M.S. and Ph.D. degrees in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 1985, 1987, and 1990, respectively. He is currently a Professor of machine learning and the Associate Provost with the Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI), Abu Dhabi, UAE. He was with different institutions in the USA. He is the author of ten books and more than 800 publications. His research interests include applied machine learning and artificial intelligence, Internet of Things (IoT), intelligent autonomous systems, smart city, and cybersecurity. He was elevated to the IEEE Fellow in 2009 and was listed as a Clarivate Analytics Highly Cited Researcher in Computer Science in 2019, 2020 and 2021. Dr. Guizani was the recipient of several research awards, including the 2015 IEEE Communications Society Best Survey Paper Award, the Best ComSoc Journal Paper Award in 2021 and five Best Paper Awards from ICC and Globecom Conferences. He was also the recipient of the 2017 IEEE Communications Society Wireless Technical Committee Recognition Award, 2018 AdHoc Technical Committee Recognition Award, and 2019 IEEE Communications and Information Security Technical Recognition Award. He was the Editor-in-Chief of IEEE Network and is currently serving on the Editorial Boards of many IEEE Transactions and Magazines. He was the Chair of the IEEE Communications Society Wireless Technical Committee and the Chair of the TAOS Technical Committee. He was the IEEE Computer Society Distinguished Speaker and is currently the IEEE ComSoc Distinguished Lecturer.