Computation Offloading and Resource Allocation Based on DT-MEC-Assisted Federated Learning Framework

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

***Abstract*—Traditional centralized machine learning uses a large amount of data for model training, which may face some privacy and security problems. On the other hand, federated learning (FL), which focuses on privacy protection, also faces challenges such as core network congestion and limited mobile device (MD) resources. The computation offloading technology of mobile edge computing (MEC) can effectively alleviate these challenges, but it ignores the effect of user mobility and the unpredictable MEC environment. In this paper, we first propose an architecture that combines digital twin (DT) and MEC technologies with the FL framework, where the DT network can virtually imitate the statue of physical entities (PEs) and network topology to be used for real-time data analysis and network resource optimization. The computation offloading technology of MEC is used to alleviate resource constraints of MDs and the core network congestion. We further leverage the FL to construct DT models based on PEs' running data. Then, we jointly optimize the problem of computation offloading and resource allocation to reduce the straggler effect in FL based on the framework. Since the solution of the objective function is a stochastic programming problem, we model a Markov decision process (MDP), and use the deep deterministic policy gradient (DDPG) algorithm to solve this objective function. The simulation results prove the feasibility of the proposed scheme, and the scheme can significantly reduce the total cost by about 50% and improve the communication performance compared with baseline schemes.**

***Index Terms*—Computation offloading, resource allocation, federated learning, deep deterministic policy gradient (DDPG).**

Manuscript received 27 February 2023; revised 27 May 2023; accepted 13 July 2023. Date of publication 26 July 2023; date of current version 8 December 2023. This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62071306, and in part by Shenzhen Science and Technology Program under Grants JCYJ20200109113601723, JSGG20210802154203011, JSGG20210420091805014, and GJHZ20180418190529516. The associate editor coordinating the review of this article and approving it for publication was Q. Ye. *(Corresponding author: Yejun He.)*

Yejun He and Mengna Yang are with the State Key Laboratory of Radio Frequency Heterogeneous Integration, Guangdong Engineering Research Center of Base Station Antennas, Shenzhen Key Laboratory of Antennas and Propagation, College of Electronics and Information Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: heyejun@ieee.org; 2060432078@email.szu.edu.cn).

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

Mohsen Guizani is with the Machine Learning Department, Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE (e-mail: mguizani@ieee.org).

Digital Object Identifier 10.1109/TCCN.2023.3298926

I. Introduction

With the commercial development of the fifth-generation (5G) communication technology, the total number of mobile users in the world is increasing. According to [1], the number of mobile users will grow to nearly 7.5 billion by year-end 2025. In recent years, the success of artificial intelligence (AI) can be applied to data analysis and mining of mobile devices (MDs). Meanwhile, the emergence of mobile edge computing (MEC) alleviates the time delay caused by communicating directly with the cloud servers [2]. The emergence of these technologies has brought the dawn for large-scale commercial use of the 5G technology.

Most of the current studies use MEC to alleviate the communication time delay and energy consumption (TDEC). In other words, most current studies utilized the MEC computation technology or combined MEC and other technologies to assist user device to compute the offloading task, thereby providing high quality of service to the devices. Ding et al. [3] considered non-orthogonal multiple access (NOMA) and MEC-assisted computation of user devices, and applied geometric programming to jointly optimize time allocation and transmission power to reduce the energy consumption of computational task offloading. In [4], the authors integrated the MEC into the Internet of Things (IoT), which enables the IoT devices of limited computation capabilities and energy to offload their computation-intensive and delay-sensitive tasks to the MEC server, and maximized the energy efficiency for offloading under the maximum delay constraints of IoT devices by jointly optimizing radio and computing resource allocation. Fang et al. [5] considered a multi-user multi-base station NOMA-MEC network system with imperfect channel state information (CSI), where the MEC server can assist users to execute the task computation, and they jointly optimized the computation task assignment, power allocation and user-server association to minimize the energy consumption. Besides, they used the bi-level programming method to derive optimal closed-form expressions of task assignment and power allocation, and then designed a two-sided matching algorithm to obtain a user-server association. Hou et al. [6] proposed a joint allocation of wireless resources and MEC computing resources (JAWC) based on the spectrum radius estimation theory to reduce the total time delay of the system, where the MEC server

can assist to compute offloading by clients and the algorithms include the Spectral-Radius-based Interference Cancellation heuristic clustering algorithm and convex optimization theory. Additionally, the energy harvesting (EH) modules deployed by MDs always serve continuous task requests, and the fine-grained offloading scheme of the MEC system can greatly reduce the time delay of computing user tasks. For example, Guo et al. [7] studied the partial computation offloading scheme of multiple MDs realized by EH in MEC, and used a new algorithm based on Lyapunov optimization to obtain the optimal solution. However, the scheme only considered a single MEC server. In addition, unmanned aerial vehicle (UAV)-assisted MEC is also a promising method. In [8], the authors considered a UAV-enabled MEC system. They utilized UAVs to power IoT devices based on wireless power transfer technology, and proposed a new workflow model based on time division multiple access (TDMA) that allows parallel transmission and execution in UAV-enabled systems, and used the convex optimization theory and flow-shop scheduling techniques to jointly optimize IoT device association, computing resource allocation, UAV hovering time, wireless powering duration and IoT device service sequence to minimize the total power consumption of UAV. However, the trajectory optimization problem of UAV is not considered. In [9], [10], [11], the authors also considered UAV-assisted MEC, and combined UAV trajectory optimization and resource allocation or task offloading strategies to minimize the problem of UAV energy consumption.

On the other hand, in addition to the TDEC problem, privacy protection is also a topic of concern to users. Therefore, most studies combine federated learning (FL) [12] with MEC, that is, users only need to upload the model parameters of local training, not to share the entire dataset, which provides the possibility for privacy protection. Meanwhile, the addition of MEC servers also alleviates the core network congestion problem in cloud servers. For example, Luo et al. [13] introduced a new hierarchical federated edge learning (HEFL) framework to migrate the model aggregation from the cloud to the edge server, and jointly performed computing and resource allocation optimization in the state of end-edge association, to minimize the global cost. Wang et al. [14] theoretically analyzed the convergence bound of a distributed gradient descent (GD) and proposed a control algorithm to balance a local model update and a global parameter aggregation under a given resource budget, thereby minimizing the loss function. On this basis, the authors of [15] demonstrated the convergence of the end-edge-cloud three layer FL system. And the system can make the model converge faster to achieve a better trade-off between communication and computation. To achieve a communication-efficient FL, some studies employed deep compression techniques [16]. Mills et al. [17] proposed a communication-efficient federated averaging (FedAvg) algorithm. The algorithm is achieved by reducing the number of rounds for convergence and compression technologies. Chen et al. [18] designed a user selection scheme to select appropriate users to participate in the FL training, and combined with wireless resource allocation to improve the model convergence speed. In [19], the authors theoretically defined the impact of each round of device scheduling, and derived the convergence bounds of FL in terms of the number of communication rounds and the number of selected devices. Besides, some privacy-preserving strategies for FL, such as differential privacy, secure multi-party computation, are studied in [20], [21], [22].

Although the distributed structure of parallel training significantly improved the training efficiency, the training of FL is limited by the slowest client. To address this problem, most studies are based on asynchronous updates, so that each participant does not need to wait for others. Xie et al. [23] showed us a new asynchronous FL optimization algorithm and found a hybrid hyperparameter to control the error caused by asynchrony. In [24], [25], the authors used an asynchronous model update approach to reduce time delay due to the slowest user and improve the transmission efficiency. However, most studies on edge computing and FL are independent, and the related work about FL usually use edge servers as parameter aggregators or schedulers. The powerful computing power of the server is neglected in the FL model training. On the other hand, the computing resources of the MDs are limited. Offloading partial data to the server can effectively reduce the computing burden of the MDs. Therefore, inspired by [26], we introduced the MEC computation offloading technology to relieve the straggler effect of MDs. At the same time, in order to alleviate the problem that network dynamics become unpredictable due to the heterogeneous deployment of edge servers, we combine MEC, FL and DT. The DT module can monitor network changes in real time and provide perception data for network decision-making modules. DT is a promising technique that instantly maps PEs to digital space, and real-time capture dynamic state information of PE. In the current studies, DT has been applied in most scenarios. For example, Sun et al. [27] deployed a DT network in the system. The DT network of edge servers estimates the state of edge servers, and the trained task data for offloading strategy provided by the DT network of the entire system. Lu et al. [24] proposed an architecture of the DT edge networks (DITENs), which combined DT and MEC to efficiently optimize the industrial IoT network. In [28], the authors proposed a new DT network to model the network topology and random task arrivals in industrial IoT systems. In addition, DT has been applied in the Internet of Vehicles. Hui et al. [29] proposed a DT-enabled scheme to promote collaborative and distributed automatic driving. And the designed architecture can assist autonomous vehicles to make collaborative driving decisions. The authors of [30] proposed a DT-enabled on-demand matching scheme for multi-task FL to resolve the two-way selection problem between task requesters (TR) and the roadside units (RU) in the scenario of multi-TR and multi-RU.

In view of the above observations, our motivation is to use the DT network to build a digital model of PEs, and use the MEC computation offloading technology to solve the resource constraints of MDs. At the same time, the addition of edge servers can also reduce the pressure on the core network. We further use FL to construct the DT model based on the operating data of PEs. Specifically, we can obtain digital models of MDs and small base stations (SBSs) and collect the status

information through the DT network deployed on the macro base station (MBS) server, so as to conduct real-time data analysis and prediction. The state awareness and real-time analysis of the DT network state can help MDs make decisions, which select reliable SBS servers to offload tasks. Then SBS servers and MDs participate in the FL model training. Next, the SBS server aggregates the model parameters and uploads partial model parameters of aggregation to the MBS aggregation server for a global model training. In addition, we jointly optimize the computation offloading and resource allocation by deep reinforcement learning (DRL) algorithm to solve the problem of limited MDs resources, thus reducing the straggler effect and improving the overall efficiency of FL training. All in all, the main contributions of this paper are summarized as follows.

- Design of a DT-MEC-enabled FL framework: We propose a framework that integrates the DT with the MEC network to build a numerical model. The architecture can build a network topology and provide perception data for decision-making modules. In addition, by the DT network, the MBS server can perceive the error dataset uploaded by MDs in the FL training, and adaptively adjust the regulatory factor to reduce the failure rate of the FL model training.
- Offloading a strategy based on DT-MEC-enabled FL framework: The SBS servers utilize the idle computing power to help MDs update offloaded tasks in model training. We formulate the computational offloading problem as an optimization problem and minimize the weighted sum of time delay and energy consumption (WSTDEC) of MDs on the basis of achieving the accuracy of the FL model, i.e., the MDs cost, to obtain an optimal offloading strategy.
- Resource allocation: We adopt a deep deterministic policy gradient (DDPG), based on a deep reinforcement learning (DRL) algorithm, to solve the optimization problem of the computational offloading and the resource allocation. Numerical results show that the proposed method outperforms benchmark polices.

The remainder of this paper is organized as follows. Section II introduces the construction of the communication and computation model as well as the description of the problem. Section III presents a solution of offloading decision and resource allocation problem. Section IV gives the simulation results and analysis. A conclusion can be drawn in Section V.

II. System Models and Problem Description

We consider a multi-server, multi-user heterogeneous cellular network scenario as shown in Fig. 1. The model architecture consists of physical entities (PEs) and a DT network. Specifically, it includes a MBS *M* deploying a DT network and aggregation server (AS), *J* SBSs with servers $\{S_1, S_2, S_3, \ldots, S_j, \ldots, S_J\}$ and *I* resource-constrained MDs $\{M_1, M_2, M_3, \ldots, M_i, \ldots, M_I\}$. Assuming that each MD has an intensive task to process at each time slot, we denote the set of tasks by $\Pi_i = \{Di, Ci, t_{max}\}$. D_i represents the task

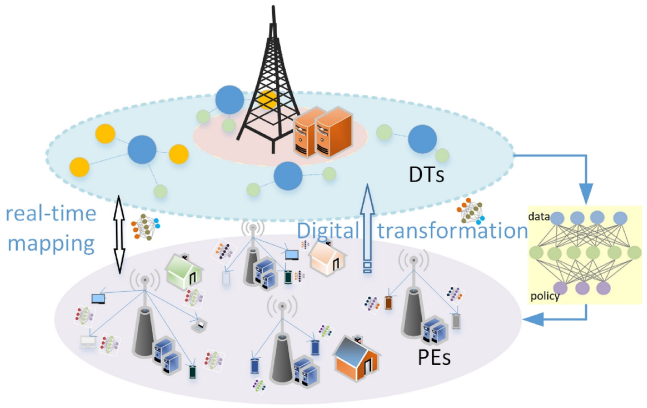


Fig. 1. DT-enabled FL-MEC system model.

size of the MD *Mi*, C_i represents the number of CPU cycles required to process one task, and let t_{max} denote the maximum tolerable delay of the MD. Note that each SBS is equipped with a server that not only has the ability to aggregate model parameters, but also has computing capabilities. In our model, we establish real time mapping and feedback between the DT network of the system and the MEC environment to obtain the transformation laws of the environment. The DT network does not need to know the implementation details of MDs and edge servers in the system, and it can build a digital representation system similar to the real environment to obtain the system performance estimate. Besides, the FL can build the DT model of the MDs according to the operation data of the MDs. And the DT model is built by the FL scheme can reduce the data transmission cost and protect the data privacy. In addition, the scheme can further adopt the MEC computation offloading technology to enable the MDs of limited computation capabilities and energy to offload their computation-intensive and delay-sensitive tasks to the network edge, thus alleviating the straggler effect and limited computation resource of MDs in FL. For convenience, the relevant notations used in this paper are summarized in Table I.

A. DT Network

Integrating the DT network into the MEC network model can monitor the state of the entire MEC network system in real time. The model can also transfer the collected data directly to decision-making modules that need to make decisions. In our system, the DT network is deployed on the MBS. The MBS can use a DT network to virtually imitate the MEC network topology, monitor the network model and obtain node parameters in real time. Besides, the MBS with the DT network can provide SBSs in the edge service layer with MDs information within SBS' service range and provide decision to help SBSs and MDs perform dynamic associations. At the same time, resource allocation strategies in the DT network can be applied to the actual network. Let Λ_{ij} denote the dynamic association between SBSs and MDs. That is, if MD M_i connects to SBS S_j, the connection can be recorded in the end-edge association set Λ_{ij}. Then, the virtual object of the DT can be represented as:

$$DT(t) = \{F(\varrho), D_i(t), f_i(t), f_j(t), \Lambda_{ij}\} \quad (1)$$

TABLE I
RELEVANT NOTATIONS

Notation	Description
D_i	The task size of the MD M_i.
C_i	The number of CPU cycles required to process one task.
t_{max}	The maximum tolerable delay of the MD.
$f_i(t)$	The computing frequency of the MD *Mi* at time slot *t*.
$f_j(t)$	The computing frequency of the SBS S_j at time slot *t*.
Λ_{ij}	The dynamic association between SBSs and MDs.
$F(\varrho)$	The global model.
$\lvert\mathfrak{D}\rvert$	The total number of training samples.
$\Theta_i, \Theta_i^{'}$	The input sample, the corresponding sample label.
ϱ_j	The model parameters in the SBS S_j.
Φ	Regulatory factor.
$\mathbf{B}$	The communication resource.
ζ_i	The communication allocation ratio for MD M_i.
ν_i	The transmission rate of the offloaded task.
Υ	The signal-noise ratio.
ι_0, α	The Rayleigh fading channel coefficient, the path loss coefficient.
N_0	The background noise power.
p_i, p_j	The transmission power of MD M_i,the transmission power of SBS S_j.
d_{ij}	The distance between the MDs and the SBSs.
h_j	The channel gain between SBS S_j and MBS.
ϑ_i	The ratio of the offloaded task of MD M_i.
t_i^{com}, e_i^{com}	The communication time delay and energy consumption of the MDs.
$t_i^{com_m}$	The time delay of the transmission model parameters between SBS S_j and MBS.
$e_i^{com_m}$	The energy consumption of the transmission model parameters between SBS S_j and MBS.
$t_j^{com_m}$	The transmission time delay between SBS S_j and MBS.
$e_j^{com_m}$	The transmission energy consumption between SBS S_j and MBS.
ϕ	The size of model parameters uploaded by MD M_i.
$\xi_i f_i$	The allocated CPU frequency ratio of MD M_i for computation.
t_i^{cmp}, e_i^{cmp}	The computation time delay and energy consumption of MD M_i.
λ_i	The effective capacitance coefficient.
ξ_i	The allocated CPU frequency of MD M_i for computation.
$E(\varphi)$	A number of local iterations to achieve a local accuracy φ.
t_j^{cmp}	The computation time delay of SBS S_j.
T, E	The total time delay and energy consumption.
$R(\psi, \varphi)$	The communication rounds to achieve the required model accuracy ψ.

where $D_i(t)$ is the set of tasks of MD M_i, $f_i(t)$ is the computing frequency of the MD *Mi*, and $f_j(t)$ is the computing frequency of the SBS S_j in a time slott. The DT builds the corresponding global model $F(\varrho)$ with the help of the FL, and continuously interacts with the actual network to maintain consistency.

B. Generation and Detection of FL Model for the DT Network

Since the edge computing (EC) can be limited by the communication resources because of uploading a bigger dataset, and the FL may be limited by computational resources due to the local updating. The above may cause the straggler effect. Therefore, the combination of the two can realize a balance between the communication and computing [26]. The goal of the FL and MEC is to build a global model for the DT network. The global model can react according to the state of the PEs, guide the dynamic association of the SBSs and MDs, judge whether the user is offline or not. Meanwhile, when the MDs upload the error dataset, the MBS with the DT network can receive the signal in time, and assist the system to adaptively adjust the model accuracy.

In the process of building the model, the role of the edge server is to assist in the computing and the model aggregation. The MD M_i offloads a part of the task to the associated SBS server S_j to compute and the remaining tasks are computed locally, and then the SBS server S_j aggregates the model parameters computed and uploaded by the MD. The SBSs server can use the aggregated model parameters to update the model. Finally, the SBS server S_j uploads the aggregated model parameters to the MBS server for the total model aggregation, and the MBS server broadcasts the aggregated model parameters to each MD M_i.

Let $\mathfrak{D} = \{\Theta_i, \Theta_i'\}_{i=1}^{|\mathfrak{D}|}$ denote the local training dataset of each MD, where Θ_i is the i-th input sample and Θ_i' is the corresponding labeled output of Θ_i for FL task. $|\mathfrak{D}|$ represents the total number of local training samples. The overall goal of the process is to minimize the global model loss $F(\varrho)$

$$F(\varrho) = \frac{1}{|\mathfrak{D}|}\sum_{j}^{J} F_j(\varrho_j) \tag{2}$$

The loss function of SBS S_j is given by

$$F(\varrho_j) = \frac{1}{\vartheta_i|\mathfrak{D}|}\sum f(\varrho_j) \tag{3}$$

where ϱ_j represents the model parameters in the SBS S_j, $\vartheta_i|\mathfrak{D}|$ denotes the size of offloading data by MD M_i. For MDs M_i, the loss function on the remaining dataset is given by

$$F(\varrho_i) = \frac{1}{(1-\vartheta_i)|\mathfrak{D}|}\sum f(\varrho_i) \tag{4}$$

After the model parameters in the MDs and SBSs are updated, the SBSs server can aggregate its own model parameters and the model parameters are uploaded by the MDs. Therefore, the global model in the SBSs server can be updated by

$$\varrho = \frac{1}{|\mathfrak{D}|}\left[\sum_{i\in I}\left((1-\vartheta_i)|\mathfrak{D}_i|\varrho_i + \vartheta_i|\mathfrak{D}_i|\varrho_j\right)\right]. \tag{5}$$

However, if the MDs offload the incorrect dataset to the SBSs, it may affect the update of the global model parameters. If the non-training dataset is denoted by $\hbar$, we have

$$\varrho' = \frac{1}{|\mathfrak{D}|}\left[\sum_{i\in I}\left((1-\vartheta_i)|\mathfrak{D}_i|\varrho_i + \vartheta_i|\mathfrak{D}_i + \hbar|\varrho_j\right)\right]. \tag{6}$$

As shown in Fig. 2, taking the MNIST dataset [32] as an example, if other datasets are uploaded, the difference between the predicted value and the actual value will become larger. According to $Loss = \frac{1}{\mathfrak{D}}\sum_{i=1}^{\mathfrak{D}}(Y - Y_p)^2$, non-training datasets' participation can increase the loss of the model training and affect the accuracy of the global model.

Remark 1 (The Convergence Performance of (6)): In our scheme, the SBSs server helps the MDs to update the model parameters together, and the scheme convergence performance is equivalent to the centralized GD algorithm. However, if the MDs upload the non-training dataset to SBSs, the accuracy of the model can be affected. After broadcasting the global model parameters, we have $\varrho_i(t) = \varrho_j(t) = \varrho(t)(t = 1, 2, 3\cdots)$.

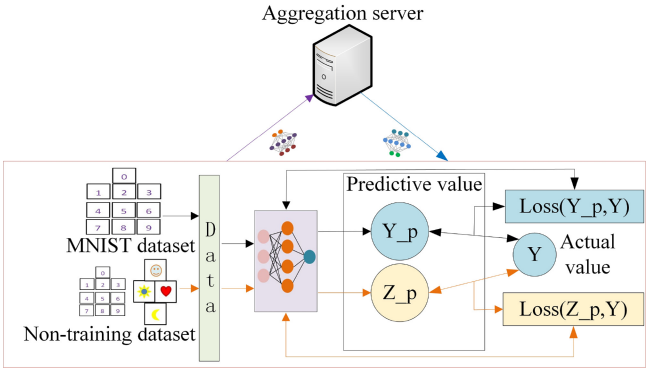


Fig. 2. The process of training MNIST dataset and non-training dataset.

Specifically, according to the above (6) and the linearity of the gradient operator, we have

$$\begin{aligned}\varrho'(t) &= \frac{1}{|\mathfrak{D}|}\left[\sum_{i\in I}\left((1-\vartheta_i)|\mathfrak{D}_i|\varrho_i(t) + \vartheta_i|\mathfrak{D}_i + \hbar|\varrho_j(t)\right)\right] \\ &= \frac{1}{\mathfrak{D}}\Bigg[\sum_{i\in I}\Big((1-\vartheta_i)|\mathfrak{D}_i|(\varrho_i(t) - \eta\nabla F(\varrho_i(t-1)) + \vartheta_i|\mathfrak{D}_i \\ &\qquad + \hbar|\left(\varrho_j(t-1) - \eta\nabla F\left(\varrho_j(t-1)\right)\right)\Big)\Bigg] \\ &= \frac{\sum_{i\in I}|\mathfrak{D}_i|\varrho(t-1) - \eta\sum_{i\in I}|\mathfrak{D}_i|\nabla F(\varrho(t-1))}{\mathfrak{D}} \\ &\quad + \frac{\sum_{i\in I}|\vartheta_i\hbar|\varrho(t-1) - \eta\sum_{i\in I}|\vartheta_i\hbar|\nabla F(\varrho(t-1))}{\mathfrak{D}} \\ &= \Gamma(\varrho(t-1) - \eta\nabla F(\varrho(t-1))\end{aligned} \tag{7}$$

where $\Gamma(\vartheta_i, \hbar) = \Phi(1 + \sum_{i\in I}\frac{\vartheta_i\hbar}{\mathfrak{D}})$, Φ is a regulatory factor. By comparing the difference between ϱ and ϱ' and adjusting the size of Φ adaptively to achieve the expected accuracy. In this case, the failure rate of the global model training can be lowered and the cost can be reduced.

C. Communication Model

The model parameter transmission and task offloading can be achieved by wireless communications between MDs and SBSs. Meanwhile, the inter-base station and the co-channel interference can be suppressed by the orthogonal frequency division multiple access (OFDMA) technology. Given the communication resource **B**, the task offloading and the transmission of model parameters need to be performed in the allocated channels. Due to the strong computing power of the MBS, we ignore the TDEC of downloading the global model. In this process, we define ζ_i as the communication allocation ratio for MD M_i, the transmission rate of the offloaded task and the model parameters from the MD M_i server to the SBS S_j is expressed as

$$\nu_i = \zeta_i \mathbf{B}\log(1 + \Upsilon_i) \tag{8}$$

where $\Upsilon = \frac{p_i|\iota_0|d_{ij}^{-\alpha}}{N_0}$ represents the signal-noise ratio. ι_0 indicates the Rayleigh fading channel coefficient and $\iota_0 \sim CN(0, 1)$ is a complex Gaussian random variable, α is the path loss coefficient, N_0 is the background noise power, p_i is the transmission power of MD M_i, and d_{ij} indicates the distance between the MDs and the SBSs. Then, the SBS server aggregates the model parameters that it calculates and the model

parameters uploaded by the MDs. Finally, the SBS server uploads the aggregated model parameters to the MBS server for a global model aggregation. The transmission rate of model parameters from the SBS server to the MBS server is given by

$$\nu_j = \sum_{i \subseteq j} \zeta_i \mathbf{B} \log\left(1 + \frac{p_j h_j}{N_0}\right) \tag{9}$$

where p_j indicates the transmission power of the SBS S_j, h_j is the channel gain between SBS S_j and MBS, and $i \subseteq j$ represents all MD M_i that select the SBS S_j.

Accordingly, considering that each (Θ_i, Θ_i') has the same size, and let ϑ_i denote the ratio of the offloaded task of MD M_i, the communication TDEC of the offloaded task between MD M_i and SBS S_j are formulated as

$$t_i^{com} = \frac{\vartheta_i |D_i|}{\nu_i} \tag{10}$$

$$e_i^{com} = p_i t_i^{com}. \tag{11}$$

At the same time, we use $t_i^{com_m}$ and $e_i^{com_m}$ to represent the TDEC of the MDs transmission model parameters, respectively. We have

$$t_i^{com_m} = \frac{\phi}{\nu_i} \tag{12}$$

$$e_i^{com_m} = p_i t_i^{com_m}. \tag{13}$$

where ϕ is the size of the model parameters for the training of the remaining tasks uploaded by MD M_i.

Additionally, the transmission TDEC of the model parameter between SBS S_j and MBS is

$$t_j^{com_m} = \frac{\phi}{\nu_j}. \tag{14}$$

$$e_j^{com_m} = p_j t_j^{com_m}. \tag{15}$$

D. Computational Model

In our system, the computational model includes the TDEC of the local computation and the computation of the SBS server. Because the server has a strong computation power, we ignore the computing energy consumption of the server.

Accordingly, the model of the computation delay and the energy consumption can be given as follows. Let $\kappa_i(1 - \vartheta_i)|D_i|$ represent the total number of CPU cycles to run one local iteration, and e denote the estimatation error. We denote the allocated CPU frequency ratio of MD M_i for computation by $\xi_i f_i$. Therefore, the estimated computation time delay of MD M_i can be formulated as

$$\widehat{t_i^{cmp}} = E(\varphi) \frac{\kappa_i (1 - \vartheta_i) |\widehat{D_i}|}{\xi_i \widehat{f_i}}, \tag{16}$$

where the real value of D_i is equal to $\frac{\widehat{D_i}}{1+e}$ and the real value of f_i is equal to $\frac{\widehat{f_i}}{1+e}$, and the estimated energy consumption of MD M_i can be given by

$$\widehat{e_i^{cmp}} = E(\varphi) \lambda_i (1 - \vartheta_i) |\widehat{D_i}| \left(\xi_i \widehat{f_i}\right)^2, \tag{17}$$

where λ_i denotes the effective capacitance coefficient, ξ_i is the allocated CPU frequency of MD M_i for computation, and the $E(\varphi) = \sigma \log(\frac{1}{\varphi})$ represents a number of local iterations to achieve a local accuracy φ [13].

Correspondingly, the estimated computation time delay of SBS S_j can be formulated as

$$\widehat{t_j^{cmp}} = E(\varphi) \frac{\kappa_j \vartheta_i |\widehat{D_i}|}{\chi_i \widehat{f_j}}, \tag{18}$$

where $\chi_i f_j$ is the computing resource allocated to MD M_i to compute the offloaded task $\vartheta_i |f_j|$ and the real value of f_j is equal to $\frac{\widehat{f_j}}{1+e}$.

At the end, the total TDEC of the system can be given by

$$T = R(\psi, \varphi) \Big(max\Big\{ max\Big\{ t_i^{com} + t_j^{cmp}, t_i^{cmp} \Big\} \Big\} + max\Big\{ t_i^{com_m} \Big\} + max\Big\{ t_j^{com_m} \Big\} \Big) \tag{19}$$

$$E = R(\psi, \varphi) \sum_i \Big(e_i^{cmp} + e_i^{com} + e_i^{com_m} + e_j^{com_m} \Big) \tag{20}$$

where $R(\psi, \varphi) = \frac{\varpi(\log(\frac{1}{\psi}))}{1-\varphi}$ is the communication rounds to achieve the required model accuracy ψ, and ϖ is a constant that depends on the learning task [13].

III. Problem Formulation and Solution

A. Problem Formulation

Considering the above computational model and the communication model, our goal is to minimize the WSTDEC of the system and achieve a model accuracy in an iterative process. By the joint optimization of variables, the MDs can reduce the straggling effect and reduce the use of electricity in the FL process. We define $\epsilon \in [0, 1]$ as the importance weighting indicator of the system TDEC. Then the optimization problem is expressed as follows:

$$P1: \min_{\varphi, \psi, \zeta_i, \xi_i, \chi_i, \vartheta_i} \epsilon * T + (1 - \epsilon) * E \tag{21}$$

$$\text{s.t. } max\Big\{ max\Big\{ t_i^{com} + t_j^{cmp}, t_i^{cmp} \Big\} \Big\} + max\Big\{ t_i^{com_m} \Big\} \le t_{max}, \tag{21a}$$

$$t_i^{com} \le t^m, \tag{21b}$$

$$\varphi, \psi \in (0, 1), \tag{21c}$$

$$f^{min} \le \xi_i f_i \le f^{max}, \tag{21d}$$

$$0 < \xi_i < 1, \tag{21e}$$

$$p^{min} \le p_i \le p^{max}, \tag{21f}$$

$$\sum_i \zeta_i = 1, \zeta_i \in (0, 1), \tag{21g}$$

$$\sum_i \chi_i f_j \le f_j, \chi_i \in (0, 1), \tag{21h}$$

$$0 < \vartheta_i < 1, \tag{21i}$$

where (21a) and (21b) represent the maximum tolerable time delay and transmission time delay constraints, respectively. Equation (21c) indicates the accuracy of the local model and

the global model. Equation (21d)-(21f) denote the computation capacity and the transmission power of MDs constraints, respectively. Equation (21g) means that the allocated communication resource cannot exceed the total bandwidth. Constraint (21h) requires that the sum of the computing resource allocated by each SBS server cannot exceed the total SBS server computing resource. Finally, our model is partial offloading, and the ratio of offloaded task size cannot exceed the limit of (21i).

Although P1 is a stochastic optimization problem which needs to determine the computation resource and communication resource at each time slot, it is difficult to solve P1 by applying the classical convex optimization algorithm. According to (19) and (20), the wireless communication resource allocation and computation resource allocation jointly determine the size of the system cost, so they are coupled together. Furthermore, the complex coupling and mixed combinatorial characteristics between the optimization variables make it difficult to solve P1. At the same time, P1 in our optimized offloading strategy is greatly related to our optimized computing resource and the communication resource. This brings challenges to designing the efficient resource allocation and offloading strategies.

The DDPG is a powerful DRL algorithm for solving a large continuous action space. Based on the Actor-Critic network, the DDPG algorithm uses the network for fitting a policy function in the action output to directly outputs the action, and can output continuous action with a large action space. In addition, the DDPG algorithm has a low optimization computational complexity and can be used to solve a mixed-integer nonlinear programming problem. Therefore, we propose a DRL-based method to solve the optimization problem of the computation offloading strategy and resource allocation strategy. Because the model parameter upload between MDs and SBS and between SBS and MBS are related to the communication resource, during the computation offloading process in a certain time slot, we can obtain a minimal TDEC of this part after the communication resources are optimized. Therefore, for ease of calculation, we simplify the P1 problem to

$$P2: \min_{\varphi,\psi,\zeta_i,\xi_i,\chi_i,\vartheta_i} \epsilon * T_{user} + (1-\epsilon) * E_{user} \tag{22}$$

$$\begin{aligned}
\text{s.t.}\ & max\Big\{max\Big\{t_i^{com}+t_j^{cmp}, t_i^{cmp}\Big\}\Big\} \\
& \le t_{max} - max\{t_i^{com_m}\}, && \text{(22a)}\\
& t_i^{com} \le t^m, && \text{(22b)}\\
& \varphi, \psi \in (0,1), && \text{(22c)}\\
& f^{min} \le \xi_i f_i \le f^{max}, && \text{(22d)}\\
& 0 < \xi_i < 1, && \text{(22e)}\\
& p^{min} \le p_i \le p^{max}, && \text{(22f)}\\
& \sum_i \zeta_i = 1, \zeta_i \in (0,1), && \text{(22g)}\\
& \sum_i \chi_i f_j \le f_j, \chi_i \in (0,1), && \text{(22h)}\\
& 0 < \vartheta_i < 1, && \text{(22i)}
\end{aligned}$$

where $T_{user} = R(\psi,\varphi)(max\{max\{t_i^{com}+t_j^{cmp}, t_i^{cmp}\}\})$ and $E_{user} = R(\psi,\varphi)(e_i^{com}+e_i^{cmp})$.

B. Selection of Local Iterations

The FedAvg algorithm [31] allows us to increase the computation by increasing the degree of parallelism and increasing the computation in each participant to reduce the number of communication rounds required to train the model. The algorithm selects a number of participants with a proportion in each iteration, and calculates the gradient and the loss function on the dataset owned by these participants. The specific process of FedAvg is as follows.

Step 1 (Initialize global model parameters): The aggregation server initializes the global model parameters and broadcasts them to randomly selected MDs. Each MD has its own local dataset.

Step 2 (Update MDs' model parameters): Each selected MD uses its own local dataset to update the received model parameters to obtain new model parameters ϱ_{t+1}^i, and send the new model parameters to the AS. The model parameters of MDs are updated by

$$\varrho_{t+1}^i \leftarrow \varrho_t^i - \varepsilon \sum_{i=1}^{I} \frac{n_i}{n} g_i \tag{23}$$

where $\sum_{i=1}^{I} \frac{n_i}{n} g_i = \nabla f(\varrho_t^i)$, ε is the learning rate, n_i is the number of samples on MD M_i, and n is all selected MDs.

Step 3 (Aggregation of the global model): The AS aggregates the model parameters uploaded by the MDs in terms of a weighted average. Then, the server can obtain a new model ϱ_{t+1}^j.

$$\varrho_{t+1}^j = \sum_{i=1}^{I} \frac{n_i}{n} \varrho_{t+1}^i \tag{24}$$

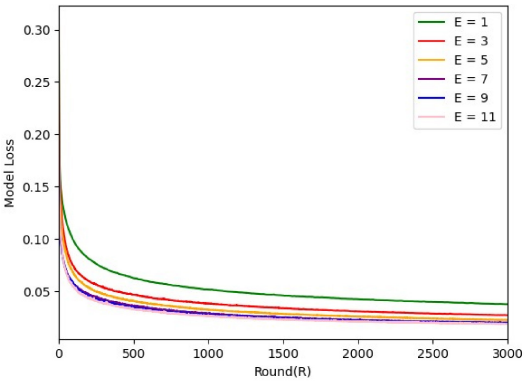
Step 4: Repeat the above steps until the model converges.

Based on the FedAvg algorithm, we take the MNIST dataset as an example for simulation. As shown in Fig. 3, we can find that under the premise of a certain number of data sets, more local rounds have a little effect on the accuracy of the model. Therefore, we have $E = 7$ to achieve the accuracy of beyond 90% by the trade-off between the local energy consumption and the model accuracy.

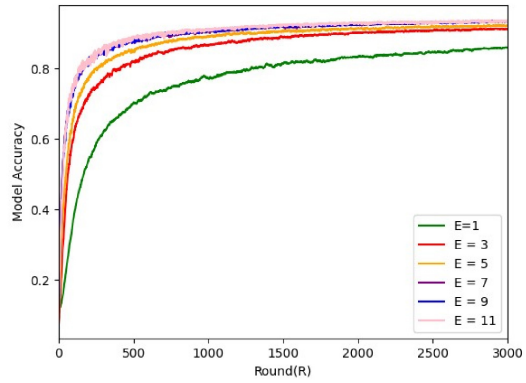
C. Optimization of the Offloading Strategy

We utilize mathematical methods to obtain the optimal solution of the offloading strategy. As we all know, in the case of a certain amount of tasks, the amount of offloaded tasks is inversely proportional to the amount of remaining tasks. Therefore, the corresponding local computation time delay t_i^{cmp} is inversely proportional to the sum of the SBS computation time delay t_j^{cmp} and the transmission time delay t_i^{com}. According to the characteristics of functions in mathematics, when $t_i^{cmp} = t_i^{com} + t_j^{cmp}$, we can obtain the minimal function value, i.e., $min\{max\{t_i^{com}+t_j^{cmp}, t_i^{cmp}\}\}$, so as to obtain the offloaded strategy that minimizes the TDEC. All in all, according to

$$\begin{aligned}
min\ & \Big\{max\Big\{t_i^{com}+t_j^{cmp}, t_i^{cmp}\Big\}\Big\}, \\
& t_i^{cmp} = t_i^{com} + t_j^{cmp},
\end{aligned} \tag{25}$$



(a) Model loss



(b) Model accuracy

Fig. 3. Model loss and accuracy of different local iterations.

we have

$$E(\varphi)\frac{\kappa_i(1-\vartheta_i)|D_i|}{\xi_i f_i} = \frac{\vartheta_i|D_i|}{\nu_i} + E(\varphi)\frac{\kappa_j\vartheta_i|D_i|}{\chi_i f_j}$$
$$\longrightarrow \vartheta_i = \frac{E(\varphi)\nu_i\chi_i f_j\kappa_i}{E(\varphi)\nu_i\chi_i f_j\kappa_i + \xi_i f_i\chi_i f_j + E(\varphi)\xi_i f_i\nu_i\kappa_j} \quad (26)$$

It can be deduced from (26) that the optimal offloading proportion is related to the allocation of the computation resources and the communication resources. Therefore, our optimization problem becomes

$$P3: \min_{\varphi,\psi,\zeta_i,\xi_i,\chi_i,\vartheta_i} \epsilon * T_{user} + (1-\epsilon) * E_{user} \quad (27)$$

$$\text{s. t.} \quad (22a)-(22h), \quad (27a)$$

$$\vartheta_i = \frac{E(\varphi)\nu_i\chi_i f_j\kappa_i}{E(\varphi)\nu_i\chi_i f_j\kappa_i + \xi_i f_i\chi_i f_j + E(\varphi)\xi_i f_i\nu_i\kappa_j} \quad (27b)$$

D. DT-Simulated Network Environment

To solve P3, we adopt the DDPG algorithm for calculation. The algorithm first models a Markov decision process (MDP) and then explores actions. The MDP is the optimal decision process of a stochastic dynamic. 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 the state transition probability. Therefore, the MDP modeling process is as follows. We use the DT network to build the network state, and then the network state is output to the agent module, as shown in Fig. 4.

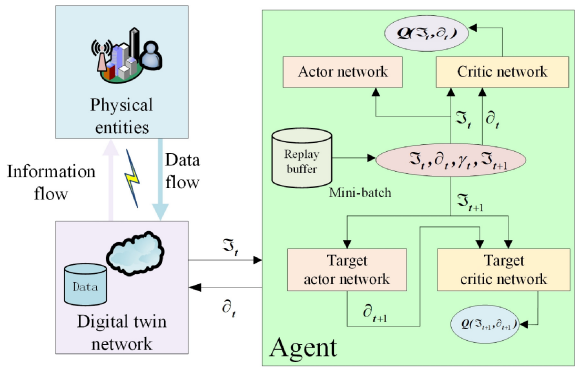


Fig. 4. DT-enabled framework of DDPG-based dynamic offloading and resource allocation.

Inspired by [28], the DT network adopts the existing K-nearest neighbor classification algorithm and the location prediction algorithm to predict the location of MDs and SBSs, which is more convenient to collect the network environment location information. After collecting the network environment information, the DT network updates the association decision between MDs and SBSs, the channel state, etc. Finally, the DT network transmits the collected state information to the DDPG agent module. In this way, the agent can obtain the state information at the beginning of time slot t, including the task size, computing frequency, CPU cycles, transmission power, channel state of MDs, and total computation and communication resources of SBSs. We have

$$\Im(\boldsymbol{t}) = \{\mathbf{D}(t), \boldsymbol{\kappa}(\boldsymbol{t}), \mathbf{F}, \mathbf{P}(t), \mathbf{H}(t), \mathbf{B}, \boldsymbol{\Lambda}(\boldsymbol{t})\} \quad (28)$$

where $\mathbf{D}(t) = [D_1, \ldots, D_i, \ldots, D_I]$ denotes $1 \times I$ task data size vector, $\boldsymbol{\kappa}(\boldsymbol{t}) = [\kappa_1, \ldots, \kappa_i, \ldots, \kappa_I]$ denotes $1 \times I$ computing density vector, and $\mathbf{P}(t) = [p_1, \ldots, p_i, \ldots, p_I]$ denotes $1 \times I$ transmission power vector at time slot t, respectively. $\mathbf{F}$ is the computation resource of SBSs; $\mathbf{B}$ is the communication resource of SBSs; $\mathbf{H}(t) = [h_{11}, \ldots, h_{1i}, \ldots, h_{1I}], \ldots, [h_{J1}, \ldots, h_{Ji}, \ldots, h_{JI}]$ denotes $J \times I$ wireless transmission channel state information matrix, and $\boldsymbol{\Lambda}(\boldsymbol{t})$ denotes $1 \times I$ set of device-server association, which is obtained by choosing the closest SBSs.

Correspondingly, the output of the DDPG network, i.e., the action of the agent, can be expressed as

$$\boldsymbol{\partial}(\boldsymbol{t}) = \{\boldsymbol{\zeta}(\boldsymbol{t}), \boldsymbol{\xi}(\boldsymbol{t}), \boldsymbol{\chi}(\boldsymbol{t})\} \quad (29)$$

where $\boldsymbol{\zeta}(\boldsymbol{t})$ is a vector that represents the allocation proportion of the communication resource, $\boldsymbol{\xi}(\boldsymbol{t})$ is a vector that represents the allocation proportion of the computation resource, and $\boldsymbol{\chi}(\boldsymbol{t})$ is a vector that represents the allocation proportion of SBSs' computation resource.

Notably, all variables in the action space are continuous. Therefore, we can exploit the policy gradient algorithm, i.e., DDPG, to explore the policy. After performing action $\boldsymbol{\partial}(\boldsymbol{t})$, the DT network estimates the immediate reward and randomly initials the state information $\Im(\boldsymbol{t}+\mathbf{1})$. And the estimates immediate reward is defined as

$$\gamma_t^e(\Im(\boldsymbol{t}), \boldsymbol{\partial}(\boldsymbol{t})) = -(\epsilon * T_{user} + (1-\epsilon) * E_{user}) \quad (30)$$

In the next time slot, the DT network transmits the updated state to the DDPG agent. Then the goal of the DDPG agent is to maximize the cumulative reward

$$\gamma_t = max\mathbb{E}\left[\sum_{t=1}^{T}\gamma_t^e(\Im(\boldsymbol{t}),\boldsymbol{\partial(t)})\right] \tag{31}$$

E. Resource Allocation Based on the DDPG Algorithm

We use the DDPG algorithm to optimize the cumulative reward. The DDPG algorithm is an off-policy algorithm based on the Actor-critic, which combines the related technologies of both the deterministic policy gradient algorithm and the deep Q-learning network (DQN) algorithm. Thus, The DDPG algorithm has a good effect in solving continuous actions, where the actor network and critic network are composed of the train network and target network, respectively. The action parameters of the actor network $\pi(\Im \mid \theta)$ need to be modified so that the actor network is more likely to obtain the maximum Q, and the actor policy is updated as

$$\nabla_\theta J(\theta) = \frac{1}{L}\sum \nabla_\partial Q(\Im_t, \partial \mid \mu)\mid_{\partial=\pi(\Im_t)} \nabla_\theta \pi(\Im_t \mid \theta) \tag{32}$$

The update of the critic network draws on the methods of DQN and double Q-learning, where two neural networks are used for calculating Q. And the critic target network generates Q^{tg} according to the next state and the next action. Then, the critic network's weights μ are updated by minimizing a loss function

$$Loss = \frac{1}{L}\sum(y_t - Q(\Im_t, \partial_t \mid \mu))^2 \tag{33}$$

where $y_t = \gamma_t^e + \delta Q^{tg}(\Im_{t+1}, \pi^{tg}(\Im_{t+1} \mid \mu^{tg}) \mid \theta^{tg})$. L is the mini-batch and δ is the discount factor. Finally, target networks are updated by

$$\begin{aligned}\theta^{tg} &\leftarrow \tau\theta + (1-\tau)\theta^{tg}\\ \mu^{tg} &\leftarrow \tau\mu + (1-\tau)\mu^{tg}\end{aligned} \tag{34}$$

where τ is the soft update coefficient. And the specific algorithm is shown in Algorithm 1.

Generally, the main computational complexity comes from the training and inference of the neural network and the operation of experience playback in the DDPG algorithm, both the Actor and Critic network use neural network. Assuming that the input dimension of the Actor network is *D*, the number of neurons in the hidden layer is *H*, the output dimension is *A*, the input dimension of the Critical network is $D + A$, the number of neurons in the hidden layer is *H*, and the output dimension is 1. And assuming that the gradient descent algorithm is used for optimization, and the number of iterations for optimization is *N*. The computational complexity of the training and inference of the neural network is: Actor network is $\mathcal{O}(N * D * H + H * A)$ and Critic network is $\mathcal{O}(N * (D + A) * H + H)$. In addition, assuming that the size of the experience pool is *M* and the number of samples sampled from it is *B* each time, the computational complexity of the experience playback operation is: complexity of storing experience samples is $\mathcal{O}(M * (D + A + 1))$ and complexity of the random sampling from experience pool is $\mathcal{O}(B * (D + A + 1))$.

Algorithm 1: Task Offloading and Resource Allocation Based on the DDPG Algorithm

```
Initialize actor-critic network with θ and μ,
and target actor-critic network with θ^tg and μ^tg.
Initialize replay buffer R^b.
for episode = 1:E^m do
    Initialize a random process Ω for ∂ to explore.
    Receive initial state information ℑ_1 from DT
    network.
    for step = 1:T do
        According to the current policy and added
        exploration noise 𝒩 to choose resources action
        ∂_t.
        Execute resource action ∂_t and reward γ_t^e and
        new state information ℑ_{t+1} from DT network.
        Store (ℑ_t, ∂_t, γ_t^e, ℑ_{t+1}) in R^b.
        Sample a random mini-batch of L
        (ℑ_i, ∂_i, γ_i^e, ℑ_{i+1}) from R^b.
        Set y_i, update critic network by minimizing the
        loss (33),
        and use the sampled policy gradient to update the
        actor network policy by (32). Update the target
        actor-critic networks by (34).
    end
end
```

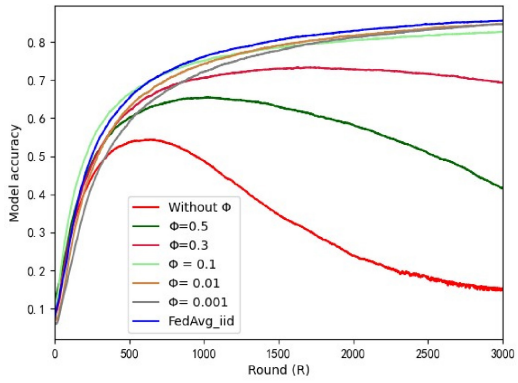
IV. Simulation Results and Analysis

In this section, we present the simulation results on a real dataset and a heterogeneous dataset. As for the real dataset, we choose the MNIST dataset to evaluate the convergence speed and accuracy of the model. As for the heterogeneous dataset, we use synthetic datasets to represent the heterogeneity between the data, where the datasets are used to evaluate the TDEC minimization problems of our model. In a heterogeneous dataset, the dataset size among MDs follows the uniform distribution of $D_i \in [100, 150]$ KBytes.

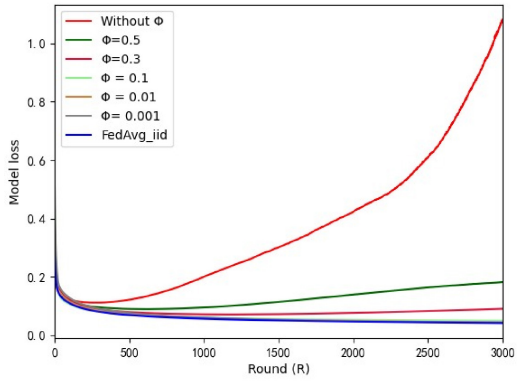
A. Regulatory Factor Effect on FL

We take a dataset of handwritten digits, i.e., MNIST, and change the number of samples among 100 MDs. Based on the experimental setup of [31] and [33], we use a deep neural network (DNN) with two fully connected layers to build the model. We follow the method in [26] to split the MNIST data unbalanced into IID and non-IID. For IID, the entire dataset is shuffled and then divided into 100 parts. For non-IID, we first sorted the dataset by numerical label, divided it into 200 shards of size 300, and randomly allocated 2 shards to each of the 100 MDs. Referring to the original FL scheme, we may randomly select some MDs to participate in the FL training.

We perform simulations on the basis of the FedAvg algorithm. Suppose the number of the uploaded non-training sample as $\hbar = 10$. As shown in Fig. 5 and Fig. 6, the different performances are achieved by the different values of regulatory factor Φ in the iid and non-iid, respectively, where the "Without Φ" scheme illustrates that if MDs upload non-training data sets, the FL model can result in a failure to train

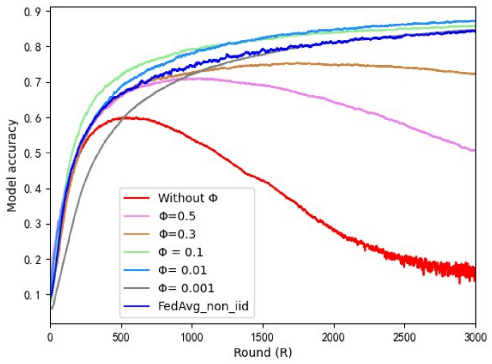


(a) Model accuracy

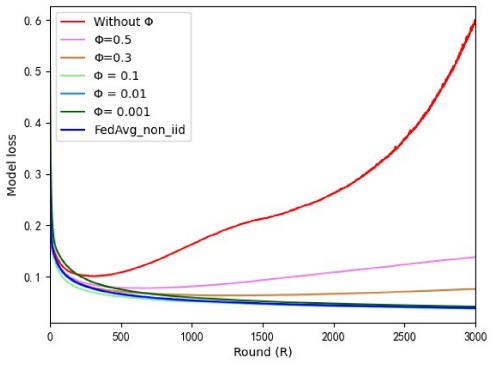


(b) Model loss

Fig. 5. The convergence performance of proposed scheme in IID.



(a) Model accuracy



(b) Model loss

Fig. 6. The convergence performance of the proposed scheme in non-IID.

and produce some system cost. However, if an appropriate regulatory factor Φ is selected, the model accuracy can reach the expected target value.

TABLE II
SIMULATION PARAMETERS

Parameter	Value
The transmission power of MDs, p_i	[0.1,0.5] W
Effective capacitance coefficient, κ_i	10^{-28}
Computation resource	15 GHz
Actor network learning rate	0.0001
Critic network learning rate	0.0001
Discount factor, δ	0.9
Batch size	128
Soft update coefficient, τ	0.01

B. Performance of Computation Offloading and Resource Allocation

1) Simulation Settings: We set up the simulation parameters for the performance evaluation of the system model as follows. We have one MBS with AS, and four SBSs with AS and a computation server. In addition, the total number of MDs with computation capabilities is set as $I = 12$. Assuming that all channels follow the Rayleigh slow fading, the path loss coefficient is set to $\alpha = 3$. The communication source is set to $\mathbf{B} = 2 \times 10^7$ Hz. And the background noise power is set as $N_0 = 10^{-12}$ W. Meanwhile, the computing density of MDs follows a uniform distribution within [300, 500] cycles/bit. The minimum and maximum computing resources of MDs are set to 0.1 and 1 (GHz).

We perform the simulation results using Python 3.7, Parl 1.3.1 and Paddlepaddle 1.6.3. The DDPG algorithm parameters are as follows: Actor-Critic network has two fully connected hidden layers with 64 neurons and adopts the ReLU function as the activation function. Then, we set the input dimension of the Actor network to $10 \times I$. "10" denotes the sum of $4 \times I$ and $6 \times I$, where the $4 \times I$ denotes the channel gain between UDs and SBSs, and $6 \times I$ represents other six-state dimensions. In addition, the output layer of the Actor network adopts a sigmoid function, and the output layer of the Critic network is a linear neuron. The values of network hyperparameters and other parameters are listed in Table II.

2) Performance and Discussion: To the best of our knowledge, there is no existing work that considers our proposed communication scenarios. Therefore, we cannot compare our research with similar research attempts. We compare the proposed scheme with other baseline schemes as follows.

The convergence graph under several different schemes is shown in Fig. 7. For the sake of observation, the value of our P3 objective function represents the total cost of MDs, and the total cost of MDs is equal to the negative value of the cumulative reward of the agent. In Fig. 7, the "Proposed scheme without B & F" represents the joint optimization of the total computation resource allocation of SBSs and computation offloading strategy; the "Proposed scheme without Fs" represents the joint optimization of computation offloading strategy, the communication resource and the local computation resource. We can see that all three converge, and in terms of the total cost, the "Proposed scheme" scheme outperforms both benchmarks because it can simultaneously

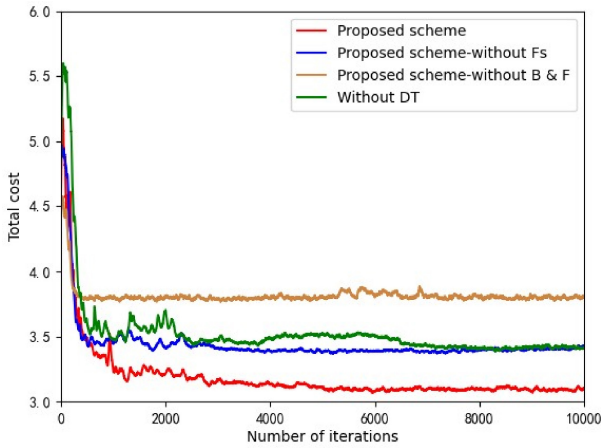


Fig. 7. Total cost of MDs with different schemes.

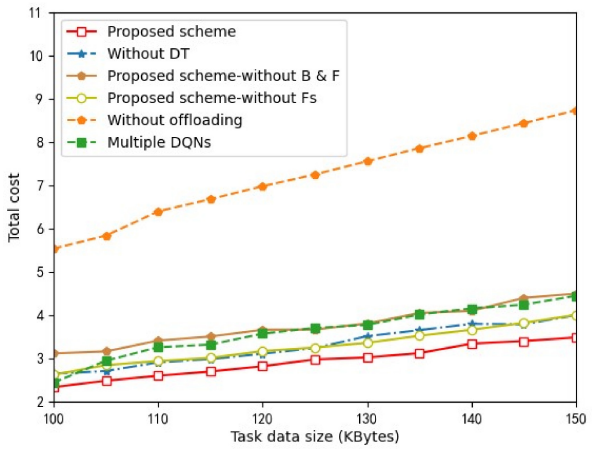


Fig. 9. Total cost of MDs v.s. different task data size.

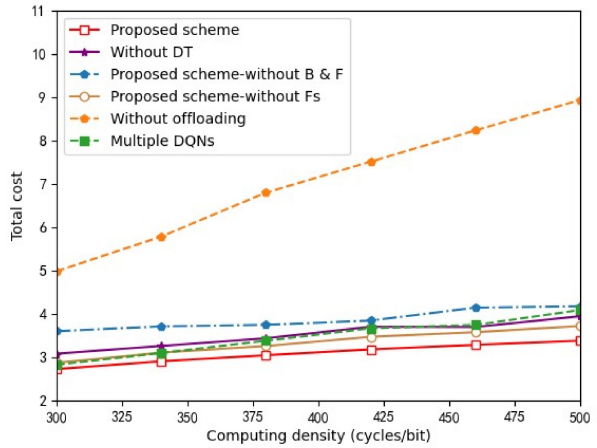


Fig. 8. Total cost of MDs v.s. different computing density.

optimize the computation offloading strategy, the communication resource, the local computation resource, and the total computation resource of SBSs. In addition, the "Proposed scheme" is better than "Without DT", because the DT network helps collect the digital information, which can reduce the time delay of unreliable communications between the MDs and the server, where "Without DT" means disabling the DT network.

Fig. 8 and Fig. 9 show the impact of the computational load on the system performance. The total cost of MDs under all schemes increases with increasing the computing density and the task data size. As we can see, with the resources, too high computing density and task data size can reduce the average acquired resource of MDs, causing a computational burden and a communication congestion, thus affecting the final performance. In any case, our scheme still shows a better performance because of the joint optimization of the computational offloading strategy and resource allocation. Specifically, in order to achieve a model accuracy of beyond 90%, we define the number of local rounds E=7 in the FL process. And under the total communication resource $\mathbf{B} = 2 \times 10^7$ Hz and the total computation resource of the SBSs $\mathbf{F} = 15 \times 10^9$ Hz, our scheme achieves a consumption cost average reduction by 20%, 7.4%, 56% and 11% as shown in Fig. 8 compared with the "Proposed scheme without B & F", "Proposed scheme without Fs", "Without offloading" and "Multiple DQNs" algorithm, respectively. Meanwhile, our scheme also shows a better performance in Fig. 9, where "Without offloading" is based on the traditional FL, that is, all tasks of all users are calculated locally, instead of using an edge server to assist in the computation. Moreover, because our proposed scheme introduces a DT network, no matter how the computing intensity and task data size change in Fig. 8 and Fig. 9, the cost of the "Proposed scheme" is always lower than that of the "Without DT" scheme. Because the DT-disabled offloading module always estimates the information of candidate edge servers through queries, and the DT-enabled scheme only needs to store the digital status of these servers to estimate their the computing capacity. Additionally, inspired by [34], we use a multi-DQN network to dynamically allocate resources discretely, i.e., continuous action discretization. The DQN algorithm plays an important role in solving discrete actions and reducing the output dimension. However, the simulation results show that the "Multiple DQNs" scheme is not the best choice for solving our problem. Fig. 8 shows that the DDPG algorithm favors the continuous action to our proposed scheme.

The estimation error of the DT network may be affected by many factors, such as the uncertainty of the model assumptions, data quality, changes in the system operating environment, the complexity of the model and the amount of the training data. We assume that the error is 1%, 5% and 10%, respectively. From Fig. 10, we can see that even if there is an estimation error between the estimated value and the real value, the performance result of the model is still better than that of the DT-disabled technology, and even very close to the real value when the estimation error is small. And the simulation results indirectly prove the feasibility of the scheme.

Fig. 11 shows that different schemes achieve different total costs with increasing MDs. The number of MDs ranges from 6 to 24. Under the system communication resource $\mathbf{B} = 2 \times 10^7$ Hz and the total computation resource of SBSs $\mathbf{F} = 15 \times 10^9$ Hz, we can observe that with the increase of the number of MDs, the total cost of six schemes also increases. The reason is that the

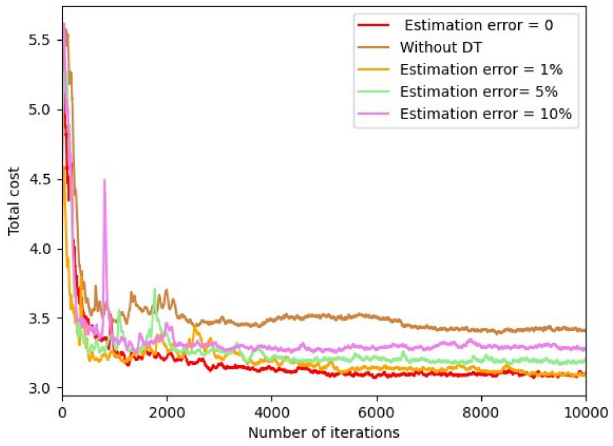


Fig. 10. Total cost of MDs with different estimation error.

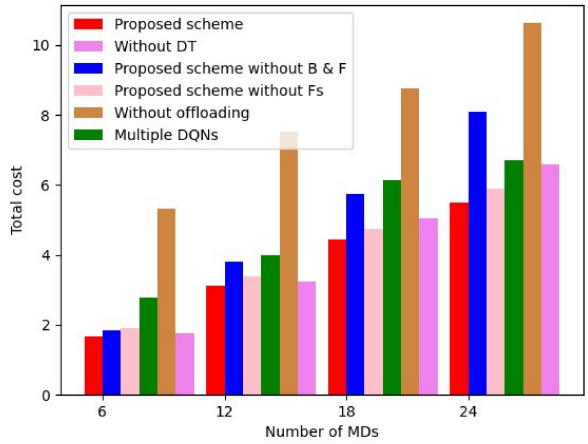


Fig. 11. Comparison of the total cost with the varying MD.

growth of MDs can lead to more offload requirements, thus consuming more communication resource and computation resource. Additionally, the "Proposed scheme" algorithm still outperforms other benchmarks, where the "Proposed scheme" algorithm performance is better than that of the "Multiple DQNs" algorithm. The main reason is that the discretization of continuous actions in DQN may lead the agent to skip better actions during the exploration. This is why we choose the DDPG algorithm instead of the DQN algorithm. Obviously, the algorithm performance of the "Proposed scheme" is better than that of "Without offloading" about 50% because the local computation resource of MDs are limited. If the MEC computing technology is not used to assist the MDs in task computation, it can cause a large time delay and energy consumption, and even increase the failure rate of a task training. In addition, with the increase of the number of MDs, the performance of the 'proposed scheme' is still better than that of "without DT" under the condition of jointly optimizing the computation offloading strategy and resource allocation strategy. And the more MDs under the condition that the number of servers is constant, the more obvious the performance effect (the proposed scheme is better than the scheme without DT). This is the reason why the number of MDs increases, the number of individuals who need to make decisions can increase, accordingly, the total time delay of offloading the module can also increase, because the offloading module needs to estimate the servers information by query.

V. Conclusion

In this paper, we presented a DT-MEC-assisted FL framework, where the DT network can provide input parameters for the decision module (DRL agent) and identify for non-training the dataset uploading in the FL process. Then, we presented the problem of the computational offloading and resource allocation. The stragglers can offload some tasks to the SBSs to reduce the computational burden. We jointly optimized the offloading strategy, the computation resource and the communication resource to minimize the total system cost and greatly improve the FL training performance. Since the objective function is a non-convex stochastic programming problem, we used the DDPG algorithm to solve the problem of resource allocation and computational offloading. Experimental results show the effectiveness of our scheme and the proposed scheme outperformed baseline algorithms. In addition, the cost of DT deployment involves many aspects, such as the cost of data collection and preprocessing, algorithm selection and optimization, hardware and software, human and material resources in deployment and maintenance. Therefore, it is meaningful to consider the performance cost trade-off of DT in our future work.

References

[1] (Radicati Group, Inc., Palo Alto, CA, USA). *Mobile Statistics Report, 2021–2025*. (Jan. 2021). [Online]. Available: https://www.radicati.com/wp/wp-content/uploads/2021/Mobile_Statistics_Report,_2021-2025_Executive_Summary.pdf

[2] 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.

[3] Z. Ding, J. Xu, O. A. Dobre, and H. V. Poor, "Joint power and time allocation for NOMACMEC offloading," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6207–6211, Jun. 2019.

[4] B. Liu, C. Liu, and M. Peng, "Resource allocation for energy-efficient MEC in NOMA-enabled massive IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 4, pp. 1015–1027, Apr. 2021.

[5] F. Fang, K. Wang, Z. Ding, and V. C. M. Leung, "Energy-efficient resource allocation for NOMA-MEC networks with imperfect CSI," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 3436–3449, May 2021.

[6] Y. Hou, C. Wang, M. Zhu, X. Xu, X. Tao, and X. Wu, "Joint allocation of wireless resource and computing capability in MEC-enabled vehicular network," *China Commun.*, vol. 18, no. 6, pp. 64–76, Jun. 2021.

[7] M. Guo, W. Wang, X. Huang, Y. Chen, L. Zhang, and L. Chen, "Lyapunov-based partial computation offloading for multiple mobile devices enabled by harvested energy in MEC," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 9025–9035, Jun. 2022.

[8] Y. Du, K. Yang, K. Wang, G. Zhang, Y. Zhao, and D. Chen, "Joint resources and workflow scheduling in UAV-enabled wirelessly-powered MEC for IoT systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 10187–10200, Oct. 2019.

[9] X. Qin, Z. Song, Y. Hao, and X. Sun, "Joint resource allocation and trajectory optimization for multi-UAV-assisted multi-access mobile edge computing," *IEEE Wireless Commun. Lett.*, vol. 10, no. 7, pp. 1400–1404, Jul. 2021.

[10] D. Wang, J. Tian, H. Zhang, and D. Wu, "Task offloading and trajectory scheduling for UAV-enabled MEC networks: An optimal transport theory perspective," *IEEE Wireless Commun. Lett.*, vol. 11, no. 1, pp. 150–154, Jan. 2022.

[11] B. Liu, Y. Wan, F. Zhou, Q. Wu, and R. Q. Hu, "Resource allocation and trajectory design for MISO UAV-assisted MEC networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 4933–4948, May 2022.

[12] J. Konecny et al., "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:161002527*.

[13] 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.

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

[15] L. Liu, J. Zhang, S. H. Song, and K. B. Letaief, "Client-edge-cloud hierarchical federated learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6.

[16] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *Proc. ICLR*, 2016, pp. 1–14.

[17] J. Mills, J. Hu, and G. Min, "Communication-efficient federated learning for wireless edge intelligence in IoT," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 5986–5994, Jul. 2020.

[18] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, Apr. 2021.

[19] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless federated learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 453–467, Jan. 2021.

[20] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Differentially private asynchronous federated learning for mobile edge computing in urban informatics," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2134–2143, Mar. 2020.

[21] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-preserving federated learning framework based on chained secure multi-party computing," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6178–6186, Apr. 2021.

[22] C. Zhou, A. Fu, S. Yu, W. Yang, H. Wang, and Y. Zhang, "Privacy-preserving federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 11, pp. 10782–10793, Nov. 2020.

[23] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization, 2019, *arXiv:1903.03934*.

[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] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2276–2288, Feb. 2021.

[26] Z. Ji, L. Chen, N. Zhao, Y. Chen, G. Wei, and F. R. Yu, "Computation offloading for edge-assisted federated learning," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9330–9344, Sep. 2021.

[27] 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.

[28] 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.

[29] Y. Hui et al., "Collaboration as a service: Digital-twin-enabled collaborative and distributed autonomous driving," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 18607–18619, Oct. 2022.

[30] Y. Hui et al., "Digital twins enabled on-demand matching for multi-task federated learning in HetVNets," *IEEE Trans. Veh. Technol.*, vol. 72, no. 2, pp. 2352–2364, Feb. 2023.

[31] H. B. Mcmahan et al., "Federated learning of deep networks using model averaging," 2016, *arXiv:1602.05629*.

[32] 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.

[33] X. Li et al., "On the convergence of FedAvg on non-IID data," 2019, *arXiv:1907.02189*.

[34] 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.

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, The 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 2006, he has been with Shenzhen University, Shenzhen, China, where he is a Full Professor with the College of Electronics and Information Engineering, the Director of Guangdong Engineering Research Center of Base Station Antennas and Propagation, and the Director of the Shenzhen Key Laboratory of Antennas and Propagation. From 2023 to 2024, he is an Advanced Research Scholar (Visiting Professor) with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. He has authored or coauthored over 280 research papers, seven books, and holds about 20 patents. His research interests include wireless communications, antennas, and radio frequency.

Dr. He was also a 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 selected as a Pengcheng Scholar Distinguished Professor, Shenzhen, and the Minjiang Scholar Chair Professor of Fujian Province in 2020 and 2022, respectively. He received the Shenzhen Science and Technology Progress Award in 2017, and has earned the Guangdong Provincial Science and Technology Progress Award for two times in 2018 and 2023, respectively. He obtained the IEEE APS outstanding Chapter Award in 2022. He has served as a Reviewer for various journals, such as the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, the IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, the IEEE WIRELESS COMMUNICATIONS, the IEEE COMMUNICATIONS LETTERS, and the *International Journal of Communication Systems*. He has also served as a Technical Program Committee Member or the Session Chair for various conferences, including the IEEE Global Telecommunications Conference, the IEEE International Conference on Communications, the IEEE Wireless Communication Networking Conference, and the IEEE Vehicular Technology Conference. He served as the TPC Chair of IEEE ComComAp 2021 and the General Chair of IEEE ComComAp 2019. He is the Principal Investigator for over 40 current or finished research projects, including the National Natural Science Foundation of China, the Science and Technology Program of Guangdong Province, and the Science and Technology Program of Shenzhen City. He is currently serving as an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION, IEEE ANTENNAS AND WIRELESS PROPAGATION LETTERS, *IEEE Antennas and Propagation Magazine*, *International Journal of Communication Systems*, *China Communications* as well as *ZTE Communications*. He is the Chair of the IEEE Antennas and Propagation Society-Shenzhen Chapter. He has been a Fellow of IET since 2016, a Senior Member of the China Institute of Communications as since 2007 as well as a Senior Member of the China Institute of Electronics since 2011.

Mengna Yang received the M.S. degree in information and communication engineering from the College of Electronics and Information Engineering, Shenzhen University, Shenzhen, China, in 2023. Her research interests include wireless communications, mobile edge computing, and federated learning.

Zhou He is currently pursing the Ph.D. degree in mechanical engineering with the University of Maryland at College Park, College Park, USA. His research interests include wireless communications, antennas, and reliability of electronic 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 with the Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE. Previously, he worked in different institutions in the USA. He is the author of 11 books, more than 1000 publications and several U.S. patents. His research interests include applied machine learning and artificial intelligence, smart city, Internet of Things, intelligent autonomous systems, and cybersecurity. He was listed as a Clarivate Analytics Highly Cited Researcher in Computer Science in 2019, 2020, 2021, and 2022. He has won several research awards, including the "2015 IEEE Communications Society Best Survey Paper Award," the Best ComSoc Journal Paper Award in 2021 as well 5 Best Paper Awards from ICC and Globecom Conferences. He is also the recipient of the 2017 IEEE Communications Society Wireless Technical Committee Recognition Award, the 2018 AdHoc Technical Committee Recognition Award, and the 2019 IEEE Communications and Information Security Technical Recognition Award. He served as 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 served as the IEEE Computer Society Distinguished Speaker and is currently the IEEE ComSoc Distinguished Lecturer.