Optimizing the Learning Performance in Mobile Augmented Reality Systems with CNN

Yinghui He, Jinke Ren, Guanding Yu, and Yunlong Cai

***Abstract*—It is an essential goal for future wireless networks to provide better artificial intelligent services. In this paper, we investigate the joint communication and computation resource optimization in the mobile edge learning system to support augmented reality applications, where the convolutional neural networks (CNNs) are deployed at the edge server. For such a system, we first develop a delay model to characterize the relation between the computation latency and the input image size of general CNN models. Then, we formulate a mixed integer nonlinear optimization problem to maximize the system computation capacity under the constraints of learning accuracy, end-to-end latency, and energy consumption. To solve this problem, we first investigate maximizing the system learning accuracy under the communication and computation resource constraints. The optimal resource allocation policy can be achieved by a low-complexity search algorithm. We further prove that the original problem is NP-hard and propose an efficient heuristic algorithm with a newly-developed offloading priority function. An upper bound for the proposed algorithm is also derived. Finally, test results validate the applicability of the delay model and demonstrate the performance improvement of the proposed algorithm as compared with the existing algorithms.**

***Index Terms*—Mobile augmented reality, edge learning, learning accuracy, resource allocation, computation capacity, offloading priority.**

I. Introduction

In the next generation networks, giga-bit data rate, millisecond end-to-end latency, and wireless network intelligence need to be achieved to meet the developing requirements of mobile applications. To improve the data rate and reduce the communication latency, some techniques, such as massive multiple-input multiple-output (MIMO), millimeter wave (mmWave) communications, and small cells deployment have been proposed [2]. On the other hand, artificial intelligence (AI) has presented its powerful ability in solving intractable problems in many fields, such as data mining, natural language processing, and computer vision (CV) because of its strong capability of learning and inference. Therefore, wireless network intelligence can be achieved by the combination of wireless networks and AI, which has been regarded as an important research direction for future wireless networks [3].

Recently, wireless edge learning has been proposed by deploying AI algorithms at the network edge, i.e., cellular base stations (BSs) [4]. The study of wireless edge learning contains two trends, i.e., 1) AI for communications, where AI is considered as an effective tool to improve the performance of wireless communication networks, such as channel estimation [4], resource allocation [6], [7], and signal detection [8]; 2) communications for AI, where wireless communication networks are designed to improve the learning (training and inference) performance of AI algorithms. For the training performance, a number of techniques have been proposed to improve the network intelligence, such as mobile edge computing (MEC) [9]–[11] and federated learning (FL) [12]–[15]. Specifically, FL is proposed as a novel AI training framework where model training is distributed in user devices and the training results are aggregated at the BS to ensure the user's privacy [12]. In [13], a FedAvg algorithm has been proposed to achieve high-quality learning models within a small number of communication rounds. The authors in [14] have proposed a broadband analog aggregation scheme to achieve the tradeoff between communication and learning in the wireless FL system. Moreover, an efficient joint batchsize selection and resource allocation policy has been developed for the FL system to accelerate the AI training process [15].

In the wireless edge learning system, it is essential to improve the quality of experience (QoE) of mobile users when AI applications are used in wireless communication systems, such as mobile augmented reality (MAR) applications and autonomous vehicles. Therefore, in this work, we focus on joint allocation of communication and computation resources to support the MAR applications, which provides users with immersive experience and has become one of the most popular mobile applications [16], [17]. The main component of MAR applications is the CV task, which is usually accomplished by the convolutional neural networks (CNN) due to its high learning accuracy. However, the processing latency and energy consumption for CNN models are generally too large to be tolerated. In this work, we consider offloading the CV tasks to the network edge server for processing. By this means, the end-to-end latency and energy consumption can be greatly reduced while the learning accuracy can be significantly improved at the same time. On the other hand, the CNN architecture

Manuscript received July 28, 2019; revised January 31, 2020; accepted April 30, 2020. This work was supported in part by the Natural Science Foundation of China under Grants 61671407, 61831004, and 61971376; the Open Research Fund of the State Key Laboratory of Integrated Services Networks, Xidian University, under Grant ISN18-13; the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under Grant LR19F010002; and the Fundamental Research Funds for the Central Universities. The associate editor coordinating the review of this paper and approving it for publication was D. Li. (*Corresponding author: G. Yu.*)

Y. He and G. Yu are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, and also with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China. (e-mail: {2014hyh, yuguanding}@zju.edu.cn).

J. Ren and Y. Cai are with the Zhejiang Provincial Key Laboratory of Information Processing, Communication and Networking, Zhejiang University, Hangzhou 310027, China. (e-mail: {renjinke, ylcai}@zju.edu.cn)

Part of this paper will be presented in IEEE ICC'2020, Dublin, Ireland, June 2020 [1].

and the input image size greatly affect the performance of the MAR applications since a deeper network can extract more information from a larger image but results in higher computational complexity.

To balance the learning accuracy and the computational complexity, several methods have been developed in the edge learning framework [18]–[23]. The authors in [18] have developed a software accelerator, namely DeepX, to divide large-scale AI models into several parts for parallel processing. This method has been further improved in [19] by exiting the CNN computation at an approximate intermediate layer. In [20], the authors have proposed a non-loss depth-wise image partitioning scheme to reduce the overall processing latency. Furthermore, the authors in [21] have characterized the relation between the computational complexity and the image size by curve fitting, and developed an efficient algorithm to achieve the tradeoff between the learning accuracy and the processing latency. This framework has been extended in [22] to achieve a better learning accuracy via optimally allocating the edge computation resource. Particularly, to balance the learning accuracy and the processing latency, the authors in [23] have developed an optimal selection policy between local computing and edge offloading under the constraints of bandwidth and energy consumption.

The aforementioned studies mainly focus on characterizing the relation between the CNN computational complexity and the image size by curve fitting and measurement. However, no theoretical model has been developed therein. Therefore, the results in these studies are not general and cannot be utilized for optimization. Moreover, joint communication and computation resource allocation has not been investigated, which is also essential to improve the system learning performance. Inspired by these issues, we consider the joint optimization of CNN structure and resource allocation to improve the performance of MAR applications in the mobile edge learning system. We first analyze the characteristics of general CNN models and develop a theoretical delay model to characterize the relation between the computational latency and the image size of general CNN models. Based on this, we formulate a mathematical optimization problem to maximize the system computation capacity, i.e., the number of supported devices, under the requirements of end-to-end latency, energy consumption, and learning accuracy. Our main result is that the optimal image size should be adaptively controlled according to the wireless condition for a better learning performance.

The main contributions of this work are summarized as follows.

- We propose a delay model to characterize the relation between the computational latency and the image size for general CNN models, which is also validated via test results. Based on this, we formulate a mixed integer optimization problem to maximize the number of supported devices under the end-to-end latency, energy consumption, and learning accuracy requirements, which is proved to be NP-hard.
- To solve the original problem, we first aim at maximizing the system learning accuracy for the scenario where the number of accessed devices is fixed. The optimal resource allocation policy is derived in closed-form and a one-dimensional search algorithm is developed to reduce the computational complexity.
- Based on above results in the simple scenario, we define a novel function to characterize the offloading priority of each device depending on the data rate and the local computation resource. Then, we develop an effective heuristic algorithm to solve the original problem. A theoretical upper bound for the proposed algorithm is derived and some special cases where the heuristic algorithm can achieve exactly the optimal one are also highlighted.

The rest of this paper is organized as follows. In Section II, we introduce the system model, establish the CNN delay model, and formulate the optimization problem. In Section III, the optimal solution to maximize the system learning accuracy for given the set of supported devices is developed. Section IV proposes an effective heuristic algorithm to solve the original problem. Test results are presented in Section V and the whole paper is concluded in Section VI.

II. System Model and Problem Formulation

In this section, we will first introduce the multi-user MEC system and establish the delay and energy consumption models. Based on this, we then formulate the optimization problem to improve the system computation capacity, i.e., the number of supported devices.

A. System Model

As shown in Fig. 1, we consider a multi-user MEC system with one BS and N devices, denoted by the set $\mathcal{N} = \{1, 2, \cdots, N\}$. Each device needs to accomplish a CV task, such as image classification or object detection, using classical CNNs. Owing to the limited computation resource of devices, the CNN is placed in the edge server located at the BS and each device can offload the task data, i.e., local images, to the edge server for data processing.

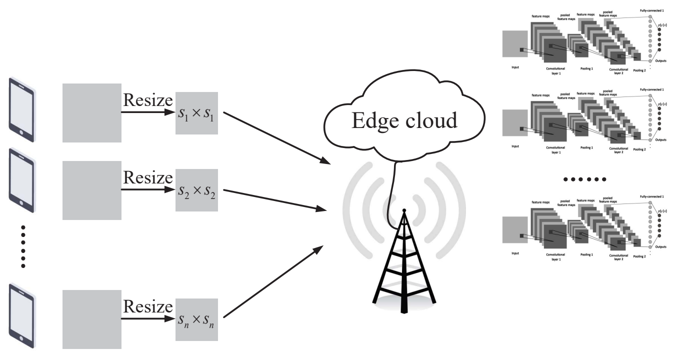

Fig. 1. Multi-user MEC system model.

To reduce the communication overhead of data offloading, each device should first compress the image resolution before transmission. After receiving the compressed images from all devices, the edge server will process them in parallel by sharing its computation resource. Finally, the computing results are sent back to each device. We should note that the delay for downloading computation results can be reasonably neglected because of their small data sizes. Moreover, the delay for local resizing should be considered since some devices may not have a high computation ability, such as Internet of things (IoT) devices. Therefore, the detailed process of each task includes three phases, i.e., *local resizing*, *image transmitting*, and *edge processing*.

B. Local Resizing

As mentioned above, each device shall compress the image data before transmission. In this paper, we assume that each device uses the bilinear interpolation method for image resizing [24]. Let s_n (in pixel) denote the width and height of the n-th device's image after local resizing. Then, the resolution of the compressed image can be expressed as s_n^2. Correspondingly, the total data-size of the compressed image is σs_n^2, where σ is the data-size for each pixel.

For the bilinear interpolation method, the RGB value of each new pixel is calculated based on those values of the four neighbor pixels in the original image. Therefore, the computational complexity of the bilinear interpolation method is proportional to the resolution of the compressed image. Consequently, the total computation cost (in CPU cycle) for resizing the image can be expressed as $C_n s_n^2$, where C_n represents the required number of CPU cycles per pixel. Let f_n (in CPU cycle/s) denote the local CPU-cycle frequency of device n with an upper bound $f_n^{\max}$. Then, the energy consumption in each CPU cycle is given by κf_n^2, where κ is a coefficient determined by the corresponding device and the original image [25]. Based on the above analysis, the latency and energy consumption for local resizing can be respectively expressed as

$$D_n^{\mathrm{L}} = \frac{C_n s_n^2}{f_n},\ \forall n \in \mathcal{N}, \tag{1}$$

$$E_n^{\mathrm{L}} = \kappa C_n f_n^2 s_n^2,\ \forall n \in \mathcal{N}. \tag{2}$$

It is worth mentioning that although we adopt the bilinear interpolation method for analysis, the delay and energy model in above can be directly utilized for other methods, e.g., bicubic interpolation method. The computational complexity of those other interpolation methods still keeps proportional to the resolution of the compressed image.

C. Image Transmission

In this work, we adopt the time-division multiple access (TDMA) method for channel access. Specifically, each time frame is divided into N time-slots for data transmission. Let t_n denote the proportion of device n's time-slot in one time frame. Define B and N_0 as the system bandwidth and the channel noise power, respectively. Let h_n denote the channel gain of device n, which is a random variable and varies with time. Denote p_n as the transmit power of device n. Since each time frame (10 ms in LTE standard) is much shorter than the transmission time (about 100ms to 1s in test), the transmission procedure will experience multiple time frames. Therefore, according to [10], we can use the average data rate to approximately evaluate the image transmission delay, as

$$R_n = B\mathbb{E}_h\left\{\log_2\left(1+\frac{|h_n|^2 p_n}{N_0}\right)\right\},\ \forall n \in \mathcal{N}, \tag{3}$$

where $\mathbb{E}_h\{\cdot\}$ is the expectation over the channel gain h_n. Based on this, the average image transmission delay of each device can be expressed as

$$D_n^{\mathrm{T}} = \frac{s_n^2 \sigma}{t_n R_n},\ \forall n \in \mathcal{N}. \tag{4}$$

It should be noted that in one time frame, only t_n proportion time is allocated to device n for image transmission, which consumes transmit energy. However, the other $1-t_n$ proportion time is its waiting delay so that no energy is needed during this period. Therefore, the energy consumption for each device to transmit its image can be expressed as

$$E_n^{\mathrm{T}} = \frac{s_n^2 \sigma}{R_n} p_n,\ \forall n \in \mathcal{N}. \tag{5}$$

D. Edge Processing

After receiving the images from all devices, the edge server will start image processing with a pre-determined CNN. Traditional CNNs require a fixed size of input image, which cannot be guaranteed in this work since different devices may have different image sizes. To address this issue, we use a spatial pyramid pooling layer to replace the fully-connected layer in classical CNN model [26]. By this means, the new CNN model can process images without considering the image size.

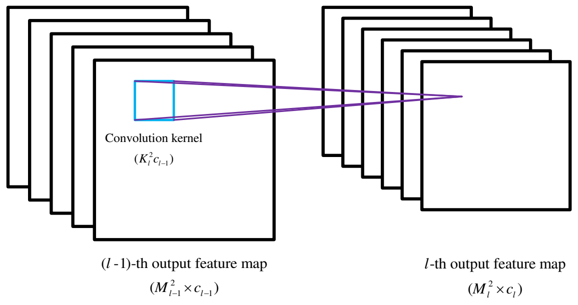
Convolution kernel
$(K_l^2 c_{l-1})$
$(l$ -1)-th output feature map
$(M_{l-1}^2 \times c_{l-1})$
l-th output feature map
$(M_l^2 \times c_l)$

Fig. 2. The l-th convolutional layer.

Now we begin to analyze the characteristics of the CNN model. The most important part of the CNN is the convolutional layer, which causes the main computation latency in edge processing. Therefore, we mainly focus on analyzing the convolutional layers, as shown in Fig. 2. Let L denote the number of convolutional layers in the CNN model. According to [27], the time complexity of the l-th convolutional layer can be expressed as

$$T_l^{\mathrm{CNN}} = \mathcal{O}\left(c_{l-1} c_l K_l^2 M_l^2\right),\quad l = 1,2,\cdots,L, \tag{6}$$

where c_{l-1} is the number of input channels, c_l is the number of output channels, K_l^2 is the size of convolution kernel, and M_l^2 is the size of output feature map. Note that L, c_{l-1}, c_l, and K_l are all determined by the CNN architecture and M_l

depends on the input image size. Specifically, according to [28], the relation between the output feature size M_l and the input image size s can be expressed as

$$M_l = \begin{cases} \dfrac{M_{l-1} - K_l + 2d_l}{r_l} + 1, & l = 2, 3, \cdots, L, \\ \dfrac{s - K_1 + 2d_1}{r_1} + 1, & l = 1, \end{cases} \tag{7}$$

where r_l is the stride (step size) and d_l is the padding size. Based on this, we can conclude that the size of output feature map in each convolutional layer is proportional to the input image size, indicating that the computational complexity of each convolutional layer is proportional to the input image resize s^2. Then, the total computational complexity of the whole CNN is given by

$$T^{\text{CNN}} = \mathcal{O}\left(s^2\right). \tag{8}$$

In this paper, we assume that the edge server processes images by CPUs[1]. Let F^{e} (in CPU cycle/s) denote the overall computation resource of the edge server, which will be allocated to all devices for parallel computing. Denote f_n^{e} as the computation resource allocated to device n, which satisfies $\sum_{n\in\mathcal{N}} f_n^{\text{e}} \le F^{\text{e}}$. Then, the edge processing delay of device n can be expressed as

$$D_n^{\text{e}} = \frac{C^{\text{e}} s_n^2}{f_n^{\text{e}}}, \ \forall n \in \mathcal{N}, \tag{9}$$

where C^{e} is the computation cost of the CNN per pixel.

Based on the above analysis, the end-to-end latency and energy consumption of each device can be expressed as

$$D_n = D_n^{\text{L}} + D_n^{\text{T}} + D_n^{\text{e}}, \ \forall n \in \mathcal{N}, \tag{10}$$

$$E_n = E_n^{\text{L}} + E_n^{\text{T}}, \ \forall n \in \mathcal{N}, \tag{11}$$

respectively.

E. Learning Accuracy

In general MAR applications, the learning accuracy is one of the most important issues that affect the QoE of mobile users. According to [21], the learning accuracy generally increases with the input image size when the image size is below a threshold. To validate this result, we adopt the SPP-net for image classification, which is constituted by replacing the last pooling layer in 34-layer ResNet [29] with a spatial pyramid pooling layer. The model is trained on Caltech101 dataset [31] with 101 categories and the experimental result is shown in Fig. 3. We shall note that the original image size in the dataset is roughly 300×200 pixels and resizing the original image to larger size would not improve the learning accuracy performance. From Fig. 3, we can observe that the learning accuracy first increases when input image size is smaller than 200^2 pixels. However, it keeps almost unchanged when the image size is larger than 200^2 pixels.

Based on the above discussions, we can define a monotone non-decreasing function $\phi(\cdot)$ to describe the relation between

[1]Note that our work can be also extended to the GPU scenario since GPU has a similar delay and energy consumption model with CPU [15], [22], [23].

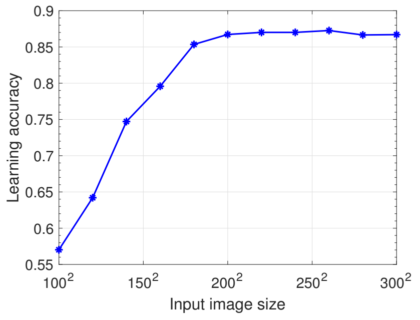

Fig. 3. Learning accuracy vs. input image size.

the learning accuracy and the input image size. Specifically, the learning accuracy of device n can be expressed as $\phi(s_n^2)$.

F. Problem Formulation

In this paper, we aim at maximizing the computation capacity of the whole system. Because of the limited communication and computation resources, not all devices can finish the tasks within the time limit. Therefore, we define the system computation capacity as the number of supported devices whose tasks can be finished within the required time limit T. Let π_n denote whether device n can finish its task on time, where $\pi_n = 1$ if device n succeeds and $\pi_n = 0$ if device n fails. Therefore, the objective function can be defined as $\sum_{n=1}^{N} \pi_n$, and the mathematical optimization problem can be formulated as

$$\mathcal{P}1: \max_{\left\{\substack{s_n, f_n, t_n, \\ f_n^{\text{e}}, \pi_n}\right\}} \sum_{n=1}^{N} \pi_n, \tag{12a}$$

$$\text{s.t.} \quad \phi(s_n^2) \ge \pi_n \alpha, \ \forall n \in \mathcal{N}, \tag{12b}$$

$$\pi_n \left(D_n^{\text{L}} + D_n^{\text{T}} + D_n^{\text{e}}\right) \le T, \ \forall n \in \mathcal{N}, \tag{12c}$$

$$\pi_n \left(E_n^{\text{L}} + E_n^{\text{T}}\right) \le E_n, \ \forall n \in \mathcal{N}, \tag{12d}$$

$$\sum_{n=1}^{N} \pi_n t_n \le 1, \tag{12e}$$

$$\sum_{n=1}^{N} \pi_n f_n^{\text{e}} \le F^{\text{e}}, \tag{12f}$$

$$0 \le f_n \le f_n^{\max}, \ \forall n \in \mathcal{N}, \tag{12g}$$

$$t_n, s_n, f_n^{\text{e}} \ge 0, \pi_n \in \{0, 1\}, \ \forall n \in \mathcal{N}, \tag{12h}$$

where T, α, and E_n denote the maximum latency requirement, the learning accuracy requirement, and the energy consumption limitation of device n, respectively. (12b) represents the learning accuracy requirement of each task, (12c) guarantees the delay tolerance of each task, (12d) bounds the maximal energy consumption of each device, (12e) is the total communication resource limitation, and (12f) limits the overall computation resource of the edge server.

The optimization variables in $\mathcal{P}1$ include image size (s_n), local CPU-cycle frequency (f_n), time-slot proportion (t_n), edge computation resource allocation (f_n^{e}), and completion indicator (π_n). Note that f_n cannot always be $f_n^{\max}$ since higher frequency will cost more energy and the total energy consumption should not exceed its limitation.

It can be easily proved that $\mathcal{P}1$ is a mixed integer non-linear problem and is hard to solve in general. To make it more tractable, we first consider a simple scenario where the set of supported devices is given (denoted by $\mathcal{I}$) and then determine whether the achievable learning accuracy can meet the corresponding accuracy requirement α. In this way, we can derive the relation between the system learning accuracy and the set of supported devices. Accordingly, the optimization problem associated with the simple scenario can be formulated to maximize the minimum accuracy among all device in $\mathcal{I}$, shown as

$$\begin{aligned}
\mathcal{P}2: \max_{\left\{\substack{s_n, f_n, \\ t_n, f_n^{\mathrm{e}}}\right\}} \quad & \min_{n\in\mathcal{I}} \left\{\phi(s_n^2)\right\}, && \text{(13a)}\\
\text{s.t.} \quad & D_n^{\mathrm{L}} + D_n^{\mathrm{T}} + D_n^{\mathrm{e}} \le T,\ \forall n\in\mathcal{I}, && \text{(13b)}\\
& E_n^{\mathrm{L}} + E_n^{\mathrm{T}} \le E_n,\ \forall n\in\mathcal{I}, && \text{(13c)}\\
& \sum_{n\in\mathcal{I}} t_n \le 1, && \text{(13d)}\\
& \sum_{n\in\mathcal{I}} f_n^{\mathrm{e}} \le F^{\mathrm{e}}, && \text{(13e)}\\
& 0 \le f_n \le f_n^{\max},\ \forall n\in\mathcal{I}, && \text{(13f)}\\
& t_n, s_n, f_n^{\mathrm{e}} \ge 0,\ \forall n\in\mathcal{I}. && \text{(13g)}
\end{aligned}$$

Then, if the optimal value of the objective function (13a) is larger than α, the solution to $\mathcal{P}2$ is also a feasible solution to $\mathcal{P}1$.

We shall note that the solution of $\mathcal{P}2$ is not the same as the solution of $\mathcal{P}1$. However, after $\mathcal{P}2$ is solved, we can compare its optimal learning accuracy with the minimum requirement in $\mathcal{P}1$. In this way, we can develop an effective algorithm to solve $\mathcal{P}1$, as elaborated in the following sections.

III. Maximizing Learning Accuracy

In this section, we first transform $\mathcal{P}2$ into a classical convex optimization problem and then derive the optimal solution in a closed-form way.

A. Optimal Solution to $\mathcal{P}2$

Recall that $\phi(\cdot)$ is a monotone non-decreasing function and $\mathcal{P}2$ aims at maximizing the minimum learning accuracy among all devices in $\mathcal{I}$. Therefore, the optimal solution is the same if we maximize $\min_{n\in\mathcal{I}}\{s_n^2\}$ instead of $\min_{n\in\mathcal{I}}\{\phi\left(s_n^2\right)\}$. Let $S = \max_{n\in\mathcal{I}}\{s_n^{-2}\}$. Then, maximizing $\min_{n\in\mathcal{I}}\{s_n^2\}$ is the same as minimizing S, which facilitates us to obtain the general results for CNN models. Thus, $\mathcal{P}2$ is equivalent to the following problem

$$\begin{aligned}
\mathcal{P}3: \min_{\left\{\substack{s_n, f_n, S, \\ t_n, f_n^{\mathrm{e}}}\right\}} \quad & S, && \text{(14a)}\\
\text{s.t.} \quad & \frac{C_n}{f_n} + \frac{\sigma}{t_n R_n} + \frac{C^{\mathrm{e}}}{f_n^{\mathrm{e}}} \le T s_n^{-2},\ \forall n\in\mathcal{I}, && \text{(14b)}\\
& \kappa C_n f_n^2 + \frac{\sigma p_n}{R_n} \le E_n s_n^{-2},\ \forall n\in\mathcal{I}, && \text{(14c)}\\
& s_n^{-2} \le S,\ \forall n\in\mathcal{I}, && \text{(14d)}\\
& \text{(13d)–(13g)}.
\end{aligned}$$

Since only (14b)–(14d) contain the variable s_n, we can combine these constraints without affecting the optimality of the solution, as

$$\frac{C_n}{f_n} + \frac{\sigma}{t_n R_n} + \frac{C^{\mathrm{e}}}{f_n^{\mathrm{e}}} \le TS,\ \forall n\in\mathcal{I}, \tag{15}$$

$$\kappa C_n f_n^2 + \frac{\sigma p_n}{R_n} \le E_n S,\ \forall n\in\mathcal{I}. \tag{16}$$

Therefore, $\mathcal{P}2$ can be reformulated into the following problem

$$\begin{aligned}
\mathcal{P}4: \min_{\{S, f_n, t_n, f_n^{\mathrm{e}}\}} \quad & S, && \text{(17)}\\
\text{s.t.} \quad & \text{(15), (16), (13d)–(13g)}.
\end{aligned}$$

We can determine whether the achievable learning accuracy of $\mathcal{I}$ can meet the corresponding accuracy requirement α by comparing the optimal value to $\mathcal{P}4$ with S_α that satisfies $\phi\left(S_\alpha^{-1}\right) = \alpha$. It can be easily verified that the objective function and all constraints in $\mathcal{P}4$ are convex, therefore it is a classical convex optimization problem. We can utilize some convex optimization tools to solve the problem. In the following, we will analyze the problem to gain some insightful results.

Based on the above analysis, we can obtain the optimal solution to $\mathcal{P}4$ using the method of Lagrange multiplier. The partial Lagrangian function can be defined as

$$\begin{aligned}
L =& S + \sum_{n\in\mathcal{I}} \lambda_n \left(\frac{C_n}{f_n} + \frac{\sigma}{t_n R_n} + \frac{C^{\mathrm{e}}}{f_n^{\mathrm{e}}} - TS\right) + \mu\left(\sum_{n\in\mathcal{I}} t_n - 1\right)\\
&+ \sum_{n\in\mathcal{I}} \omega_n \left(\kappa C_n f_n^2 + \frac{\sigma p_n}{R_n} - E_n S\right) + v\left(\sum_{n\in\mathcal{I}} f_n^{\mathrm{e}} - F^{\mathrm{e}}\right),
\end{aligned} \tag{18}$$

where λ_n, ω_n, μ, and v are the Lagrange multipliers associated with the constraints (15), (16), (13d), and (13e), respectively. Let $\{S^*, f_n^*, t_n^*, f_n^{\mathrm{e},*}\}$ denote the optimal solution to $\mathcal{P}4$. To better characterize the structure of the optimal solution, we first define an auxiliary *local resource function*, as

$$f_n(S) = \sqrt{\frac{E_n S R_n - \sigma p_n}{\kappa C_n R_n}}, \tag{19}$$

which is obtained according to (16) and represents the required local computation resource of each device when the image size of this device is S^{-1}. Then, based on the Karush-Kuhn-Tucker (KKT) conditions, we can derive the optimal solution to $\mathcal{P}4$ in closed-form expressions, as shown in the following theorem.

Theorem 1: The optimal solution to $\mathcal{P}4$ is given by

$$\begin{cases}
f_n^* = \min\left\{f_n^{\max}, f_n(S^*)\right\}, & \forall k\in\mathcal{I}, \quad \text{(20a)}\\
t_n^* = \dfrac{\sigma/R_n + C^{\mathrm{e}}/\sqrt{\theta^* R_n}}{TS^* - C_n/f_n^*}, & \forall k\in\mathcal{I}, \quad \text{(20b)}\\
f_n^{\mathrm{e},*} = \dfrac{\sqrt{\theta^*/R_n}\,\sigma + C^{\mathrm{e}}}{TS^* - C_n/f_n^*}, & \forall k\in\mathcal{I}, \quad \text{(20c)}
\end{cases}$$

where $\theta^* = \dfrac{C^{\mathrm{e}}\mu^*}{\sigma v^*}$. Moreover, S^* and θ^* satisfy the communication and computation resource limitations $\sum_{n\in\mathcal{I}} t_n^* = 1$ and $\sum_{n\in\mathcal{I}} f_n^{\mathrm{e},*} = F^{\mathrm{e}}$.

Proof: Please refer to Appendix A. ■

Remark 1: From Theorem 1, we can see that the optimal local CPU-cycle frequency is determined by the energy limitation and the corresponding upper limit, i.e., $f_n^{\max}$. When the device's energy is inadequate, the optimal local CPU-cycle frequency will increase with the energy limitation. However, when the local CPU frequency meets its upper limit, it keeps unchanged. Besides, the optimal communication and computation resource allocation policy is determined by the communication data rate, the local CPU-cycle frequency, and the required CPU cycles per pixel for local resizing together. When the wireless channel of a device becomes better, less communication resource and edge computation resource should be allocated to this device. This result is reasonable because it can ensure the fairness among all devices. Moreover, if the device costs more time in local resizing, more resources should be allocated to this device.

Until now, we have derived the optimal solution for $\mathcal{P}2$. Then, we have the following theorem to show the relation of the optimal image sizes among different devices.

Theorem 2: The optimal image sizes of all devices are the same, i.e., $(s_n^*)^2 = S^{-1}$, $\forall n \in \mathcal{I}$.

Proof: According to the proof in Appendix A, λ_n^*, $\forall n \in \mathcal{I}$ are larger than zero, which means that $\frac{C_n}{f_n^*} + \frac{\sigma}{t_n^* R_n} + \frac{C}{f_n^{\mathrm{e},*}} = TS^*$, $\forall n \in \mathcal{I}$. Meanwhile, we have $(s_n^*)^{-2} \le S^*$, $\forall n \in \mathcal{I}$ and $\frac{C_n}{f_n^*} + \frac{\sigma}{t_n^* R_n} + \frac{C}{f_n^{\mathrm{e},*}} \le T\left(s_n^*\right)^{-2}$, $\forall n \in \mathcal{I}$. Therefore, $(s_n^*)^2 = S^{-1}$, $\forall n \in \mathcal{I}$. ■

Remark 2: $\mathcal{P}2$ aims at maximizing the minimum accuracy among all devices in $\mathcal{I}$, which can be regarded as a classical fair resource allocation problem in wireless communication systems. From the results in Theorem 2, we can conclude that the fairness can be guaranteed when all devices achieve the same learning accuracy.

B. The Low-Complexity Algorithm

We now determine the optimal values of the two parameters S^* and θ^* in Theorem 1. Classical solution is to perform the time-consuming two-dimensional search algorithm. To reduce the computational complexity, we now develop an efficient algorithm to derive the optimal solution to $\mathcal{P}2$.

By analyzing the results in Theorem 1, we can reduce the number of variables by expressing θ^* as a function of S^*, as presented in Theorem 3.

Theorem 3: The optimal θ^* can be expressed as a function of S^*, i.e., $\theta^* = g_{\mathcal{I}}^2(S^*)$. Moreover, S^* satisfies $h_{\mathcal{I}}(S^*)g_{\mathcal{I}}(S^*) = 1$, where $h_{\mathcal{I}}(S)$ and $g_{\mathcal{I}}(S)$ are two monotone increasing functions, as

$$h_{\mathcal{I}}(S) = \left(\sum_{n\in\mathcal{I}} \frac{C^{\mathrm{e}}/\sqrt{R_n}}{TS - C_n/f_n(S)}\right)^{-1} \left(1 - \sum_{n\in\mathcal{I}} \frac{\sigma/R_n}{TS - C_n/f_n(S)}\right), \tag{21}$$

$$g_{\mathcal{I}}(S) = \left(\sum_{n\in\mathcal{I}} \frac{\sigma/\sqrt{R_n}}{TS - C_n/f_n(S)}\right)^{-1} \left(F^{\mathrm{e}} - \sum_{n\in\mathcal{I}} \frac{C^{\mathrm{e}}}{TS - C_n/f_n(S)}\right). \tag{22}$$

Proof: Please refer to Appendix B. ■

We shall note that $h_{\mathcal{I}}(S)$ is obtained from the communication resource constraint (13d) and $g_{\mathcal{I}}(S)$ is derived from the computation resource constraint (13e). They can be interpreted as the usage functions, which represent the usage of communication and computation resource, respectively. Both $h_{\mathcal{I}}(S)$ and $g_{\mathcal{I}}(S)$ decrease with the image size, $1/S$, which means that the resource utilization rate increases with $1/S$. When $h_{\mathcal{I}}(S^*)g_{\mathcal{I}}(S^*) > 1$, the communication and edge computation resources are not fully utilized. When $h_{\mathcal{I}}(S^*)g_{\mathcal{I}}(S^*) < 1$, the constraints of communication and edge computation resources cannot be satisfied. Only when $h_{\mathcal{I}}(S^*)g_{\mathcal{I}}(S^*) = 1$, all resources have been allocated to each device based on Theorem 1.

Based on Theorem 3, the method of two-dimensional search over S^* and θ^* can be transformed into the one-dimensional search over S^*. Specifically, the range of S^* is given in the following lemma.

Lemma 1: The optimal S^* should satisfy $S_{\min} \le S^* \le S_{\max}$, where

$$S_{\min} = \max_{n\in\mathcal{I}} \left\{\frac{\sigma p_n}{E_n R_n}\right\}, \tag{23}$$

$$S_{\max} = \max_{n\in\mathcal{I}} \left\{ \max\left\{ \frac{\kappa C_n \left(f_n^{\max}\right)^2}{E_n} + \frac{\sigma p_n}{R_n E_n}, \frac{C_n}{f_n^{\max} T} + \frac{\sigma I}{R_n T} + \frac{C^{\mathrm{e}} I}{F^{\mathrm{e}} T} \right\}\right\}, \tag{24}$$

where I is the size of the set $\mathcal{I}$.

Proof: Please refer to Appendix C. ■

Combining Theorem 3 with Lemma 1, we now develop a low-complexity algorithm to find the optimal solution to $\mathcal{P}2$, as shown in Table I. Note that in each iteration, the computational complexity for calculating $h_{\mathcal{I}}(S)g_{\mathcal{I}}(S)$ is $\mathcal{O}(N)$. Therefore, the total computational complexity can be expressed as $\mathcal{O}\left(N\log(1/\epsilon)\right)$, where ϵ is the maximum tolerance.

TABLE I
ONE-DIMENSIONAL SEARCHING ALGORITHM FOR $\mathcal{P}2$.

Algorithm 1 One-dimensional searching algorithm for $\mathcal{P}2$.

```
1  Set the maximal error tolerance $\epsilon$.
2  Set $S_\ell = S_{\min}$, $S_u = S_{\max}$, let $S = (S_\ell + S_u)/2$.
3  While $|h_{\mathcal{I}}(S)g_{\mathcal{I}}(S) - 1| > \epsilon$ do
4      if $h_{\mathcal{I}}(S)g_{\mathcal{I}}(S) > 1$, then
5          $S_u = S$.
6      else
7          $S_\ell = S$.
8      $S = (S_\ell + S_u)/2$.
9  End while
10 Output the optimal value $\phi\left(S^{-1}\right)$.
```

IV. MAXIMIZING SUPPORTED DEVICE'S NUMBER

In this section, we will first prove that $\mathcal{P}1$ is NP-hard and then develop an efficient heuristic algorithm to solve it. A theoretical upper bound is also developed to guarantee the performance of the proposed algorithm.

A. The Heuristic Algorithm

In $\mathcal{P}1$, the communication and computation resources should be jointly allocated to all devices. Therefore, $\mathcal{P}1$ is similar to the two-dimensional Knapsack problem that is generally NP-hard. Thus, we can give the following theorem.

Theorem 4: $\mathcal{P}1$ is an NP-hard problem.

Proof: Please refer to Appendix D. ■

According to Theorem 4, it is hard to find the optimal solution to $\mathcal{P}1$ with polynomial complexity. With this regard, we now start to develop a heuristic algorithm to find a suboptimal one. For ease of presentation, we first define the offloading priority function for each device, as

$$P_n = \frac{TS_\alpha - C_n/f_n(S_\alpha)}{\sigma/R_n + C^{\mathrm{e}}/F^{\mathrm{e}}},\ \forall n \in \mathcal{N}, \tag{25}$$

which represents the maximal image size that each device can achieve if all communication and computation resources could be allocated to it. Recall that the learning accuracy is positively related to the image size according to Fig. 3. Therefore, those devices with higher P_n can utilize the resource more efficient and improve the learning accuracy more quickly as compared with those devices with lower P_n.

Based on the offloading priority function, we can first rearrange all devices in non-increasing order, i.e. $P_{\tau(1)} \geq P_{\tau(2)} \geq \cdots \geq P_{\tau(N)}$, where $\boldsymbol{\tau} = \{\tau(1), \tau(2), \cdots, \tau(N)\}$ is a permutation on $\{1, 2, ..., N\}$. Then, we can sequentially add one device into the set $\mathcal{I}$ and check whether it is feasible to $\mathcal{P}1$ by comparing the optimal value to $\mathcal{P}2$ with the accuracy requirement. By applying the exhaustive search over the number of supported devices from 1 to N, we can find a solution to $\mathcal{P}1$ and then obtain the optimal resource allocation strategy according to Algorithm 1. In addition, we can easily check whether the set $\mathcal{I}$ in $\mathcal{P}2$ is also feasible for $\mathcal{P}1$, as shown in the following lemma.

Lemma 2: The optimal learning accuracy of $\mathcal{P}2$ can meet the requirement α when $h_\mathcal{I}(S_\alpha)g_\mathcal{I}(S_\alpha) \geq 1$.

Proof: Due to the fact that $h(S)$ and $g(S)$ are both monotone increasing functions, we have $S_\alpha \geq S^*$ only when $h_\mathcal{I}(S_\alpha)g_\mathcal{I}(S_\alpha) \geq h_\mathcal{I}(S^*)g_\mathcal{I}(S^*) = 1$. Further, the function $\phi(\cdot)$ decreases with S so that we have $\phi\left((S^*)^{-1}\right) \geq \phi(S_\alpha^{-1}) = \alpha$. ■

Based on the above analysis, the detailed procedures of the heuristic algorithm are presented in Table II. Similar to the analysis of Algorithm 1, the computational complexity of each iteration is $\mathcal{O}\left(\log(1/\epsilon)\right)$ where ϵ is the maximum tolerance and the maximal iterative number is N. Besides, the computational complexity of sorting algorithm is $\mathcal{O}\left(N \log N\right)$. Therefore, the total computational complexity can be expressed as $\mathcal{O}\left(N \log N + N \log(1/\epsilon)\right)$.

B. Performance Analysis and Discussions

We shall note that Algorithm 2 cannot always achieve the optimal solution to $\mathcal{P}1$ since the offloading priority function does not jointly consider the communication and computation resources. Take a three-device system as an example, where the offloading priority is $P_1 > P_2 > P_3$. Moreover, we assume that device 1 and device 2 have higher data rates

TABLE II
THE HEURISTIC ALGORITHM TO $\mathcal{P}1$.

Algorithm 2 The heuristic algorithm to $\mathcal{P}1$.
```
1 Calculate the offloading priority function $P_n$ according to (25).
2 Rearrange all devices in a non-increasing order according to their offloading priority and construct $\boldsymbol{\tau}$.
3 Set the number of supported devices $I = N + 1$.
4 Do
5     $I = I - 1$.
6     $\mathcal{I} = \{\tau(1), \tau(2), \cdots, \tau(I)\}$.
7 Until $h_\mathcal{I}(S_\alpha)g_\mathcal{I}(S_\alpha) \geq 1$.
8 Obtain the optimal resource allocation strategy according to Algorithm 1.
```

and need more computation resources to achieve the learning accuracy requirement while device 3 has lower data rate and requires more communication resources to achieve the learning accuracy requirement. Then according to Algorithm 2, only device 1 can be supported since the overall computation resource may not meet the requirements of both device 1 and device 2. However, it can support device 1 and device 3 simultaneously since they require different kinds of resources.

Therefore, in this section, we first develop an upper bound for the resource allocation problem $\mathcal{P}1$. Then, based on this, we further discuss some special scenarios where the proposed heuristic algorithm achieves exactly the optimal one. First, the constraints of $\mathcal{P}1$ can be transformed into the following constraint according to the proof of Theorem 4, as

$$1 - \sum_{n\in\mathcal{N}} \pi_n P_n^{-1} + \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{N}} \pi_n \pi_i G_{n,i} \geq 0, \tag{26}$$

where

$$G_{n,i} \triangleq \frac{C^{\mathrm{e}}\sigma\left(1/\sqrt{R_n} - 1/\sqrt{R_i}\right)^2}{2F^{\mathrm{e}}\left(TS_\alpha - C_n/f_n(S_\alpha)\right)\left(TS_\alpha - C_i/f_i(S_\alpha)\right)},\ n, i \in \mathcal{N}. \tag{27}$$

$G_{n,i}$ can be interpreted as the gain brought from device n and device i. In other words, adding those pairs of devices with larger $G_{n,i}$ will consume less resource and increase the number of supported devices to the maximum extent.

To calculate the detailed expression of the upper bound, we denote $\boldsymbol{\rho}$ as the non-ascending order of $G_{n,i}, i \in \{1, 2, \cdots, n-1\}, n \in \mathcal{N}$, i.e., $G_{\rho(1)} \geq G_{\rho(2)} \geq \cdots \geq G_{\rho(N(N-1)/2)}$. Moreover, let I^{HA} and I^{OPT} denote the objective value obtained from Algorithm 2 and the optimal algorithm, respectively. Then the corresponding upper bound of optimal solution can be presented in the following theorem.

Theorem 5: The upper bound of I^{OPT}, denoted by I^{UB}, should satisfy the following constraint

$$1 - \sum_{n=1}^{I} P_{\tau(n)}^{-1} + \sum_{n=1}^{I(I-1)/2} G_{\rho(n)} \geq 0. \tag{28}$$

Proof: Please refer to Appendix E. ■

From Theorem 5, we can find that the gap between I^{UB} and I^{HA} is caused by $G_{n,i}$, which means ignoring $G_{n,i}$ is the reason why the heuristic algorithm cannot achieve the optimal solution. Therefore, when $G_{n,i}$ is equal to zero, the heuristic algorithm is optimal. Then, we have the following three cases where the results are exactly the optimal ones.

Case 1 (Adequate computation resource): In this case, the edge computation resource is infinite, i.e., $F^{\mathrm{e}} \to \infty$. Then we can see that $G_{n,i}$ approaches zero. Therefore, the proposed Algorithm 2 can achieve the optimal solution since only communication resource should be allocated and the offloading priority can completely reflect the communication resource requirement for all devices.

Case 2 (Adequate communication resource): Similar to case 1, when the communication resource is infinite, only computation resource needs to be allocated. Therefore, the offloading priority function completely reflects the computation resource requirement for all devices so that the obtained result is the optimal one.

Case 3 (Identical data rate): In this case, the data rates of all devices are identical. Thus, $G_{n,i}$ equals zero according to (27). Therefore, the exclusive difference between all devices is their local computing resource, which has been completely considered in the offloading priority function. Therefore, the corresponding result is the optimal one.

V. Test Results

In this section, we conduct experiments to validate the proposed model and demonstrate the performance improvement of our algorithm.

A. Methodology

For the local resizing and image transmission, we adopt numerical simulations and the parameters are provided as follows unless otherwise specified. The BS has a coverage of 300 m and there are 10 devices randomly distributed in the area. The system bandwidth is 5 MHz unless otherwise stated and the noise spectral density is -174 dBm/Hz according to the long term evolution (LTE) standards [30]. The channel gains of cellular links are all generated according to independent and identically distributed (i.i.d.) Rayleigh random variables with unit variances. All devices have the same transmit power, i.e., $p_n = 24$ dBm, $\forall n \in \mathcal{N}$. The delay requirement is 0.4 s for all tasks and the maximum energy consumption is 0.2 J for all devices. The accuracy requirement is set to be 0.80 for all devices. We assume that the CPU frequency of each device, i.e., f_n, is uniformly generated from 1 to 2 GHz. The training dataset is Caltech101 [31]. All images are converted to BMP format before simulation and the data size per pixel is 24 bits/pixel. The required number of CPU cycles per pixel follows the uniform distribution within $C_n \in [1000, 2000]$ CPU cycles/pixel. The coefficient κ is set to be 10^{-28} according to [25].

For the edge processing, we conduct experiments to show the learning accuracy performance. As mentioned earlier, we adopt SPP-net constituted by the 34-layer ResNet and a spatial pyramid pooling layer for image classification. It is also trained on Caltech101. The SPP-net is implemented on the Linux server with Intel(R) Core(TM) i7-8700K CPU. To achieve the parallel processing and perform the proposed resource allocation policy, we utilize the tool *cpulimit* to limit the CPU usage of a process (expressed in percentage) [32]. The computation resource allocated to each device can be controlled with this tool and each device can run its own task with the allocated resource in parallel. Note that the maximal computation resource of the CPU is 1200% since the CPU has 12 cores. Here, we assume that the computation resource of the edge cloud is 300%. Prior to running the simulations, we measure the actual delay when the computation resource and the image size are both fixed. Then, we can obtain that the computation cost for the SPP-net is 0.0004143 s/pixel by fitting. The detailed procedures of the simulation and experiment are presented in Fig. 4.

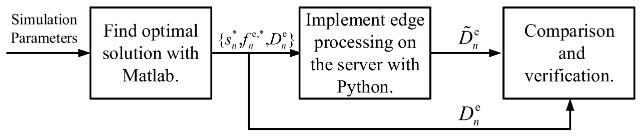
Simulation Parameters
Find optimal solution with Matlab.
$\{s_n^*, f_n^{\mathrm{e},*}, D_n^{\mathrm{e}}\}$
Implement edge processing on the server with Python.
$\tilde{D}_n^{\mathrm{e}}$
Comparison and verification.
D_n^{e}

Fig. 4. The detailed procedures of the simulation and experiment.

In our test, we first run the proposed Algorithm 2 in Matlab to find the solution. Then, the solution is used for test of edge processing. Finally, we compare the experimental results with the simulation results to demonstrate the effectiveness of the proposed delay model. Note that the delay of edge processing is experimental result while the delays of local resizing and image transmitting are simulation results. Through our test, the delay of edge processing contributes almost 80% to the total delay while the delays of local resizing and image transmitting contribute about 8% and 12%, respectively.

B. Model Verification

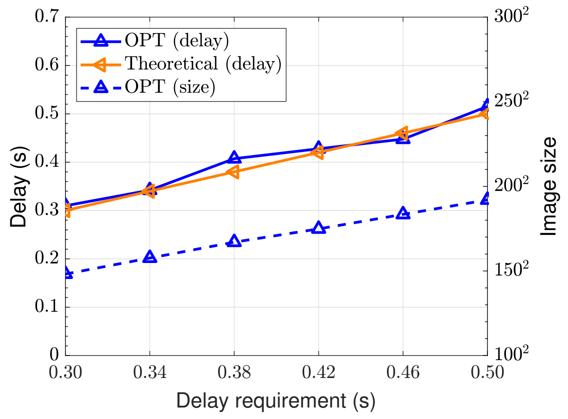

Fig. 5. The delay and optimal image size vs. the delay requirement. The curve, OPT (delay), is the experimental delay of the optimal solution to $\mathcal{P}2$. The curve, theoretical (delay), is the theoretical delay of the optimal solution to $\mathcal{P}2$. The curve, OPT (size), is the optimal image size to $\mathcal{P}2$.

Fig. 5 shows the actual delay and the optimal image size by running the proposed Algorithm 1 with different delay requirements. As mentioned earlier, only the delay of edge processing is obtained from experiment, which contributes almost 80% to the total delay. It can be seen that the optimal image size will increase with the increasing delay requirement since larger image size can be used for better accuracy. Moreover, the test delay almost equals the theoretical delay, which proves the accuracy of our model for edge processing.

C. Simulation Results of $\mathcal{P}2$

In this part, we will show the performance improvement of the proposed Algorithm 1 as compared with the baseline equal resource allocation scheme. For brevity, we abbreviate our proposed algorithm as *OPT* and the equal resource allocation scheme as *EQU*.

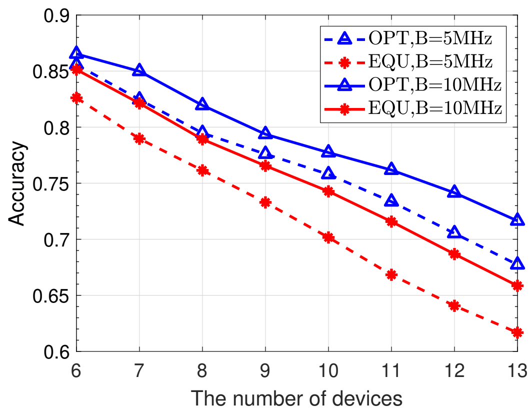

Fig. 6. The learning accuracy vs. the number of devices.

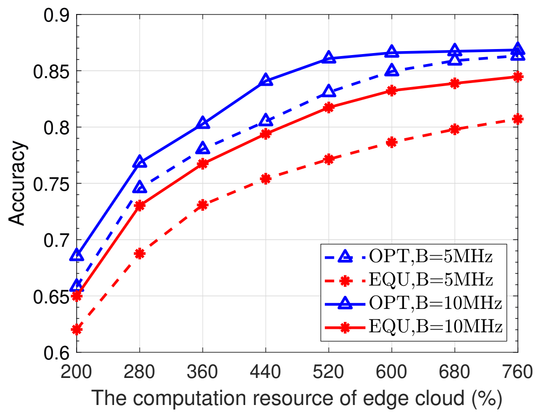

Fig. 7. The learning accuracy vs. the edge computation resource.

Fig. 6 depicts the learning accuracy versus the number of devices. It can be seen that the learning accuracy decreases with the number of devices. This is because that few communication and computation resources can be allocated to each device as the number of devices increases. Besides, compared with the equal resource allocation policy, our algorithm can achieve a significant performance improvement. Moreover, the learning accuracy also increases with the system bandwidth since more information can be transmitted to the edge server with more communication resources.

Fig. 7 shows the learning accuracy versus the edge computation resource. From the figure, we can find that the learning accuracy of both the proposed algorithm and the equal resource allocation policy increases with the edge computation resource. It is intuitive that the edge server can process images with higher resolution when its computation capacity increases. Particularly, the learning accuracy of the proposed scheme keeps invariant when the edge computation resource becomes large enough. The reason is that when the edge computation resource is large enough, the input image size of CNN is bigger than 200^2 pixels but the learning accuracy will not increase according to Fig. 3.

D. Simulation Results of $\mathcal{P}1$

In this section, we will show the effectiveness of the proposed Algorithm 2 (denoted by *HA*) as compared with the optimal solution (denoted by *OPT*) obtained from exhaustive searching. Moreover, the upper bound I^{UB} (denoted by *UB*) is also used to show the performance gap between the proposed algorithm and the optimal one.

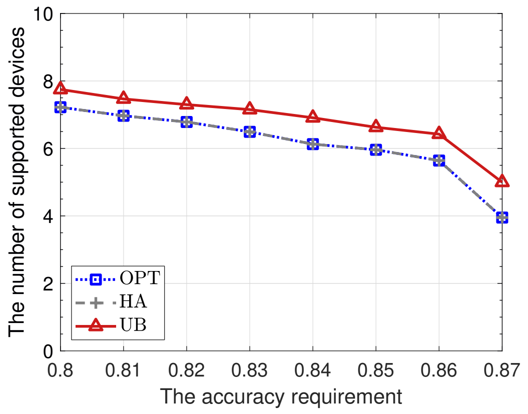

Fig. 8. The number of supported devices vs. the accuracy requirement.

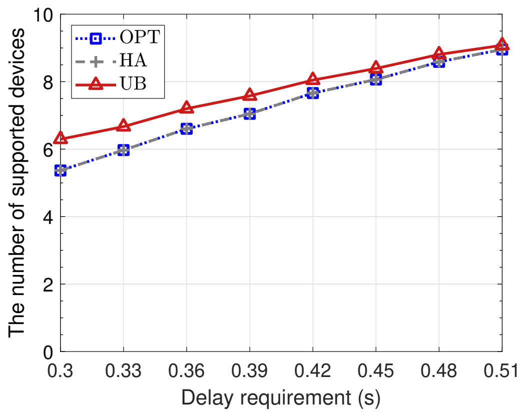

Fig. 9. The number of supported devices vs. the delay requirement.

Fig. 8 depicts the relation between the number of supported devices and the learning accuracy requirement. From this figure, we can see that our proposed algorithm can always achieve almost the same performance with the optimal one, which demonstrates the effectiveness of the proposed algorithm. Moreover, the number of supported devices decreases with the accuracy requirement. It is because that each supported device will occupy more resources with higher accuracy requirement. In particular, when the accuracy requirement is larger than 0.86, the number of supported devices drops evidently. The reason can be explained as follows. According to Fig. 3, the accuracy cannot be higher than 0.87 due to the limitation of the original image size under the given CNN model. Therefore, given learning accuracy above 0.86, it will consume much more resources to improve the learning accuracy such that fewer devices can finish their task on time.

Fig. 9 shows the number of supported devices versus the delay requirement. It can be seen that the number of supported devices increases with the delay requirement since loose delay requirement means less communication and computation resource are required for each device. Once again, the proposed heuristic algorithm can still achieve almost the same performance with the optimal one, which demonstrates the applicability of our proposal.

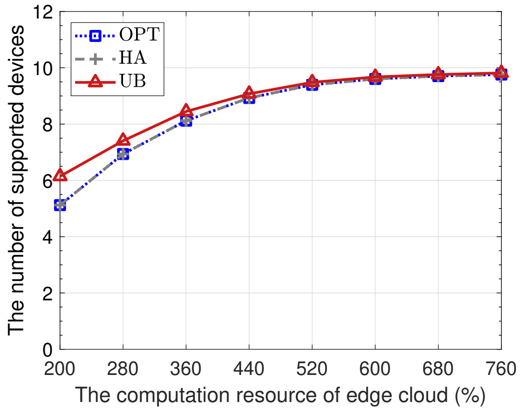

Fig. 10. The number of supported devices vs. the computation resource of edge cloud.

Fig. 10 depicts the number of supported devices versus the computation resource of the edge server. We can observe that the number of supported devices increases with the edge computation resource. Specifically, when the edge computation resource is large enough (more than 680% in our test), the supported number of devices keeps almost invariant. Moreover, the gap between the upper bound and the optimal solution becomes smaller when the edge computation resource increases.

VI. Conclusion

In this paper, we aim at maximizing the number of supported devices in a multi-user MEC system where each device uploads image to the edge cloud for recognition. A novel delay model for general CNNs is first developed to characterize the relation between the computation latency and the input image size. Thereafter, an NP-hard optimization problem is formulated to maximize the number of supported devices under the learning accuracy, end-to-end latency, and energy consumption constraints. To solve this problem, we first analyze a special case where the supported devices are given and turn to maximize the system learning accuracy. The joint optimal communication and computation resource allocation policy is developed in closed-form. Based on this, an efficient heuristic algorithm is proposed to solve the original problem by developing an offloading priority function for each device. Moreover, based on a theoretical upper bound for the optimal solution to original problem, we analyze three cases where the heuristic algorithm can achieve the optimal one. Numerical simulation and experimental results are finally conducted to demonstrate the effectiveness of the theoretical delay model and the performance improvement of the proposed algorithms.

Appendix A
Proof of Theorem 1

According to the KKT conditions, we can derive the following necessary and sufficient conditions, as

$$\frac{\partial L}{\partial S^*} = 1 - T\sum_{n\in\mathcal{I}}\lambda_n^* - \sum_{n\in\mathcal{I}}\omega_n^* E_n = 0, \tag{29}$$

$$\frac{\partial L}{\partial f_n^*} = -\lambda_n^*\frac{C_n}{(f_n^*)^2} + 2\omega_n^* C_n f_n^* \begin{cases} \geq 0, & f_n^* = 0, \\ = 0, & 0 < f_n^* < f_n^{\max}, \\ \leq 0, & f_n^* = f_n^{\max}, \end{cases} \tag{30}$$

$$\frac{\partial L}{\partial t_n^*} = -\lambda_n^*\frac{\sigma}{(t_n^*)^2 R_n} + u^* \begin{cases} \geq 0, & t_n^* = 0, \\ = 0, & t_n^* > 0, \end{cases} \tag{31}$$

$$\frac{\partial L}{\partial f_n^{\mathrm{e},*}} = -\lambda_n^*\frac{C^{\mathrm{e}}}{\left(f_n^{\mathrm{e},*}\right)^2} + v^* \begin{cases} \geq 0, & f_n^{\mathrm{e},*} = 0, \\ = 0, & f_n^{\mathrm{e},*} > 0, \end{cases} \tag{32}$$

$$\lambda_n^*\left(\frac{C_n}{f_n^*} + \frac{\sigma}{t_n^* R_n} + \frac{C^{\mathrm{e}}}{f_n^{\mathrm{e},*}} - TS^*\right) = 0, \tag{33}$$

$$\omega_n^*\left(\kappa C_n (f_n^*)^2 + \frac{\sigma p_n}{R_n} - E_n S^*\right) = 0, \tag{34}$$

$$\mu^*\left(\sum_{n\in\mathcal{I}} t_n^* - 1\right) = 0, \quad v^*\left(\sum_{n\in\mathcal{I}} f_n^{\mathrm{e},*} - F^{\mathrm{e}}\right) = 0, \tag{35}$$

$$\lambda_n^*, \omega_n^*, \mu^*, v^* \geq 0. \tag{36}$$

It can be easily verified that f_n^*, t_n^*, and $f_n^{\mathrm{e},*}$ are all positive according to (15) and (16). Therefore, according to (29), there exists at least one device whose λ_n^* and ω^* are also positive. We can discuss the following two cases.

Case 1: If there exists a device whose $\lambda_n^* > 0$, then μ^*, v^*, and $\lambda_n^*, \forall n \in \mathcal{I}$ must be larger than zero according to (31) and (32). Besides, we can derive that $f_n^{\mathrm{e},*} = \sqrt{\theta^* R_n} t_n$, where $\theta^* = \frac{C^{\mathrm{e}}\mu^*}{\sigma v^*}$. On the other hand, if $\omega_n > 0$, f_n^* can be derived as $\sqrt{\frac{E_n S^* R_n - \sigma p_n}{\kappa C_n R_n}}$ according to (34). Otherwise, f_n^* equals $f_n^{\max}$. Therefore, we have

$$\begin{aligned} f_n^* &= \min\left\{f_n^{\max}, \sqrt{\frac{E_n S^* R_n - \sigma p_n}{\kappa C_n R_n}}\right\} \\ &= \min\left\{f_n^{\max}, f_n(S^*)\right\}. \end{aligned} \tag{37}$$

Meanwhile, since $\mu^*, v^*, \lambda_n^* > 0, \forall n \in \mathcal{I}$, we have

$$\frac{C_n}{f_n^*} + \frac{\sigma}{t_n^* R_n} + \frac{C^{\mathrm{e}}}{f_n^{\mathrm{e},*}} - TS^* = 0, \tag{38}$$

$$\sum_{n\in\mathcal{I}} t_n^* - 1 = 0, \quad \sum_{n\in\mathcal{I}} f_n^{\mathrm{e},*} - F^{\mathrm{e}} = 0. \tag{39}$$

Combining (37), (38) with $f_n^{\mathrm{e},*} = \sqrt{\theta^* R_n} t_n$, the optimal solution to $\mathcal{P}2$ can be expressed as

$$t_n^* = \frac{\sigma/R_n + C^{\mathrm{e}}/\sqrt{\theta^* R_n}}{TS^* - C_n/f_n^*}, \tag{40}$$

$$f_n^{\mathrm{e},*} = \frac{\sqrt{\theta^*/R_n}\sigma + C^{\mathrm{e}}}{TS^* - C_n/f_n^*}. \tag{41}$$

Case 2: If there exists a device whose $\omega_n^* > 0$, then λ_n^* must be larger than zero according to (30). With the similar derivation, we can obtain the same results as those in case 1.

APPENDIX B
PROOF OF THEOREM 3

According to Theorem 1, S^* and θ^* should satisfy

$$\sum_{n\in\mathcal{I}} \frac{\sigma/R_n + C^{\mathrm{e}}/\sqrt{\theta^* R_n}}{TS^* - C_n/f_n(S^*)} = 1, \tag{42}$$

$$\sum_{n\in\mathcal{I}} \frac{\sqrt{\theta^*/R_n}\sigma + C^{\mathrm{e}}}{TS^* - C_n/f_n(S^*)} = F^{\mathrm{e}}, \tag{43}$$

which can be rewritten as $\frac{1}{\sqrt{\theta^*}} = h_{\mathcal{I}}(S^*)$ and $\sqrt{\theta^*} = g_{\mathcal{I}}(S^*)$. Therefore, we only need to search S^* until $h_{\mathcal{I}}(S^*)g_{\mathcal{I}}(S^*) = 1$.

Since $f_n(S)$ increases with S, $TS - C_n/f_n(S)$ also increases with S. Then, it is easy to find that both $h_{\mathcal{I}}(S)$ and $g_{\mathcal{I}}(S)$ are monotone increasing functions.

APPENDIX C
PROOF OF LEMMA 1

We can obtain the range of S^* by analyzing the following two cases.

Case 1: From (16), we can observe that $E_n S^*$ is larger than $\sigma p_n/R_n$, i.e., $S^* \geq \max_{n\in\mathcal{I}} \left\{ \frac{\sigma p_n}{E_n R_n} \right\}$. This gives the lower bound of S^*.

Case 2: Consider a special case where each device is allocated with the same communication and computation resource, i.e., $f_n^{\mathrm{e}} = F^{\mathrm{e}}/I, t_n = 1/I$, and the CPU frequency of each device is set to be $f_n = f_n^{\max}$. In this case, the objective function can be derived as (24). Since we aim at minimizing S in $\mathcal{P}4$, $S_{\max}$ can be regraded as an upper bound of S^*.

Combining the above two cases, we can obtain the range of S^* in Lemma 1.

APPENDIX D
PROOF OF THEOREM 4

We can rewrite $\mathcal{P}1$ into the following problem

$$\mathcal{P}5: \max_{\pi_n} \sum_{n=1}^{N} \pi_n, \tag{44a}$$

$$\text{s.t.} \quad m(\pi_1, \pi_2, \cdots, \pi_N) \geq a, \tag{44b}$$

$$\pi_n \in \{0, 1\}, \ n \in \mathcal{N}, \tag{44c}$$

where $a = 1$ and $m(\pi_1, \pi_2, \cdots, \pi_N)$ is given by

$$m(\pi_1, \pi_2, \cdots, \pi_N) = \left(\sum_{n\in\mathcal{N}} \frac{\pi_n C^{\mathrm{e}}/\sqrt{R_n}}{TS_\alpha - C_n/f_n(S_\alpha)}\right)^{-1} \left(1 - \sum_{n\in\mathcal{N}} \frac{\pi_n \sigma/R_n}{TS_\alpha - C_n/f_n(S_\alpha)}\right) \left(\sum_{n\in\mathcal{N}} \frac{\pi_n \sigma/\sqrt{R_n}}{TS_\alpha - C_n/f_n(S_\alpha)}\right)^{-1} \left(F^{\mathrm{e}} - \sum_{n\in\mathcal{N}} \frac{\pi_n C^{\mathrm{e}}}{TS_\alpha - C_n/f_n(S_\alpha)}\right). \tag{45}$$

Then, the constraint (44b) is quadratic. With simple mathematical calculations, we can observe that $\mathcal{P}5$ can be reduced into the following mixed integer quadratic program, as

$$\mathcal{P}6: \max_{\pi_n} \ m(\pi_1, \pi_2, \cdots, \pi_N), \tag{46a}$$

$$\text{s.t.} \quad \sum_{n=1}^{N} \pi_n = I, \tag{46b}$$

$$\pi_n \in \{0, 1\}, \ n \in \mathcal{N}, \tag{46c}$$

which is an NP-hard problem according to [33]. If $\mathcal{P}5$ can be solved, we can use the one-dimensional binary search over a until the optimal value of $\mathcal{P}5$ equals I. Then the corresponding a is also the optimal solution to $\mathcal{P}6$. However, since $\mathcal{P}6$ is NP-hard, we cannot solve it in polynomial complexity. Therefore, $\mathcal{P}5$ is also an NP-hard problem.

APPENDIX E
PROOF OF THEOREM 5

According to Appendix D, $\mathcal{P}1$ is equivalent to $\mathcal{P}5$ so that we can analyze the performance gap based on $\mathcal{P}5$. The constraint (44b) can be rewritten as (26). From it, we can find that $G_{n,i}$ is not included in the offloading priority function so that the proposed solution is not optimal. However, we can find an upper bound for the optimal solution I^{OPT} as follows. First, assume that $G_{n,i}$ is the largest one among all supported devices. Then, the upper bound, denote by I^{UB} should satisfy

$$1 - \sum_{n=1}^{I} P_{\tau(n)}^{-1} + \sum_{n=1}^{I(I-1)/2} G_{\rho(n)} \geq 0, \tag{47}$$

which ends the proof.

REFERENCES

[1] Y. He, J. Ren, G. Yu, and Y. Cai, "Optimizing the learning accuracy in mobile augmented reality systems with CNN," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, Jun. 2020.

[2] S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. C.K. Soong, and J. C. Zhang, "What will 5G be?," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[3] J. Park, S. Samarakoon, M. Bennis, and M. Debbah. "Wireless network intelligence at the edge," *Proc. IEEE*, vol. 107, no. 11, pp. 2204–2239, Nov. 2019.

[4] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," [Online]. Available: https://arxiv.org/abs/1809.00343

[5] P. Dong, H. Zhang, G. Y. Li, N. NaderiAlizadeh, and I. Gaspar, "Deep CNN for wideband mmWave massive MIMO channel estimation using frequency correlation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Brighton, UK., May 2019, pp. 4529-4533.

[6] H. Ye and G. Y. Li, "Deep reinforcement learning for resource allocation in V2V communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, May 2018, pp. 1–6.

[7] M. Lee, Y. Xiong, G. Yu, and G. Y. Li, "Deep neural networks for linear sum assignment problems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 6, pp. 962–965, Dec. 2018.

[8] P. Jiang, T. Wang, B. Han, X. Gao, J. Zhang, C.-K. Wen, S. Jin, and G. Y. Li, "Artificial intelligence-aided OFDM receiver: Design and experimental results," [Online]. Available: https://arxiv.org/abs/1812.06638.

[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tutor.*, vol. 19, no. 4, pp. 2322–2358, Aug. 2017.

[10] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.

[11] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5031–5044, Mar. 2019.

[12] J. Konecný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: distributed machine learning for on-device intelligence," [Online]. Available: https://arxiv.org/abs/1610.02527

[13] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," [Online]. Available: http://arxiv.org/abs/1602.05629

[14] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 491–506, Jan. 2020.

[15] J. Ren, G. Yu, and G. Ding, "Accelerating DNN training in wireless federated edge learning system", [Online]. Available: https://arxiv.org/abs/1905.09712

[16] D. Chatzopoulos, C. Bermejo, Z. Huang, and P. Hui, "Mobile augmented reality survey: From where we are to where we go," *IEEE Access*, vol. 5, pp. 6917–6950, Apr. 2017.

[17] J. Ren, Y. He, G. Huang, G. Yu, Y. Cai, and Z. Zhang, "An edge-computing based architecture for mobile augmented reality," *IEEE Netw.*, vol. 33, no. 4, pp. 162–169, Aug. 2019.

[18] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, L. Jiao, L. Qendro, and F. Kawsar, "DeepX: A software accelerator for low-power deep learning inference on mobile devices," in *Proc. 15th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw.*, Vienna, Austria, Apr. 2016, pp. 1–12.

[19] E. Li, Z. Zhou, and X. Chen, "Edge intelligence: On-demand deep learning model co-inference with device-edge synergy," in *Proc. Workshop Mobile Edge Commun. (MECOMM SIGCOMM)*, Budapest, Hungary, Aug. 2018, pp. 31–36.

[20] S. Dey, A. Mukherjee, A. Pal, and P. Balamuralidhar, "Partitioning of CNN models for execution on fog devices" in *Proc. 1st ACM Int. Workshop on Smart Cities and Fog Comput.*, Shenzhen, China, Nov. 2018, pp. 19–24.

[21] Q. Liu, S. Huang, J. Opadere, and T. Han, "An edge network orchestrator for mobile augmented reality," in *Proc. IEEE Int. Conf. Comput. Comnun. (INFOCOM)*, Honolulu, HI, Apr. 2018, pp. 756–764.

[22] Q. Liu and T. Han, "DARE: Dynamic adaptive mobile augmented reality with edge computing," in *Proc. IEEE Int. Conf. Netw. Protocols*, Cambridge, UK, Sep. 2018, pp. 1–11.

[23] X. Ran, H. Chen, X. Zhu, Z. Liu and J. Chen, "DeepDecision: A mobile deep learning framework for edge video analytics" in *Proc. IEEE Int. Conf. Comput. Comnun. (INFOCOM)*, Honolulu, HI, Apr. 2018, pp. 1421-1429.

[24] T. Lillesand and R. Kiefer, *Remote Sensing and Image Interpretation*, 3rd ed. New York: Wiley, 1994.

[25] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[27] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, Boston, MA, Jun. 2015, pp. 3791–3799.

[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, Jun. 2016, pp. 770–778.

[30] 3GPP, "LTE; Evolved universal terrestrial radio access (E-UTRA); Physical channels and modulation (3GPP TS 36.211 version 15.6.0 Release 15)," *TS 36 211 V15.6.0*, Jul. 2019.

[31] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Washington, DC, Jun. 2004, p. 178.

[32] A. Marletta, "Cpulimit." https://github.com/opsengine/cpulimit, 2012.

[33] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York: W. H. Freeman, 1979.

Yinghui He received the B.S.E. degree in information engineering from Zhejiang University, Hangzhou, China, in 2018. He is currently pursuing the master's degree with the College of Information Science and Electronic Engineering, Zhejiang University. His research interests mainly include mobile edge computing, device-to-device communications, and machine learning.

Jinke Ren (S'17) received the B.S.E degree in information engineering from Zhejiang University, Hangzhou, China, in 2017. He is currently pursuing the Ph.D. degree with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. Since December 2019, he has been working as a Visiting Research Scholar with the Department of Electrical and Computer Engineering, Northwestern University, IL, USA. His research interests include wireless edge intelligence, distributed machine learning, and 5G technologies such as mobile edge computing and device-to-device communication.

Guanding Yu (S'05-M'07-SM'13) received the B.E. and Ph.D. degrees in communication engineering from Zhejiang University, Hangzhou, China, in 2001 and 2006, respectively. He joined Zhejiang University in 2006, and is now a Full Professor with the College of Information and Electronic Engineering. From 2013 to 2015, he was also a Visiting Professor at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. His research interests include 5G communications and networks, mobile edge computing, and machine learning for wireless networks.

Dr. Yu has served as a guest editor of *IEEE Communications Magazine* special issue on Full-Duplex Communications, an editor of *IEEE Journal on Selected Areas in Communications* Series on Green Communications and Networking, and a lead guest editor of *IEEE Wireless Communications Magazine* special issue on LTE in Unlicensed Spectrum, and an Editor of IEEE Access. He is now serving as an editor of *IEEE Transactions on Green Communications and Networking* and an editor of *IEEE Wireless Communications Letters*. He received the 2016 IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award. He regularly sits on the technical program committee (TPC) boards of prominent IEEE conferences such as ICC, GLOBECOM, and VTC. He also serves as a Symposium Co-Chair for IEEE Globecom 2019 and a Track Chair for IEEE VTC 2019'Fall.

Yunlong Cai Yunlong Cai (S'07-M'10-SM'16) received the Ph.D. degree in electronic engineering from the University of York, York, U.K., in 2010. Since February 2011, he has been with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, where he is currently a Full Professor. From August 2016 to January 2017, he was a Visiting Scholar at the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. His research interests include transceiver design for multiple-antenna systems, mmWave communications, full-duplex communications, UAV communications and cooperative and relay communications. He is an associate editor of IEEE Signal Processing Letters.