Distributed Machine Learning for Multiuser Mobile Edge Computing Systems

Yinghao Guo, Rui Zhao, Shiwei Lai, Lisheng Fan, *Member, IEEE*, Xianfu Lei, and George K. Karagiannidis, *Fellow, IEEE*

***Abstract*—In this paper, we investigate a distributed machine learning approach for a multiuser mobile edge computing (MEC) network in a cognitive eavesdropping environment, where multiple secondary devices (SDs) have some tasks with different priorities to be computed. The SDs can be allowed to use the wireless spectrum as long as the interference to the primary user is tolerated, and an eavesdropper in the network can overhear the confidential message from the SDs, which threatens the data offloading. For the considered system, we firstly present three optimization criteria, whereas *criterion I* aims to minimize the linear combination of latency and energy consumption, *criterion II* tries to minimize the latency under a constraint on the energy consumption, and *criterion III* is to minimize the energy consumption under a constraint on the latency. We then exploit a federated learning framework to solve these optimization problems, by optimizing the offloading ratio, bandwidth and computational capability allocation ratio. Simulation results are finally demonstrated to show that the proposed method can effectively reduce the system cost in terms of latency and energy consumption, and meanwhile ensure more bandwidth and computational capability allocated to the user with a higher task priority.**

***Index Terms*—Computational offloading, federated learning, resource allocation, task priority.**

I. Introduction

A. Literature Review

IN RECENT years, the emerging paradigm of Internet of Things (IoT) has attracted much attention from academy and industry, since it can fulfill the intelligent identification, positioning, tracking, etc. To support various applications of IoT, mobile edge computing (MEC) has been proposed to overcome the limitations of traditional cloud computing [1]–[3]. In the MEC networks, mobile users can offload the computational tasks to the nearby computational access points (CAPs), in order to meet the requirement of latency and energy consumption. Hence, offloading strategy has become a key design in MEC networks [4]–[6]. In general, computational offloading can be divided into two types of *binary offloading* and *partial offloading*. In binary offloading, the tasks from the users should be computed either locally or by the CAPs through offloading [7]. In [8], the authors studied a binary offloading and user attack model, and then designed an offloading strategy based on the Lyapunov optimization framework to protect the user privacy and ensure the user experience. In the partial offloading, some parts can be computed locally, while the rest parts are computed by the CAPs through offloading [9]. For massive multiple-input multiple-output (MIMO) MEC networks, a partial offloading strategy can be designed by taking into account the latency constraint, through a nested algorithm based on convex optimization [10]. Moreover, the technique of intelligent reflecting surfaces (IRS) technique can be integrated into the MEC networks to enhance the system transmission performance significantly, where the key system design on the IRS was given in the works [11]–[13].

The development of machine learning algorithms such as deep learning (DL) and deep reinforcement learning (DRL) provides a novel way to solve the problems in wireless communication systems, which has attracted a lot of attention [14]–[20]. For example, the authors in [14] proposed a novel framework based on deep neural network (DNN) to address the problem of antenna selection and power allocation in the wireless communication systems. By mapping the hybrid beamforming architecture to a neural network (NN), [15] presented a novel design scheme based on the machine learning for multiple beamforming architectures, which provides critical guidance in the design of intelligent radio networks. The authors in [16] developed two novel antenna selection methods based on the support vector machine and DNN, in order to solve the antenna selection problem in full-duplex spatial modulation system. In addition, for vehicular fog computing networks, a DRL-based task offloading method was proposed by jointly exploiting the task priority, mobility of vehicles and service availability [17]. Although the aforementioned researches can find a proper offloading strategy for the MEC networks, they are centralized optimization in

Manuscript received July 21, 2021; revised November 23, 2021 and December 21, 2021; accepted December 26, 2021. Date of publication January 5, 2022; date of current version May 17, 2022. This work was supported in part by NSFC under Grant 61871139 and Grant 62101145, in part by the Natural Science Foundation of Guangdong Province under Grant 2021A1515011392, and in part by the Research Program of Guangzhou University under Grant YJ2021003. The work of Xianfu Lei was supported in part by the National Key R&D Program of China under Grant 2019YFB1803400, in part by the National Natural Science Foundation of China under Grant 61971360, in part by the Fundamental Research Funds for the Central Universities under Grant XJ2021KJZK007, and in part by the open research fund of National Mobile Communications Research Laboratory, Southeast University under Grant 2021D05. The guest editor coordinating the review of this manuscript and approving it for publication was Prof. Ping Yang. *(Corresponding authors: Lisheng Fan; Xianfu Lei.)*

Yinghao Guo, Rui Zhao, Shiwei Lai, and Lisheng Fan are with the School of Computer Science, Guangzhou University, Guangzhou 510006, China (e-mail: 2111906042@e.gzhu.edu.cn; ruizhao@e.gzhu.edu.cn; shiweilai@e.gzhu.edu.cn; lsfan@gzhu.edu.cn).

Xianfu Lei is with the Provincial Key Lab of Information Coding and Transmission, Southwest Jiaotong University, Chengdu 610031, China, and also with National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: xflei@swjtu.edu.cn).

George K. Karagiannidis is with the Aristotle University of Thessaloniki, 54636 Thessaloniki, Greece (e-mail: geokarag@auth.gr).

Digital Object Identifier 10.1109/JSTSP.2022.3140660

essence, which requires to know the full information of all users. This is however harmful to the privacy protection, and meanwhile causes a high communication overhead.

Some distributed learning algorithms have been proposed to solve the optimization problems in wireless networks [21]–[23], by taking into account the following advantages of the distributed learning framework. Firstly, the distributed learning allows mobile devices not to upload their full information to the CAPs, so as to protect the privacy of the mobile devices. Secondly, the distributed learning can help reduce the large communication overhead in the process of centralized learning. Thirdly, unlike the centralized learning, the distributed learning provides a flexible way to the system design, thanks to its characteristic of multi-module cooperation. Due to these reasons, distributed optimization methods are more attractive in solving the optimization problems of the MEC networks. For example, a fully distributed computation offloading algorithm was devised to reduce the average system-wide execution cost in [21], and a distributed branch-and-bound method could be applied to minimize the long-term average delay subject to the constraints of computation resources and power consumption [22]. However, it is notable that there still exist some challenges in the conventional distributed learning. One challenge is the distribution characteristics of data, where the data is often assumed to follow the independent and identical distribution (IID). This however may not hold in the practical MEC networks, where the task characteristics of mobile devices are varying and non-independently and identically distributed (nIID). Another challenge is the lack of interaction among users, which may cause the whole system to lose robustness and result in a high exploration cost.

To solve the problems in the conventional distributed learning, Google researchers proposed a pioneering concept of federated learning (FL) to break the data island in the machine learning [24]–[27], which puts the research of machine learning to a new altitude. In general, the process of FL contains four steps. Firstly, each user trains its own data at local, and then uploads the trained network parameters to a central node. Secondly, according to the federated learning framework, the central node aggregates parameters of users participating into the FL. After the aggregation, the central node sends the aggregated network parameters to each user. The above steps are repeated until the system performance becomes convergent. Different from the conventional distributed learning, FL is a multi-user cooperative learning without data share, and it allows users to interact and have nIID data [28], [29]. Compared with the centralized learning, the federated learning has been shown to achieve some advantages in the following three folds. Firstly, as a typical distributed learning, the federated learning allows mobile devices not to upload their full information to the CAPs, so as to protect the privacy of the mobile devices. Secondly, the federated learning can help reduce the large communication overhead in the process of centralized learning. Thirdly, unlike the centralized learning, the federated learning provides a flexible way to the system design, thanks to its characteristic of multi-module cooperation. Due to these advantages, a lot of researches have been put into the study of FL [30]–[32]. In particular, a new resource allocation framework was proposed for the wireless federated learning in [30], in order to optimize the convergence of the FL. To optimize the system performance constrained by a latency threshold, the authors in [31] proposed two bandwidth allocation strategies to maximize the number of users participating into FL. Therefore, FL provides a new inspiration to the computation offloading optimization in the MEC networks, which becomes the motivation of the work in this paper.

It is notable that there still exist some challenges in the federated learning, from the aspects of implementation and communication overhead. One challenge is the limited battery supply at the mobile devices. Once these mobile devices run out of their batteries, they will be dropped from the system operation, which may cause a severe performance degradation in the federated learning. Another challenge is the communication overhead, as mobile devices need to upload their local models for multiple rounds, in order to reach a high learning performance. However, this may cause a large communication overhead and result in a training bottleneck, particularly when some training networks consist of millions of parameters, such as convolutional neural networks.

B. Contribution

In this paper, we investigate distributed machine learning for a multiuser mobile edge computing system, where multiple users have some computational tasks of different priorities, which can be computed through offloading to multiple CAPs. We consider a cognitive eavesdropping environment where the users can use the wireless spectrum as long as the interference to the primary user is tolerated, and an eavesdropper in the network can overhear the confidential message from the users. For the considered system, we firstly present three optimization criteria, whereas *criterion I* aims to minimize the linear combination of latency and energy consumption, *criterion II* tries to minimize the latency with a given threshold of energy consumption, while *criterion III* is to minimize the energy consumption with a given latency threshold. We then solve the optimization problem in a distributed machine learning way by optimizing the offloading ratio as well as the allocation of bandwidth and computational capability, by taking into account the task priority among users. Specifically, a FL optimization framework is used, where each user employs DRL to solve the optimization. Simulation results are finally demonstrated to show that the proposed method can effectively reduce the system cost in terms of latency and energy consumption, and meanwhile ensure more bandwidth and computational capability allocated to the user with a higher task priority.

The main contributions of this work can be summarized as follows,

- This paper studies distributed machine learning for a multiuser multi-CAP MEC network in cognitive and secure environments, where the task priority of users is taken into account.
- We present three optimization criteria for the MEC networks, which provide a flexible choice for the system design and optimization, by taking into account the requirement of latency and energy consumption.

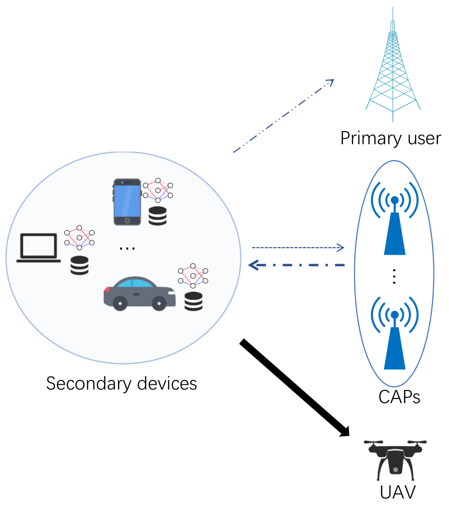


Fig. 1. System model of a multiuser and multi-CAP MEC network in cognitive eavesdropping environments.

- We solve the optimization problem through optimizing the offloading ratio as well as the allocation of bandwidth and computational capability in the FL framework, where each user employs DRL to solve the optimization.
- Simulation results are demonstrated to show the effectiveness of the proposed method in reducing the system latency and energy consumption. In particular, the user with a higher task priority is ensured to be allocated more bandwidth and computational capability.

C. Structure

The rest of this paper is organized as follows. After the introduction, Section II describes the system communication and computation models, and Section III provides three criteria for the system optimization in the MEC networks. After that, Section IV presents the FL based distributed algorithm to solve the offloading optimization problem in the MEC networks. Simulation results are demonstrated in Section V to verify the effectiveness of the proposed scheme, and finally we conclude this work in Section VI.

II. System Model

Fig. 1 depicts the system model of a distributed wireless mobile edge computing network in eavesdropping environments, where there are M secondary devices (SDs) within the coverage of K CAPs. The users work in an underlay spectrum-sharing mode, and they can use the spectrum of the primary user, as long as their interference to the primary user is below a given threshold I_P. We consider the eavesdropping environments, where there is one eavesdropper such as a unmanned aerial vehicle (UAV), which can overhear the confidential communication from the users[1] to the CAPs. The users have some computational tasks to be computed, which can be assisted by the CAPs with much more powerful computational capabilities.

[1]In this paper, the SDs and users are used interchangeably.

Let $D_m = (s_m, d_m)$ denote the characteristics of the task of the user SD$_m$, where s_m is the task length, and d_m is the task priority. In practice, the task characteristics may be dynamically varying due to many factors, which can be characterized by the uniform distribution, without loss of generality. Specifically, s_m is subject to the uniform distribution of $\mathcal{U}(s_{min}, s_{max})$[2], where s_{min} and s_{max} are the minimum and maximum lengths, respectively.

As the users have limited computational capabilities, the tasks can not be computed at local completely, given the requirement from the perspectives of latency and energy consumption. Hence, the tasks of users should be partially or fully offloaded to the CAPs. Specifically, let $\alpha_{m,k} \in [0, 1]$ denote the offloading ratio from the user SD$_m$ to CAP$_k$ through wireless transmission. Accordingly, $\alpha_{m,0}$ part of the task will be computed at local, while the rest part will be computed by the CAPs. As the users have different task characteristics, such as task length or priority, the communication and computational resources should be allocated dynamically, in order to enhance the system performance by reducing the latency and energy consumption.

In the following, we will firstly describe the task priority, and then present the offloading and computational model for the considered secure MEC network.

A. Task Priority

In practice, different tasks may have different priorities in the latency and energy consumption. For example, the signal control tasks such as car navigation and autopilot, have a higher priority in the latency. For the IoT nodes with limited energy, the energy resources are quite limited, and computational tasks have a high priority in the energy consumption. Overall, the task priority plays an important role in the task implementation and system performance.

To support the tasks with a higher priority, the system communication and computational resources should be allocated more to the tasks, in order to meet the task requirement. In particular, we can allocate the transmission bandwidth and computational capability, by taking into account the task priority. As to the user SD$_m$ with priority d_m, we need to allocate the bandwidth from the k-th edge node CAP$_k$ for the user SD$_m$. A feasible solution of the bandwidth allocation is,

$$\widetilde{\beta}_{m,k} = \frac{\beta_{m,k} d_m}{\sum_{m_1=1}^{M} \beta_{m_1,k} d_{m_1}}, \tag{1}$$

where $\beta_{m,k}$ denotes the expected bandwidth allocation ratio from CAP$_k$ to the user SD$_m$, while $\widetilde{\beta}_{m,k}$ is a weighted version of $\beta_{m,k}$ by taking into account the task priority. As the training of users operates in a distributed way, each user can adjust its own bandwidth only, which may cause the sum of bandwidth allocated to all users be larger than the system total bandwidth.

[2]In practice, the task size s_m may vary due to many factors, such as dynamic application requirement and processes in the user terminal. Without loss of generality, we assume that s_m follows the uniform distribution within the interval of $[s_{min}, s_{max}]$. Moreover, when the task size follows some other distributions, such as normal distribution or exponential distribution, the results in this work will be changed accordingly, whereas the optimization methods in this work can be easily extended.

To solve this issue, we use (1) to determine the allocation ratio of the bandwidth for each user. Specifically, for the right-hand side (RHS) of (1), the numerator represents the bandwidth share which the user SD$_m$ with task priority d_m expects to be allocated from CAP k, while the denominator is the total bandwidth share which M SDs expect to be allocated from CAP k. This normalized processing is exploited to ensure the total allocation ratio of the bandwidth equal to unity.

Similar to the process of allocating the bandwidth ratio, we can allocate the computational capability to the users by taking into account the task priority as,

$$\widetilde{\gamma}_{m,k} = \frac{\gamma_{m,k} d_m}{\sum_{m_1=1}^{M} \gamma_{m_1,k} d_{m_1}}, \tag{2}$$

where $\gamma_{m,k}$ denotes the expected allocation ratio of the computational capability from CAP$_k$ to the user SD$_m$, while $\widetilde{\gamma}_{m,k}$ is a weighted version of $\gamma_{m,k}$ by taking into account the task priority. In a word, by allocating more resources such as the bandwidth and computational capability, the tasks with a higher priority can be implemented with a lower latency and energy consumption, which can help enhance the system performance of the considered secure MEC network.

B. Offloading Model

When the local users cannot calculate the computational tasks completely by themselves, some or full parts of the task should be offloaded to the CAPs through wireless channels from the users to the CAPs. In this paper, the secondary users work in the cognitive environments, and the transmit power of the secondary users are limited by the peak interference from the primary user. Let I_P denote the peak interference power to the primary user imposed by the secondary users, and the transmit power of the user SD$_m$ is given by,

$$P_m = \frac{I_P}{|g_m|^2}, \tag{3}$$

where $g_m \sim \mathcal{CN}(0, \zeta_p)$ denotes the channel parameter of the link from the user SD$_m$ to the primary user. From P_m, we can write the data rate of the link from SD$_m$ to CAP$_k$ as,

$$R_{m,k} = W_k^{total} \widetilde{\beta}_{m,k} \log_2\left(1 + \frac{P_m |h_{m,k}|^2}{\sigma_k^2}\right), \tag{4}$$

$$= W_k^{total} \widetilde{\beta}_{m,k} \log_2\left(1 + \frac{I_P |h_{m,k}|^2}{\sigma_k^2 |g_m|^2}\right), \tag{5}$$

where W_k^{total} is the total wireless bandwidth of CAP$_k$, $h_{m,k} \sim \mathcal{CN}(0, \zeta_k)$ denotes the channel parameter of the link from the user SD$_m$ to CAP$_k$, and σ_k^2 is the variance of the additive white Gaussian noise (AWGN) at CAP$_k$. When the UAV attacker overhears the confidential communication from the users to the CAPs, the eavesdropping data rate of the link from the user SD$_m$ to the UAV is,

$$R_{m,\mathrm{e}} = W_k^{total} \widetilde{\beta}_{m,k} \log_2\left(1 + \frac{P_m |h_{m,UAV}|^2}{\sigma_{UAV}^2}\right), \tag{6}$$

$$= W_k^{total} \widetilde{\beta}_{m,k} \log_2\left(1 + \frac{I_P |h_{m,UAV}|^2}{\sigma_{UAV}^2 |g_m|^2}\right), \tag{7}$$

where $h_{m,UAV} \sim \mathcal{CN}(0, \zeta_{UAV})$ denotes the channel parameter of the link from SD$_m$ to the UAV, and σ_{UAV}^2 is the variance of the AWGN at the UAV. From (5) and (7), the secrecy data rate under eavesdropping can be obtained as[3],

$$R_{m,k}^s = [R_{m,k} - R_{m,e}]^+, \tag{8}$$

where $[X]^+$ returns X if X is positive, or zero otherwise. From (8), we can write the transmission latency of the offloading as,

$$l_{m,k}^{tran} = \frac{s_m \alpha_{m,k}}{R_{m,k}^s}. \tag{9}$$

From (3) and (9), we can write the transmission energy consumption of the offloading as,

$$e_{m,k}^{tran} = l_{m,k}^{tran} P_m,$$

$$= l_{m,k}^{tran} \frac{I_P}{|g_m|^2}. \tag{10}$$

C. Computation Model

Note that the computational task can be executed either at the local device or at the edge nodes. When the $\alpha_{m,0}$ part of the task D_m is executed locally, the local latency is expressed as,

$$l_m^{local} = \frac{s_m \alpha_{m,0}}{f_m}, \tag{11}$$

where f_m denotes the computational capability of the user SD$_m$. Accordingly, the local energy consumption is,

$$e_m^{local} = P_m^{comp} l_m^{local}, \tag{12}$$

where P_m^{comp} is the computational power of the user SD$_m$.

When the $\alpha_{m,k}$ part of the task D_m is executed by CAP$_k$, the computational latency is,

$$l_{m,k}^{comp} = \frac{s_m \alpha_{m,k}}{f_k^{total} \widetilde{\gamma}_{m,k}}, \tag{13}$$

where f_k^{total} denotes the total computational capability of CAP$_k$. Moreover, the energy consumption at the CAP$_k$ is,

$$e_{m,k} = P_k^{comp} l_{m,k}^{comp}, \tag{14}$$

where P_k^{comp} is the computational power at CAP$_k$.

Since the task can be computed by the CAPs after the offloading, the offloading and computational latency from the user SD$_m$ to CAP$_k$ is,

$$l_{m,k} = l_{m,k}^{tran} + l_{m,k}^{comp}. \tag{15}$$

As the processing of offloading and local computation can be carried in parallel, the latency of computing the task of the user SD$_m$ is,

$$l_m^{total} = \max\{l_m^{local}, \max\{l_{m,1}, 1_{m,2}, \ldots, l_{m,K}\}\}. \tag{16}$$

Accordingly, the total latency of the whole system is,

$$L_{total} = \max\{l_1^{total}, l_2^{total}, \ldots, l_M^{total}\}. \tag{17}$$

[3]Note that if there is no eavesdropper to overhear the confidential message from the users, the data transmission rate from the user SD$_m$ to the CAP$_k$ will become into $R_{m,k}$. As $R_{m,k}^s \leq R_{m,k}$ holds, we can find that the existence of eavesdroppers will be harmful to the data transmission rate.

As shown in the above equation, the system total latency is the maximal latency among M users, as the users perform the offloading and computing in parallel. In particular, the computing is performed among M users in parallel, as each user uses its individual computational resource, either local computational capability or the allocated computational capability from the CAPs, to calculate the tasks. Moreover, the offloading is implemented in parallel among M users, as each user employs an orthogonal spectrum frequency for the wireless transmission.

In further, the energy consumption at the user SD_m is given by,

$$e_m^{total} = e_m^{local} + \sum_{k=1}^{K} \left(e_{m,k} + e_{m,k}^{tran} \right). \tag{18}$$

Hence, the total energy consumption of the whole system is,

$$E_{total} = \sum_{m=1}^{M} e_m^{total}. \tag{19}$$

III. Problem Formulation

As different offloading and bandwidth allocation strategies yield different latency and energy consumption for the considered secure MEC network, we need to optimize the system performance by allocating the system resources. To this end, we need to firstly formulate the optimization problem. In this paper, we provide three optimization criteria for the considered secure MEC network. Specifically, criterion I is to minimize a linear combination of latency and energy consumption, through adjusting the offloading ratio, bandwidth allocation, and computational capability allocation, given by,

$$\min_{\{\alpha_{m,k}, \widetilde{\beta}_{m,k}, \widetilde{\gamma}_{m,k}\}} \quad \Phi_{\mathrm{I}} = \lambda L_{total} + (1-\lambda) E_{total}, \tag{20a}$$

$$\text{s.t.} \quad \mathrm{C}_1: \alpha_{m,0} + \sum_{k=1}^{K} \alpha_{m,k} = 1, \forall m \in \mathcal{M}, \tag{20b}$$

$$\mathrm{C}_2: \sum_{m=1}^{M} \widetilde{\beta}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{20c}$$

$$\mathrm{C}_3: \sum_{m=1}^{M} \widetilde{\gamma}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{20d}$$

where $\lambda \in [0,1]$ is a weight factor, and constraint C_1 indicates that the computational task is fully computed by the local computing and the CAPs through offloading. Constraint C_2 ensures that the bandwidth allocated to the uses should not exceed the total bandwidth of the CAPs, and constraint C_3 guarantees that the computational capability allocated to the users should not exceed the total computational capability of the CAPs. Note that the linear combination form in criterion I is feasible to measure the system cost in some application scenarios, where the system needs to adjust the importance of the latency and energy consumption dynamically, such as smart home, UAV networks and electric autopilot, from the following two aspects. Firstly, the latency and energy consumption can be jointly optimized by minimizing the linear combination. In particular, when λ is equal to 0, minimizing the linear combination is equivalent to minimizing the energy consumption, while when λ is equal to 1, minimizing the linear combination is equivalent to minimizing the latency. Secondly, the linear combination can provide a flexible optimization form of the latency and energy consumption for the MEC networks. We can increase the value of λ, when the latency becomes dominant in some scenarios, while we can decrease the value of λ, when the energy consumption becomes dominant. Due to these two reasons, most of existing works, such as [33], use the linear combination of the latency and energy consumption to measure the system cost of the MEC networks.

In addition to criterion I, we provide criterion II in the following to minimize the system latency with a given constraint on the energy consumption, given by,

$$\min_{\{\alpha_{m,k}, \widetilde{\beta}_{m,k}, \widetilde{\gamma}_{m,k}\}} \quad \Phi_{\mathrm{II}} = L_{total}, \tag{21a}$$

$$\text{s.t.} \quad \mathrm{C}_1: E_{total} \le E_{th}, \tag{21b}$$

$$\mathrm{C}_2: \alpha_{m,0} + \sum_{k=1}^{K} \alpha_{m,k} = 1, \forall m \in \mathcal{M}, \tag{21c}$$

$$\mathrm{C}_3: \sum_{m=1}^{M} \widetilde{\beta}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{21d}$$

$$\mathrm{C}_4: \sum_{m=1}^{M} \widetilde{\gamma}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{21e}$$

where E_{th} is the energy consumption threshold predetermined by the system, and constraint C_1 ensures that the system energy consumption should not exceed the threshold. Criterion II can be used in the latency-sensitive scenarios, such as the automatic driving in Internet of Vehicles and control systems.

In further, we provide criterion III in the following to minimize the system energy consumption with a given constraint on the latency, given by,

$$\min_{\{\alpha_{m,k}, \widetilde{\beta}_{m,k}, \widetilde{\gamma}_{m,k}\}} \quad \Phi_{\mathrm{III}} = E_{total}, \tag{22a}$$

$$\text{s.t.} \quad \mathrm{C}_1: L_{total} \le L_{th}, \tag{22b}$$

$$\mathrm{C}_2: \alpha_{m,0} + \sum_{k=1}^{K} \alpha_{m,k} = 1, \forall m \in \mathcal{M}, \tag{22c}$$

$$\mathrm{C}_3: \sum_{m=1}^{M} \widetilde{\beta}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{22d}$$

$$\mathrm{C}_4: \sum_{m=1}^{M} \widetilde{\gamma}_{m,k} = 1, \forall k \in \mathcal{K}, \tag{22e}$$

where L_{th} is the latency threshold predetermined by the system, and constraint C_1 ensures that the system latency should not exceed the threshold. Criterion III can be used in the energy-sensitive scenarios, such as the IoT networks where the nodes have limited energy.

Note that the above three criteria can be applicable to different application scenarios with different requirements on the system performance. Specifically, for the application scenarios such as smart home, electric autopilot and UAV networks, criterion I tends to be used, since both the metrics of latency and energy consumption are important. On the other hand, for the application scenarios such as video transmission and navigation, the latency plays a much more important role in the system performance, and we should use criterion II to design the MEC system. In further, for the application scenarios such as energy-aware IoT networks, criterion III should be used for the system design. Overall, this paper provides a flexible choice for the system design and optimization of the MEC networks.

IV. FL Based Distributed Optimization

In this part, we propose a FL based distributed optimization scheme, which aims to minimize the system cost by adjusting the offloading ratio, bandwidth allocation ratio and computational capability ratio. Specifically, we will firstly formulate the task offloading and resource allocation into a Markov decision process (MDP) problem, and then design the state and action spaces in the DQN. In further, we employ the FL framework to solve the optimization problems in (20)-(22) in a distributed way, which can help reduce the communication overhead and protect the data privacy. The details of the FL based distributed optimization is given as follows.

A. Markov Decision Process

Since the characteristics of computational tasks and the wireless channels from the users to the CAPs are time-varying, we use the MDP to characterize the tasks offloading and resource allocation problem. In particular, the MDP can be characterized by a three-tuple $\{S, A, R\}$, where S is the state space, A is the action space, and R is the reward function. The MDP model is elaborated in detail as follows.

1) State Space: Let $S_m(t) = \{\mathbf{C}(t), \mathbf{B}(t), \mathbf{F}(t)\}$ denote the state space of the user SD$_m$ at time slot t ($t = 1, 2, \ldots, t_{max}$), where t_{max} is the maximum time slot. The state space $S_m(t)$ is expanded as,

$$S_m(t) = \begin{cases} \mathbf{C}(t) = \{\alpha_{m,0}(t), \alpha_{m,1}(t), \ldots, \alpha_{m,K}(t)\}, \\ \mathbf{B}(t) = \{\beta_{m,1}(t), \beta_{m,2}(t), \ldots, \beta_{m,K}(t)\}, \\ \mathbf{F}(t) = \{\gamma_{m,1}(t), \gamma_{m,2}(t), \ldots, \gamma_{m,K}(t)\}, \end{cases} \tag{23}$$

where $\alpha_{m,0}(t)$ denotes the local computational ratio of the user SD$_m$ at time slot t, $\alpha_{m,k}(t)$ is the task offloading ratio from SD$_m$ to CAP$_k$ at time slot t, and $\beta_{m,k}(t)$ and $\gamma_{m,k}(t)$ represent the expected bandwidth and computational capability allocation ratio from CAP$_k$ to SD$_m$, respectively.

2) Action Space: A discrete vector $A_m(t)$ is used to denote the action space of the user SD$_m$ at time slot t, and we use the index of the action space $A_m(t)$ to denote the action command. In general, the state space will change after performing an action, and the action can update only one element of the state space once. The action space $A_m(t)$ is expressed by,

$$A_m(t) = \{a_m^j(t) | 0 \leq j \leq 6K+1\}, \tag{24}$$

$$a_m^j(t) = \begin{cases} \xi, & \text{If } j\%2 = 0, \\ -\xi, & \text{If } j\%2 = 1, \end{cases} \tag{25}$$

where j is the index of $A_m(t)$ and ξ denotes the step size. According to the range of j, we can update the offloading ratio, expected bandwidth and computational capability allocation ratio. Specifically, when $0 \leq j \leq 2K+1$ holds, the task offloading ratio is updated, which can be expressed as,

$$\alpha_{m,v}(t) = \alpha_{m,v}(t-1) + a_m^j(t-1), \tag{26}$$

where $v = [0, 1, 2, \ldots, K]$ indicates the local user or the CAPs. When $2K+2 \leq j \leq 4K+1$ holds, the expected bandwidth allocation ratio is updated as,

$$\beta_{m,k}(t) = \beta_{m,k}(t-1) + a_m^j(t-1). \tag{27}$$

When $4K+2 \leq j \leq 6K+1$ holds, the expected computational capability allocation ratio is updated as,

$$\gamma_{m,k}(t) = \gamma_{m,k}(t-1) + a_m^j(t-1). \tag{28}$$

After defining the action command, we then describe the way these actions execute. In general, we expect that the agent can not only exploit the prior experience, but also explore some unknown state actions. Therefore, we use the ϵ-greedy algorithm to explore the action $A_m(t)$. In particular, when the probability is ϵ, the action is selected randomly; while when the probability is $1-\epsilon$, the action with the maximum Q function is selected. The action selection is modeled by,

$$A_m(t) = \begin{cases} random, & \epsilon, \\ \arg\max\limits_{A_m} Q_m(S_m(t), A_m(t); \omega_m), & 1-\epsilon, \end{cases} \tag{29}$$

where $0 \leq \epsilon \leq 1$. After selecting the action $A_m(t)$, the state $S_m(t)$ is changed accordingly.

3) Reward Function: After the state space is updated, we compare the cost of the user SD$_m$ at the current time slot with that at the previous time slot, in order to obtain a reward function for the agent SD$_m$. In particular, when the cost of the user SD$_m$ at time slot $t-1$ is equal to that at the current time slot t, the agent SD$_m$ gets a zero feedback. When the cost of SD$_m$ at the current moment is lower than that at the previous time slot, the agent SD$_m$ gets a positive feedback. Otherwise, the agent SD$_m$ gets a negative feedback.

When the system chooses criterion I as the optimization objective, the reward function of the user SD$_m$ is given by,

$$R_m^{\mathrm{I}}(t) = \begin{cases} 0, & \text{If } \Phi_{\mathrm{I}}(t) = \Phi_{\mathrm{I}}(t-1), \\ 1, & \text{If } \Phi_{\mathrm{I}}(t) < \Phi_{\mathrm{I}}(t-1), \\ -1, & \text{If } \Phi_{\mathrm{I}}(t) > \Phi_{\mathrm{I}}(t-1), \end{cases} \tag{30}$$

where $\Phi_{\mathrm{I}}(t)$ is the cost of the user SD$_m$ under criterion I at time slot t, while $\Phi_{\mathrm{I}}(t-1)$ is the cost at time slot $t-1$.

Similarly, when the system chooses criterion II as the optimization objective, the reward function of the user SD$_m$ is,

$$R_m^{\mathrm{II}}(t) = \begin{cases} 0, & \text{If } \Phi_{\mathrm{II}}(t) = \Phi_{\mathrm{II}}(t-1), \\ 1, & \text{If } \Phi_{\mathrm{II}}(t) < \Phi_{\mathrm{II}}(t-1), \\ -1, & \text{If } \Phi_{\mathrm{II}}(t) > \Phi_{\mathrm{II}}(t-1), \end{cases} \tag{31}$$

where $\Phi_{\mathrm{II}}(t)$ is the cost of the user SD$_m$ under criterion II at time slot t, while $\Phi_{\mathrm{II}}(t-1)$ is the cost at time slot $t-1$.

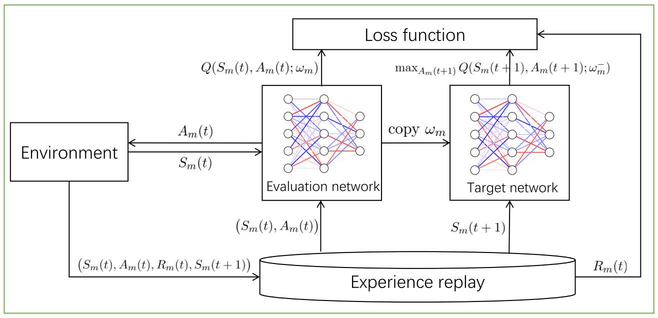


Fig. 2. Structure of DQN of the agent SD$_m$.

When the system chooses criterion III as the optimization objective, the reward function of the user SD$_m$ is expressed as,

$$R_m^{\text{III}}(t) = \begin{cases} 0, & \text{If } \Phi_{\text{III}}(t) = \Phi_{\text{III}}(t-1), \\ 1, & \text{If } \Phi_{\text{III}}(t) < \Phi_{\text{III}}(t-1), \\ -1, & \text{If } \Phi_{\text{III}}(t) > \Phi_{\text{III}}(t-1), \end{cases} \tag{32}$$

where $\Phi_{\text{III}}(t)$ is the cost of the user SD$_m$ under criterion III at time slot t, while $\Phi_{\text{III}}(t-1)$ is the cost at time slot $t-1$.

B. Local Training

Since the state space and the action space in a dynamic system are enormous and complicated, most of reinforcement algorithms such as Q-learning cannot work well. Therefore, we turn to employ DQN-based algorithm to train the local models. Different from Q-learning, DQN uses a DNN to approximate the Q function and leverages an experience relay to break the correlation between the data. Fig. 2 depicts the structure of the DQN used in this paper, where the user SD$_m$ puts the current state and observation from experience pool into evaluation network, and then obtains the action-state value $Q_m(S_m(t), A_m(t); \omega_m)$, in which ω_m denotes the weighted parameter in the evaluation network. Specifically, the loss function is the difference between the action-state value in the evaluation network and the next state value in the target network, which is given by,

$$\hat{L}_m(t) = (Y_m(t) - Q_m(S_m(t), A_m(t); \omega_m))^2, \tag{33}$$

where $Y_m(t)$ is the target value function, represented as,

$$Y_m(t) = R_m(t) + \delta \arg\max_{A_m(t+1)} Q_m(S_m(t+1), A_m(t+1); \omega_m^-), \tag{34}$$

in which δ is a discount factor and ω_m^- denotes the weighted parameter in the target network. Note that the target network has the same structure and initial weights as the evaluation network. However, the evaluation network is updated at each iteration, while the target network is updated every c steps. In particular, the update process of the evaluation network of SD$_m$ is,

$$\omega_m(t+1) = \omega_m(t) - \hat{\eta} \frac{\partial \hat{L}_m(t)}{\partial \omega_m(t)}, \tag{35}$$

where $\hat{\eta}$ is the learning rate of the users.

Algorithm 1: Proposed FL Based Distributed Optimization Scheme.

```
    Input: $D_m$, $W_k^{total}$, $f_k^{total}$
    Output: $\alpha_{m,k}$, $\alpha_{m,0}$, $\widetilde{\beta}_{m,k}$, $\widetilde{\gamma}_{m,k}$
 1: Initialize experience relay pool G
 2: Initialize weighted parameters $\omega_m$ and $\omega_m^-$
 3: for Epoch=1 : $R$ do
 4:   for Episode=1 : $Z$ do
 5:     for m=1 : $M$ do
 6:       Initialize state $S_m$ and environment of agent $m$
 7:     end for
 8:     for t=1 : $t_{max}$ do
 9:       for m=1 : $M$ do
10:         Choose an action $A_m(t)$ according to (29)
11:         Execute the action $A_m(t)$ and get reward
            $R_m(t)$, then observe the next state $S_m(t+1)$
12:         Store transition
            $(S_m(t), A_m(t), R_m(t), S_m(t+1))$ in
            experience relay pool G
13:         Choose random mini-batch of transitions
            $(S_m(t), A_m(t), R_m(t), S_m(t+1))$ from G
14:         Calculate loss function $\hat{L}_m(t)$ according to
            (33)
15:         Perform a gradient descent step with respect to
            $\omega_m$
16:         Update $Q(t+1) = Q(t)$ every $c$ steps
17:         Update the weighted parameter of the user
            SD$_m$ according to (35)
18:       end for
19:       if t%$\varrho$==0 then
20:         Aggregate the weighted parameters of all users
            at the central node according to (36)
21:         The central node distributes the global
            parameter to all users according to (37)
22:       end if
23:     end for
24:   end for
25: end for
```

C. Model Aggregation and Model Distribution

After elaborating the process that each user finds its own allocation strategy through DQN, we further describe how the users cooperate in the FL framework to optimize the whole system performance. Fig. 3 shows the process of the FL framework, which consists of several key steps: local training, model upload, model aggregation and model distribution. Specifically, the users firstly initialize and train their own DQN models locally for ϱ rounds, and then upload their own model parameters to a central node which is responsible for aggregation. According to the federated average algorithm, the model aggregation at time slot $i\varrho$ is expressed as,

$$\omega(i\varrho + 1) = \frac{1}{M} \sum_{m=1}^{M} \omega_m(i\varrho), \tag{36}$$

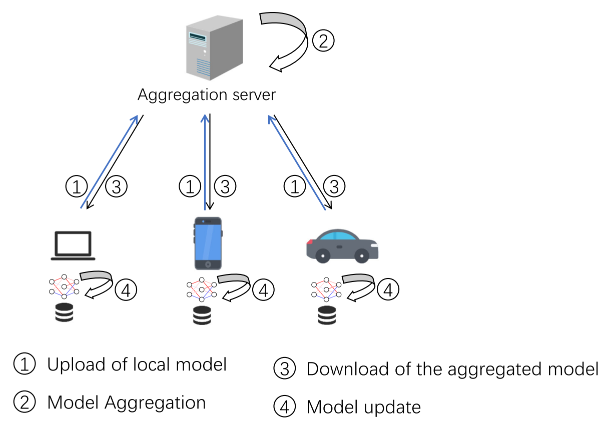


Fig. 3. The process of the federated learning.

where i $(i = 1, 2, \ldots, i_{max})$ represents the number of the aggregation, and $\omega(i\varrho + 1)$ denotes the global model parameter of the evaluation network at time slot $i\varrho + 1$. Afterwards, the central node distributes the global model parameter to all users. The model distribution from the central node to the users can be formulated as,

$$\omega_m(t\varrho + 1) = \omega(t\varrho + 1). \tag{37}$$

After the users participating into the FL have updated their local model parameters of the evaluation network, one iteration of the FL process is completed. This iterative process repeats until the system performance becomes convergent. The whole procedure of the FL based distribution resource allocation is summarized in Algorithm 1.

V. Simulation Results

In this section, we perform some simulations to evaluate the performance of the proposed schemes. The channels in the network follow Rayleigh flat fading [34], [35], and the average channel gains of the links from the users to the UAV, primary user and CAPs are set to 0.01, 1 and 1, respectively. The variance of the AWGN at the UAV and CAPs is set to 0.1. Moreover, the power of the peak interference to the primary user is set to 2 W, while the maximum transmit power of users is limited to 5 W. The computational power of the local users and CAPs are 1 W. The step size ξ for adjusting the offload ratio, bandwidth allocation ratio, and computing capability allocation ratio is set to 0.1. Without loss of generality, the wireless bandwidth of CAP k is set to $(50 + k)$ MHz, and the computational capability of CAP k is set to $(8 + 0.1\ k)$ 10^6cycles/s, where k varies in $\{1, \ldots, K\}$. The local computational capability of the user SD$_m$ is $(5 + 0.1\ m)$ 10^4cycles/s, where m varies in $\{1, \ldots, M\}$. In further, the task size of the user SD$_m$ is uniformly distributed in the range of $[10 + 5\ m, 10 + 5(m + 1)]$ MB, and we assume that the priority of the last user is 3, while that of the other users is set to 1. Considering the requirements from the application scenarios in criterion II and criterion III, we set the latency threshold L_{th} and the energy consumption E_{th} to 10 and 20, respectively, when $M = 5$ and $K = 2$.

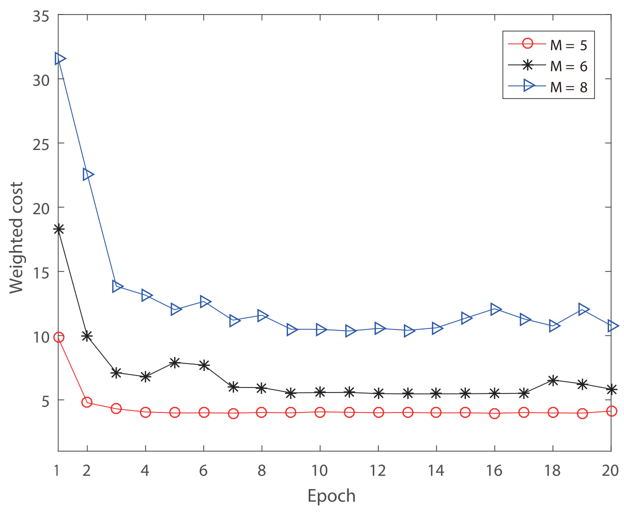


Fig. 4. Convergence of the proposed method with $K = 2$ and $\lambda = 0.5$: Criterion I.

Fig. 4 depicts the convergence of the proposed method under criterion I versus the epoch, where the number of users varies in $\{5, 6, 8\}$, the number of CAPs is 2, λ is 0.5, and the number of epochs varies from 1 to 20. As observed from Fig. 4, we can find that for various numbers of users, the weighted cost declines along with the increasing number of epochs. In particular, the cost with 5 users declines roughly from 10 to 4 and it converges at epoch 2, the cost with 6 users declines roughly from 18 to 6 and it converges at epoch 7, and the cost with 8 users declines roughly from 32 to 11 and it converges at epoch 9. Moreover, the weighted cost with $M = 6$ fluctuates at epoch 5 and 6, as the system may perform a bad exploration in some poor channels. In further, we can see that the weighted cost increases along with a larger M, since more users increase the burden on the communication and computation and result in a larger cost. Therefore, Fig. 4 verifies that the proposed method can help reduce the system cost effectively for various numbers of users under criterion I.

Fig. 5 depicts the convergence of the proposed method under criterion II versus the epoch, where the number of users varies in $\{5, 6, 8\}$, the number of CAPs is 2, and the number of epochs varies from 1 to 20. As observed from Fig. 5, we can find that for various numbers of users, the latency declines along with the increasing number of epochs. In particular, the latency with 5 users declines roughly from 5.5 to 1.3 and it converges at epoch 6, the latency with 6 users declines roughly from 5.5 to 1.7 and it converges at epoch 6, and the latency with 8 users declines roughly from 7.3 to 2.5 and it converges at epoch 17. Moreover, the latency with $M = 8$ fluctuates at epoch 10 and 15, as the system may perform a bad exploration in some poor channels. In further, we can see that the latency increases along with a larger M, since more users increase the burden on the communication and computation and result in a larger latency. Therefore, Fig. 5 verifies that the proposed method can help reduce the system latency effectively for various numbers of users under criterion II.

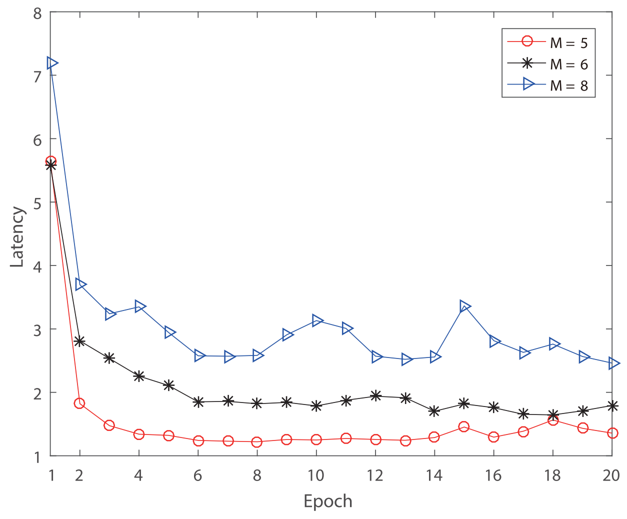


Fig. 5. Convergence of the proposed method under with $K = 2$: Criterion II.

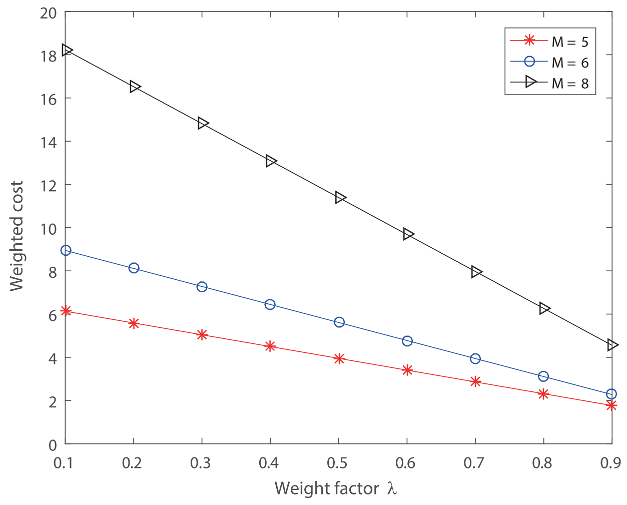


Fig. 7. Weighted cost of criterion I versus λ with $K = 2$.

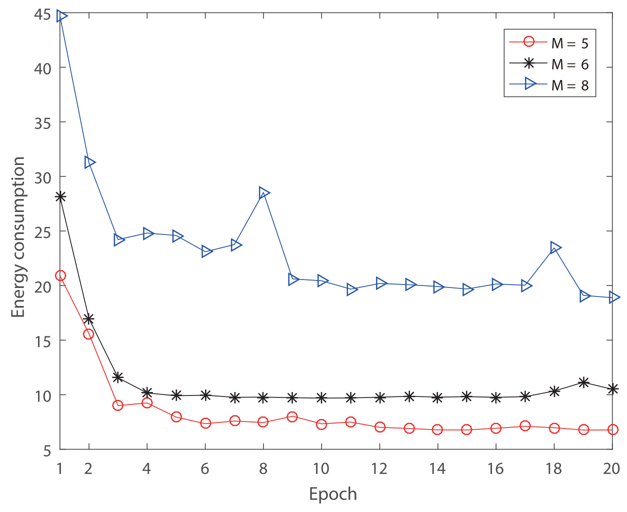
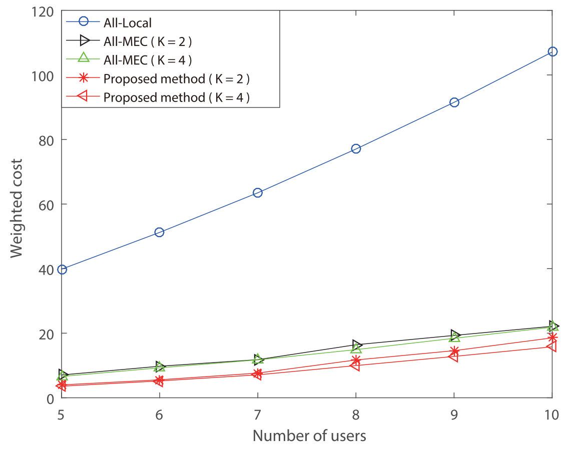


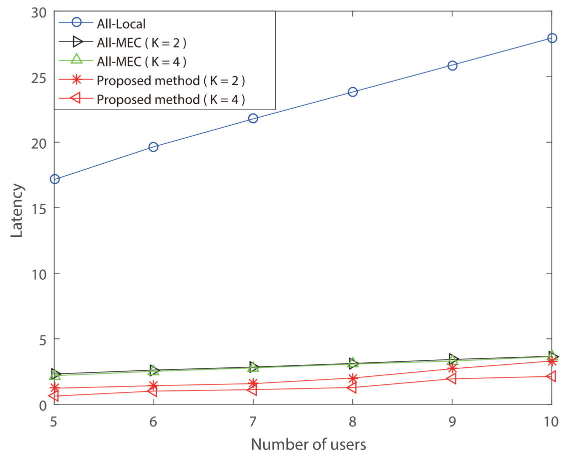
Fig. 6. Convergence of the proposed method with $K = 2$: Criterion III.

Fig. 6 depicts the convergence of the proposed method under criterion III versus the epoch, where the number of users varies in $\{5, 6, 8\}$, the number of CAPs is 2, and the number of epochs varies from 1 to 20. As observed from Fig. 6, we can find that for various numbers of users, the energy consumption declines along with the increasing number of epochs. In particular, the energy consumption with 5 users declines roughly from 21 to 7 and it converges at epoch 6, the energy consumption with 6 users declines roughly from 28 to 11 and it converges at epoch 5, and the energy consumption with 8 users declines roughly from 45 to 18 and it converges at epoch 9. Moreover, the energy consumption with $M = 8$ fluctuates at epoch 8 and 18, as the system may perform a bad explorations in some poor channels. In further, we can see that the energy consumption increases along with a larger M, since more users will increase the burden on the communication and computation and result in a larger energy consumption. Therefore, Fig. 6 verifies that the proposed method can help reduce the system energy consumption effectively for various numbers of users under criterion III. Fig. 7 demonstrates the linearly weighted cost of criterion I versus the weight factor λ, where λ varies from 0.1 to 0.9, the number of users varies in $\{5, 6, 8\}$ and the number of CAPs is 2. As observed from this figure, we can find that the weighted cost decreases along with a larger λ, as the energy consumption dominates in the weighted cost of the considered system. In particular, the weighted cost with $M = 5$ declines from 6 to 2, the weighted cost with $M = 6$ declines from 9 to 3, and the weighted cost with $M = 8$ declines from 18 to 5. Moreover, we can see that the weighted cost increases with a larger M, since more users will increase the system overhead in the communication and computation. Hence, the results in Fig. 7 further verify that the proposed method can provide an effective strategy for various values of λ under criterion I. Fig. 8 shows the performance comparison of the three criteria in terms of system cost, where the number of users varies from 5 to 10, the number of CAPs is 2 or 4, and $\lambda = 0.5$. Specifically, Fig. 8(a), 8(b) and 8(c) correspond to criterion I, II and III, respectively. For comparison, we also plot the system cost of "All-Local" which computes all tasks at local, and "All-MEC" which computes all tasks at the CAPs. As observed from these three figures, we can find that the system cost of the three criteria increases along with a larger M, as more users will cause more burden on the communication and computation, which leads to a higher system cost. Moreover, the system cost in these figures decreases with a larger K, as more CAPs can help reduce the burden on the communication and computation, which leads to a lower system cost. In further, the proposed method outperforms the conventional "All-Local" and "All-MEC" methods under the three criteria. For example, when $K = 2$, the system cost of "All-Local" in Fig. 8(a) roughly increases from 40 to 109, and that of "All-MEC" roughly increases from 9 to 21. In contrast, the system cost of the proposed method slightly increases from 4 to 19. Therefore, the results in these figures verify the effectiveness of the proposed method for the considered system furthermore.

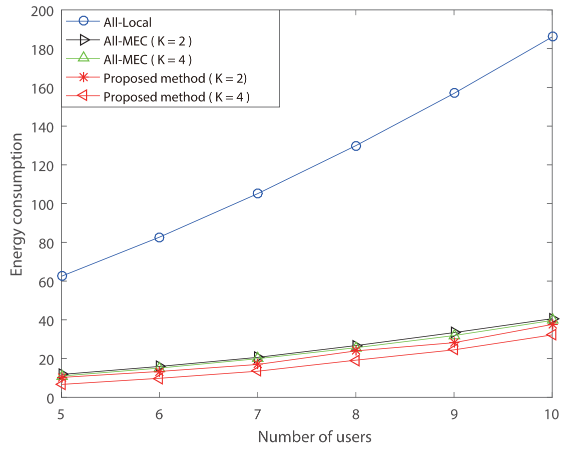
To show the effect of task priority on the bandwidth allocation under the three criteria, Fig. 9(a) and 9(b) are presented to show the allocated bandwidth of each user from the CAPs, where



(a) Criterion I

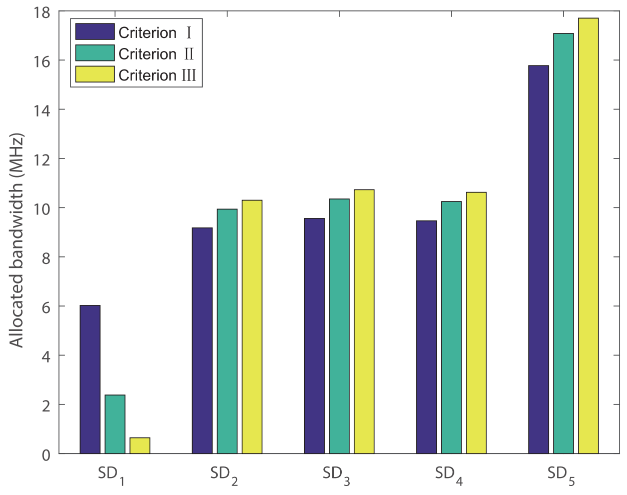


(b) Criterion II

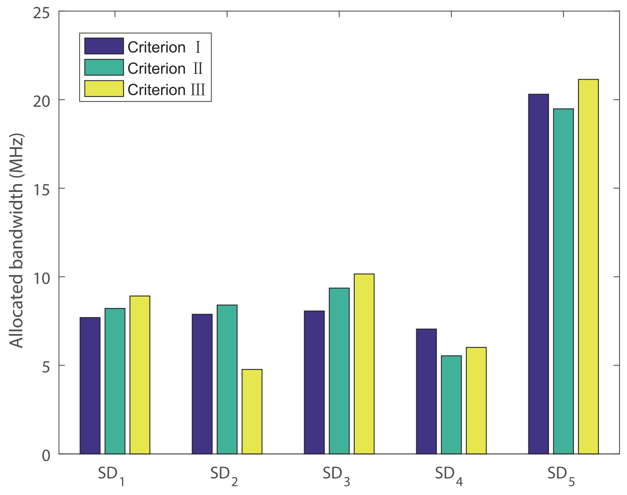


(c) Criterion III

Fig. 8. Performance comparison of the three criteria versus the number of users.



(a) CAP$_1$

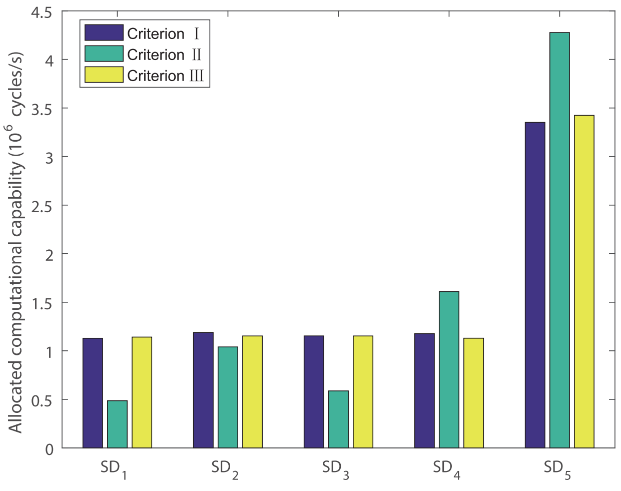


(b) CAP$_2$

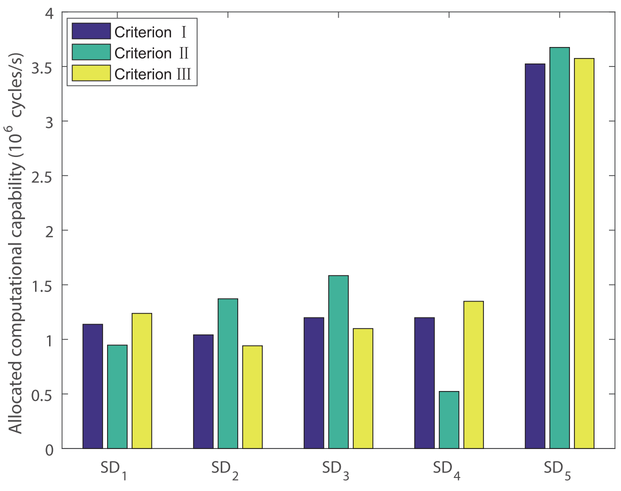
Fig. 9. Allocated bandwidth from CAPs to the users under the three criteria with $M = 5$ and $K = 2$.

$M = 5$, $K = 2$, $\lambda = 0.5$, and the task priority of the five users is $\{1, 1, 1, 1, 3\}$. As observed from Fig. 9(a), we can find that for the three criteria, the bandwidth allocated from CAP$_1$ to the user SD$_5$ roughly reaches 15.7 MHz, 17.1 MHz and 17.7 MHz under the three criteria, respectively, while that from CAP$_1$ to other users does not exceed 10.7 MHz. Moreover, the bandwidth allocated to SD$_1$ is lower than SD$_{2-4}$, due to the wireless link from SD$_1$ to CAP$_1$ is better than those from the other users to CAP$_1$. Similarly, as observed from Fig. 9(b), we can see that for the three criteria, the bandwidth allocated from CAP$_2$ to the user SD$_5$ roughly reaches 20.3 MHz, 19.5 MHz and 21.1 MHz, respectively, while that from CAP$_2$ to other users does not exceed 10.2 MHz. Therefore, Fig. 9 validates the effectiveness of the proposed priority based bandwidth allocation policy.

To show the impact of task priority on the computational capability allocation under the three criteria, Fig. 10(a) and 10(b) are demonstrated to show the allocated computational capability of each user from the CAPs, where $M = 5$, $K = 2$, $\lambda = 0.5$, and the task priority of the five users is $\{1, 1, 1, 1, 3\}$. As observed from Fig. 10(a), we can find that for the three criteria, the computational capability allocated from CAP$_1$ to the user SD$_5$ roughly reaches 3.4×10^6cycles/s, 4.3×10^6cycles/s and 3.5×10^6cycles/s, respectively, while that allocated from CAP$_1$ to other users does not exceed 1.6×10^6cycles/s. Similarly, as observed from Fig. 10(b), we can see that for the

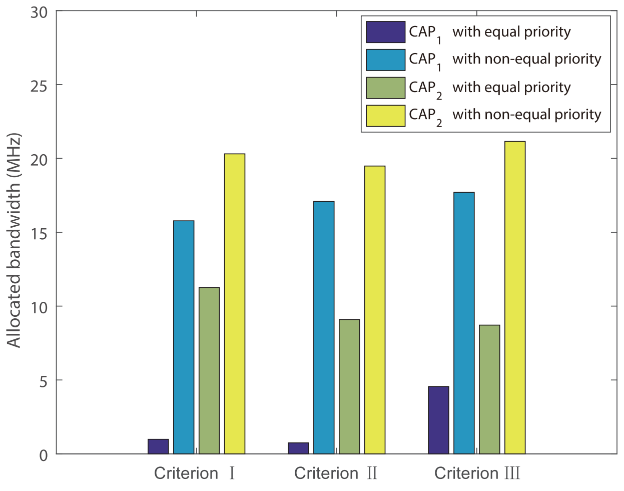


(a) CAP_1

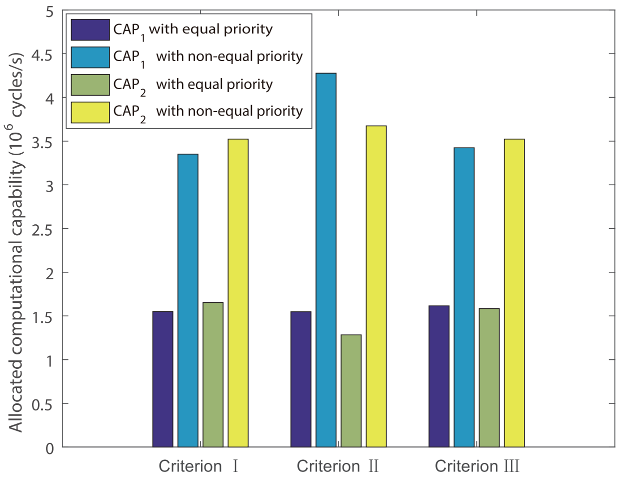


(b) CAP_2

Fig. 10. Allocated computational capability from CAPs to the users under the three criteria with $M = 5$ and $K = 2$.



(a) Allocated bandwidth



(b) Allocated computational capability

Fig. 11. Allocated resources from CAPs to the user SD_5 with equal and non-equal priorities under the three criteria.

three criteria, the bandwidth allocated from CAP_2 to the user SD_5 roughly reaches 3.5×10^6cycles/s, 3.7×10^6cycles/s and 3.6×10^6cycles/s, respectively, while that from CAP_2 to other users does not exceed 1.6×10^6cycles/s. Therefore, Fig. 10 verifies the effectiveness of the proposed priority based computational capability allocation policy. Fig. 11 compares the allocated bandwidth and computational capability from CAPs to the user SD_5 with equal and non-equal priorities under the three criteria, where $M = 5$, $K = 2$, and $\lambda = 0.5$. The equal task priority of the users is $\{1, 1, 1, 1, 1\}$, while the non-equal priority of the users is set to $\{1, 1, 1, 1, 3\}$. As observed from Fig. 11(a), we can find that for the three criteria, the bandwidth allocated from CAP_1 to SD_5 with a higher priority roughly reaches 15.7 MHz, 17.1 MHz and 17.7 MHz, respectively, while that allocated from CAP_1 to SD_5 with an equal task priority is about 1.0 MHz, 0.7 MHz and 4.5 MHz, respectively. Moreover, the bandwidth allocated from CAP_2 to SD_5 with a higher priority roughly reaches 20.3 MHz, 19.5 MHz and 21.1 MHz, respectively, while that allocated from CAP_2 to SD_5 with an equal priority is about 11.3 MHz, 9.1 MHz and 8.7 MHz, respectively. By comparing the allocated bandwidth of SD_5 with equal and non-equal priorities under three criteria, we can find that SD_5 with a higher priority can be allocated more bandwidth than SD_5 with an equal priority. Similarly, as observed from Fig. 11(b), we can see that for the three criteria, the computational capability allocated from CAP_1 to SD_5 with a higher priority roughly reaches 3.4×10^6cycles/s, 4.3×10^6cycles/s and 3.5×10^6cycles/s, respectively, while that allocated from CAP_1 to SD_5 with an equal priority is about 1.6×10^6cycles/s, 1.5×10^6cycles/s and 1.6×10^6cycles/s, respectively. In further, the computational capability allocated from CAP_2 to SD_5 with a higher priority roughly reaches 3.5×10^6cycles/s, 3.7×10^6cycles/s and 3.6×10^6cycles/s, respectively, while that allocated from CAP_2 to SD_5 with an equal priority is about 1.7×10^6cycles/s, 1.3×10^6cycles/s and 1.6×10^6cycles/s, respectively. By comparing the allocated computational capability of SD_5 with equal and non-equal priorities under three criteria, we can find that SD_5 with non-equal priority can be allocated more computational capability than SD_5 with equal priority. Therefore, Fig. 11 further

TABLE I
NOTATION LIST

Notation	Definition
M	Number of users
K	Number of CAPs
D_m	Task characteristics of the user SD$_m$
s_m	Size of the task D$_m$
d_m	Priority of the task D$_m$
$\alpha_{m,k}$	Task offloading ratio from the user SD$_m$ to CAP$_k$
$\alpha_{m,0}$	Local computation ratio of the task D$_m$
$\widetilde{\beta}_{m,k}$	Allocated bandwidth ratio from CAP$_k$ to the user SD$_m$
$\widetilde{\gamma}_{m,k}$	Allocated computational capability ratio from CAP$_k$ to SD$_m$
W_k^{total}	Bandwidth of CAP$_k$
f_k^{total}	Computational capability of CAP$_k$
$R_{m,k}^s$	Secrecy data rate of the link from the user SD$_m$ to CAP$_k$
$l_{m,k}^{tran}$	Transmission latency from SD$_m$ to CAP$_k$
$e_{m,k}^{tran}$	Transmission energy consumption from SD$_m$ to CAP$_k$
P_m^{comp}	Computational power at SD$_m$
P_k^{comp}	Computational power at CAP$_k$
l_m^{local}	Local computational latency of the task D$_m$
e_m^{local}	Local computational energy consumption of the task D$_m$
$l_{m,k}^{comp}$	Computational latency of the task D$_m$ at CAP$_k$
$e_{m,k}$	Computational energy consumption of the task D$_m$ at CAP$_k$
l_m^{total}	Latency of computing the task D$_m$
e_m^{total}	Energy consumption of the task D$_m$
L_{total}	Total latency of the system
E_{total}	Total energy consumption of the system
L_{th}	Latency threshold
E_{th}	Energy consumption threshold

TABLE II
PARAMETER SETTING

Parameter	Value
Average channel gain of h_m^{UAV}	0.01
Average channel gain of g_m	1
Average channel gain of h_m^k	1
Variance of the AWGN at the UAV	0.1
Variance of the AWGN at the CAPs	0.1
Peak interference power	2 W
Computational power at the local users	1 W
Computational power at the CAP	1 W
Step size ξ	0.1
Number of users	M
Number of CAPs	K
Bandwidth of CAP k	$\{50+k\}$ (MHz)
Computational capability of CAP k	$\{8+0.1k\}$ (10^6 cycles/s)
Local computational capability of user m	$\{5+0.1m\}$ (10^4 cycles/s)
Task size of user m (MB)	$[10+5m, 10+5(m+1)]$ $(m=0,1,\ldots,M)$
Task priority of users	$\{1,\cdots,1,3\}$

validates that the proposed allocation strategy can dynamically allocate the bandwidth and computational capability according to the task priority.

VI. Conclusion

In this work, we studied the FL framework for the system design of a multiuser MEC system, where multiple users had some computational tasks of different priorities, to be computed by multiple CAPs. We considered a cognitive eavesdropping environment where the users could use the wireless spectrum as long as the interference to the primary user was tolerated, and an eavesdropper in the network could overhear the confidential message from the users. For the considered system, we firstly presented three optimization criteria and then solved the optimization problem in a distributed machine learning way through optimizing the offloading ratio as well as the allocation of bandwidth and computational capability, by taking into account the task priority among users. In particular, a FL optimization framework was used, and each user employed DRL to solve the optimization. Simulation results were finally demonstrated to show that the proposed method can effectively reduce the system cost in terms of latency and energy consumption, and meanwhile ensure that more bandwidth and computational capability are allocated to the user with a higher task priority.

Regarding future works extending from this paper, we will take into account some discrepancy in the fairness among users, due to the parameters such as the channel condition, task size and computational capability. We can alleviate the unfair issue by some methods such as imposing some different weights on the SDs' latency and energy consumption. Moreover, as most of mobile devices are powered by a limited battery, it is hard to keep all mobile devices online. Therefore, how to provide an efficient and flexible energy supply to the mobile devices becomes a critical issue in the MEC networks. Motivated by this, we tend to exploit the energy harvesting model to solve the above issue in the future.

References

[1] J. Gong, Q. Kuang, and X. Chen, "Joint transmission and computing scheduling for status update with mobile edge computing," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[2] X. Lai and J. Xia, "Secure mobile edge computing networks in the presence of multiple eavesdroppers," *IEEE Trans. Commun.*, early access, 2022, doi: 10.1109/TCOMM.2021.3119075.

[3] C. Park and J. Lee, "Mobile edge computing-enabled heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1038–1051, Feb. 2021.

[4] C. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.

[5] X. Lai and J. Xia, "Outdated access point selection for mobile edge computing with cochannel interference," *IEEE Trans. Veh. Technol.*, early access, 2022.

[6] S. Tang and L. Chen, "Computational intelligence and deep learning for next-generation edge-enabled industrial IoT," *IEEE Trans. Netw. Sci. Eng.*, early access, 2022.

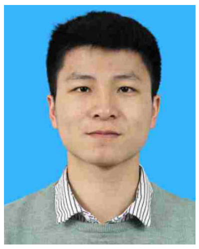
[7] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.

[8] X. He, R. Jin, and H. Dai, "Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 1814–1824, Mar. 2020.

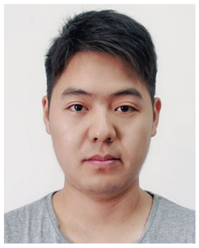
[9] M. Sheng, Y. Wang, X. Wang, and J. Li, "Energy-efficient multiuser partial computation offloading with collaboration of terminals, radio access network, and edge server," *IEEE Trans. Commun.*, vol. 68, no. 3, pp. 1524–1537, Mar. 2020.

[10] R. Malik and M. Vu, "Energy-efficient computation offloading in delay-constrained massive MIMO enabled edge network using data partitioning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 10, pp. 6977–6991, Oct. 2020.

[11] X. Hu, J. Wang, and C. Zhong, “Statistical CSI based design for intelligent reflecting surface assisted MISO systems,” *Sci. China, Inf. Sci.*, vol. 63, no. 12, 2020, Art. no. 222303.
[12] J. Zhang, Y. Zhang, C. Zhong, and Z. Zhang, “Robust design for intelligent reflecting surfaces assisted MISO systems,” *IEEE Commun. Lett.*, vol. 24, no. 10, pp. 2353–2357, Oct. 2020.
[13] Q. Tao, J. Wang, and C. Zhong, “Performance analysis of intelligent reflecting surface aided communication systems,” *IEEE Commun. Lett.*, vol. 24, no. 11, pp. 2464–2468, Nov. 2020.
[14] P. Yang, Y. Xiao, M. Xiao, Y. L. Guan, S. Li, and W. Xiang, “Adaptive spatial modulation MIMO based on machine learning,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 9, pp. 2117–2131, Sep. 2019.
[15] J. Chen *et al.*, “Hybrid beamforming/combining for millimeter wave MIMO: A machine learning approach,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 11353–11368, Oct. 2020.
[16] H. Liu, Y. Xiao, P. Yang, J. Fu, S. Li, and W. Xiang, “Transmit antenna selection for full-duplex spatial modulation based on machine learning,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10695–10708, Oct. 2021.
[17] J. Shi, J. Du, J. Wang, J. Wang, and J. Yuan, “Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16067–16081, Dec. 2020.
[18] S. Tang, “Dilated convolution based CSI feedback compression for massive MIMO systems,” *IEEE Trans. Veh. Technol.*, early access, 2022.
[19] L. He and K. He, “Towards optimally efficient search with deep learning for large-scale MIMO systems,” *IEEE Trans. Commun.*, early access, 2022.
[20] K. He and Y. Deng, “Efficient memory-bounded optimal detection for GSM-MIMO systems,” *IEEE Trans. Commun.*, early access, 2022.
[21] H. Cao and J. Cai, “Distributed multiuser computation offloading for cloudlet-based mobile cloud computing: A game-theoretic machine learning approach,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 752–764, Jan. 2018.
[22] R. Lin *et al.*, “Distributed optimization for computation offloading in edge computing,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 12, pp. 8179–8194, Dec. 2020.
[23] X. Cai, X. Mo, J. Chen, and J. Xu, “D2D-enabled data sharing for distributed machine learning at wireless network edge,” *IEEE Wireless Commun. Lett.*, vol. 9, no. 9, pp. 1457–1461, Sep. 2020.
[24] M. Wu, D. Ye, J. Ding, Y. Guo, R. Yu, and M. Pan, “Incentivizing differentially private federated learning: A multidimensional contract approach,” *IEEE Internet Things J.*, vol. 8, no. 13, pp. 10639–10651, Jul. 2021.
[25] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
[26] S. Zheng, C. Shen, and X. Chen, “Design and analysis of uplink and downlink communications for federated learning,” *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2150–2167, 2021.
[27] Y. Liu *et al.*, “A communication efficient vertical federated learning framework,” 2020, *arXiv:1912.11187*.
[28] W. Y. B. Lim *et al.*, “Federated learning in mobile edge networks: A comprehensive survey,” *IEEE Commun. Surv. Tut.*, vol. 22, no. 3, pp. 2031–2063, Mar. 2020.
[29] C. Shen, J. Xu, S. Zheng, and X. Chen, “Resource rationing for wireless federated learning: Concept, benefits, and challenges,” *IEEE Commun. Mag.*, vol. 59, no. 5, pp. 82–87, May 2021.
[30] R. Yu and P. Li, “Toward resource-efficient federated learning in mobile edge computing,” *IEEE Netw.*, vol. 35, no. 1, pp. 148–155, Jan. 2021.
[31] Z. Zhao, J. Xia, L. Fan, X. Lei, and G. K. Karagiannidis, “System optimization of federated learning networks with a constrained latency,” *IEEE Trans. Veh. Technol.*, early access, 2022, doi: 10.1109/TVT.2021.3128559.
[32] X. Huang, P. Li, R. Yu, Y. Wu, K. Xie, and S. Xie, “Fedparking: A federated learning based parking space estimation with parked vehicle assisted edge computing,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 9355–9368, Sep. 2021.
[33] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, and S. Yu, “Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14198–14211, Dec. 2020.
[34] N. Yang, L. Wang, G. Geraci, M. Elkashlan, J. Yuan, and M. D. Renzo, “Safeguarding 5G wireless communication networks using physical layer security,” *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 20–27, Apr. 2015.
[35] C. Liu, N. Yang, R. A. Malaney, and J. Yuan, “Artificial-noise-aided transmission in multi-antenna relay wiretap channels with spatially random eavesdroppers,” *IEEE Trans. Wireless Commun.*, vol. 15, no. 11, pp. 7444–7456, Nov. 2016.



Yinghao Guo received the bachelor’s degree in software engineering from the Zhengzhou University of Education, Zhengzhou, China, in June 2019. He is currently a Graduate Student with the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China. His current research interests include Machine Learning and mobile edge computing resource scheduling algorithms.



Rui Zhao received the B.E. degree in computer science and technology from Bohai University, Jinzhou, China, in 2018, and the master’s degree from the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China, in 2022. His current research interests include Machine Learning and mobile edge computing resource scheduling algorithms.

Shiwei Lai received the bachelor’s degree in computer science and technology from the Guangdong University of Education, Guangzhou, China, in June, 2019. She is currently a Graduate Student with the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China. Her current research interests include Machine Learning and mobile edge computing resource scheduling algorithms.

Lisheng Fan (Member IEEE) received the bachelor’s degree from the Department of Electronic Engineering, Fudan University, Shanghai, China, in 2002, the master’s degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2005, and the Ph.D. degree from the Department of Communications and Integrated Systems, Tokyo Institute of Technology, Tokyo, Japan, in 2008. He is currently a Professor with the School of Computer Science, GuangZhou University, Guangzhou, China. His research interests include wireless cooperative communications, physical-layer secure communications, intelligent communications, and system performance evaluation. He has authored or coauthored many papers in international journals such as IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON INFORMATION THEORY, and also papers in conferences such as IEEE ICC, IEEE Globecom, and IEEE WCNC. He is the Editor of the *China Communications*, and was the Guest Editor of many journals such as *Physical Communication*, *EURASIP Journal on Wireless Communications and Networking*, and *Wireless Communications and Mobile Computing*. He has been awarded as Exemplary Reviewer by IEEE TRANSACTIONS ON COMMUNICATIONS and IEEE COMMUNICATIONS LETTERS.

Xianfu Lei received the Ph.D. degree from Southwest Jiaotong University, Chengdu, China, in 2012. Since 2015, he has been an Associate Professor with the School of Information Science and Technology, Southwest Jiaotong University. From 2012 to 2014, he was a Research Fellow with the Department of Electrical and Computer Engineering, Utah State University, Logan, UT, USA. His research interests include 5G/6G networks, cooperative and energy harvesting networks, and physical-layer security. He is currently the Area Editor of IEEE COMMUNICATIONS LETTERS and an Associate Editor for IEEE WIRELESS COMMUNICATIONS LETTERS and IEEE TRANSACTIONS ON COMMUNICATIONS. He was the Senior Editor and an Associate Editor for IEEE COMMUNICATIONS LETTERS from 2014 to 2019. He was the recipient of the Best Paper Award in IEEE/CIC ICCC2020, the Best Paper Award in WCSP2018, the WCSP Ten-Year Anniversary Excellent Paper Award, IEEE Communications Letters Exemplary Editor 2019, and Natural Science Award of China Institute of Communications (2019).

George K. Karagiannidis (Fellow, IEEE) was born in Pithagorion, Samos Island, Greece. He received the University Diploma (five years) and the Ph.D. degree in electrical and computer engineering from the University of Patras, Patras, Greece, in 1987 and 1999, respectively. From 2000 to 2004, he was a Senior Researcher with the Institute for Space Applications and Remote Sensing, National Observatory of Athens, Athens, Greece. In June 2004, he joined the Faculty of the Aristotle University of Thessaloniki, Thessaloniki, Greece, where he is currently a Professor with the Electrical and Computer Engineering Department and the Head of the Wireless Communications and Information Processing (WCIP) Group. He is also an Honorary Professor with South West Jiaotong University, Chengdu, China. His research interests include the broad area of digital communications systems and signal processing, with emphasis on wireless communications, optical wireless communications, wireless power transfer and applications, and communications and signal processing for biomedical engineering. Dr. Karagiannidis has been involved as the General Chair, the Technical Program Chair, and a member of the Technical Program Committees in several IEEE and non-IEEE conferences. In the past, he was an Editor of several IEEE journals. From 2012 to 2015, he was the Editor-in-Chief of IEEE COMMUNICATIONS LETTERS. He is currently the Associate Editor-in-Chief for the IEEE OPEN JOURNAL OF COMMUNICATIONS SOCIETY. He is one of the highly-cited authors across all areas of Electrical Engineering, recognized from Clarivate Analytics as Web-of-Science Highly-Cited Researcher in the six consecutive years 2015–2020.