UAV-Enabled Mobile-Edge Computing for AI Applications: Joint Model Decision, Resource Allocation, and Trajectory Optimization

Cailian Deng, Xuming Fang, *Senior Member, IEEE*, and Xianbin Wang, *Fellow, IEEE*

***Abstract*—Due to the flexible mobility and agility, unmanned aerial vehicles (UAVs) are expected to be deployed as aerial base stations (BSs) in future air–ground-integrated wireless networks, providing temporary and controllable coverage and additional computation capabilities for ground Internet of Things (IoT) devices with or without infrastructure support. Meanwhile, with the breakthrough of artificial intelligence (AI), more and more AI applications relying on AI methods such as deep neural networks (DNNs) are expected to be applied in various fields, such as smart homes, smart factories, and smart cities, to improve our lifestyles and efficiency dramatically. However, AI applications are generally computation intensive, latency sensitive, and energy consuming, making resource-constrained IoT devices unable to benefit from AI anytime and anywhere. In this article, we study mobile-edge computing (MEC) for AI applications in air–ground-integrated wireless networks. Our goal is to minimize the service latency while ensuring the learning accuracy requirements and energy consumption. To achieve that, we take DNN as the typical AI application and formulate an optimization problem that optimizes the DNN model decision, computation and communication resource allocation, and UAV trajectory control, subject to the energy consumption, latency, computation, and communication resource constraints. Considering the formulated problem is nonconvex, we decompose it into multiple convex subproblems and then alternately solve them till they converge to the desired solution. Simulation results show that the proposed algorithm significantly improves the system performance for AI applications.**

***Index Terms*—Internet of Things (IoT), mobile edge computing (MEC), resource allocation, trajectory control, unmanned aerial vehicle (UAV).**

I. Introduction

With the rapid development of 5G networks, Internet of Things (IoTs), and artificial intelligence (AI) technologies, a large number of AI applications relying on deep neural networks (DNNs) have been emerging in various application fields, ranging from manufacturing, medical service, finance, security, entertainment, education, transportation, and logistics. Advanced DNN models [e.g., artificial neural network (ANN), convolutional neural network (CNN), and deep reinforcement learning (DRL)] [1] could provide better learning results so that human roles can be replaced in many key work areas with high intensity, difficulty, and danger areas, such as virtual reality, intelligent surveillance, and autonomous driving. Compared with normal applications, AI applications relying on AI technologies have some new computational characteristics and new challenges. The implementation of AI applications often involves training and inference, that is, training the models through sample training, fitting, and environment interaction, and then using the trained models to process the data. Parameters of AI computing are large, requiring a large amount of computation, high storage capacity, and low-latency memory access capacity. Directly running complicated AI models, such as DNNs with high computing power requirements on resource-constrained IoT devices, will introduce long processing latency and high energy consumption, which will hinder the widespread use of AI technology. Thus, efficiently deploying AI applications with high requirements on computation and storage resources has become an urgent problem.

Today, most AI applications are deployed on the cloud to leverage vast computational resources to execute resource-demanding DNN models [2]–[4]. Unfortunately, cloud computing has one inherent limitation, i.e., the long transmission distance between the cloud and IoT devices, which often incurs unexpected latency, energy consumption, and packet loss issues. Besides, such cloud-based method is only applicable when network access is reliable. Thus, cloud computing is not suitable for a wide range of emerging latency-critical AI applications. Mobile-edge computing (MEC) has been recognized as a promising alternative to reduce execution latency and energy consumption [5]–[8]. By deploying extensive computation and storage resources to the network edge, such as Wi-Fi access points and base stations (BSs), the execution latency and energy consumption of AI applications can be significantly reduced. Many current works [9]–[14] mainly focus on developing MEC systems under the coverage of terrestrial networks to improve the performance of AI applications. However, limited by the network coverage, terrestrial networks

Manuscript received 26 August 2021; revised 29 October 2021 and 24 December 2021; accepted 7 February 2022. Date of publication 15 February 2022; date of current version 24 March 2023. The work of Cailian Deng and Xuming Fang was supported in part by NSFC under Grant 62071393; in part by NSFC High-Speed Rail Joint Foundation under Grant U1834210; in part by the Sichuan Provincial Applied Basic Research Project under Grant 2020YJ0218; and in part by the Fundamental Research Funds for the Central Universities under Grant 2682021CF019. *(Corresponding author: Xuming Fang.)*

Cailian Deng and Xuming Fang are with the Key Laboratory of Information Coding and Transmission, Southwest Jiaotong University, Chengdu 611756, China (e-mail: dengcailian@my.swjtu.edu.cn; xmfang@swjtu.edu.cn).

Xianbin Wang is with the Department of Electrical and Computer Engineering, University of Western Ontario, London, ON N6A 5B9, Canada (e-mail: xianbin.wang@uwo.ca).

Digital Object Identifier 10.1109/JIOT.2022.3151619

may fail to provide high-quality intelligent services with relatively high computing requirements for IoT devices anytime and anywhere. Furthermore, it is costly and impractical to densely deploy static edge servers for ubiquitous intelligent services in a realistic environment, such as disaster response, emergency relief, or rural environments. To tackle these challenges, the unmanned aerial vehicles (UAVs)-assisted MEC system has received growing popularity [15]–[20]. Due to the mobility, flexibility, and cost-effective features, UAVs can work as flying BSs to provide temporary and controllable coverage and additional communication and computation support for ground IoT devices, especially for wireless communication systems with limited or no available infrastructures.

However, developing a UAV-enabled MEC system for AI applications introduces several challenges. First, the high mobility of the UAV results in dynamic coverage and channel conditions, leading to intermittent connections and possibly increased communication latency. To guarantee the system performance, the UAV's location should be carefully designed to strengthen the coverage and provide better channel conditions for computation task offloading. Second, since the UAV equipped with edge servers is relatively resource restrained compared to the cloud server, designing an efficient resource allocation while considering the resource constraints and complex and dynamic network conditions becomes critical for UAV-enabled MEC for supporting computation-intensive, latency-sensitive, and energy-consuming AI applications. Third, besides the service latency and energy consumption, AI model accuracy becomes a key performance indicator of AI applications. When multiple indicators are involved in a service, we need to comprehensively consider the complex interaction of multiple indicators and network conditions. Fourth, to further improve the model accuracy, for an example, DNN models become deeper and require larger-scale input data [10], [13], [21], which introduces the long processing latency and high energy consumption. However, it may not always be the best choice for running the deepest DNN model, especially when accuracy, latency, and energy are important. Thus, how to choose the best one from available DNN models deployed on the UAV to meet the system performance requirements becomes an important issue.

To the best of our knowledge, the UAV-enabled MEC system for AI applications has not been well investigated. To facilitate the development and deployment of real-time AI applications in future air–ground-integrated wireless networks, we investigate the service latency minimization problem in an air–ground-integrated MEC network, while ensuring accuracy requirements and energy limitations. In addition, since the fixed-wing UAV can provide powerful transportation capability and longer service time than the rotary-wing UAV in an air–ground integrated MEC network, we adopt the fixed-wing UAV in our system as an example. The main contributions of this article are summarized as follows.

1) We consider an air–ground-integrated MEC system, where the UAV is equipped with an MEC server to provide computation service for ground IoT devices with limited processing capabilities and energy resources. In the air–ground-integrated MEC system, we first investigate the service latency minimization problem to facilitate the deployment and development of real-time AI applications.
2) We formulate the service latency minimization problem with learning accuracy, task processing latency, and energy consumption constraints, by jointly optimizing the DNN model decision, computation and communication resource allocation, and trajectory planing of the UAV. The formulated problem is a nonconvex mixed-integer nonlinear programming (MINLP) problem.
3) To address this MINLP problem, we transform the original problem into three more tractable subproblems, i.e., the DNN model decision given computation and communication resource allocation and UAV trajectory control, the optimal allocation of computation and communication resource according to the current model decision and UAV trajectory, and the UAV trajectory planing according to the current model decision and resource allocation. We solve these problems iteratively and show that our proposed UAV-enabled AI-computing system significantly reduces the total execution latency while ensuring that all tasks are successfully processed within the tolerable accuracy of the system and energy consumption of IoT devices.
4) Through trajectory optimization, the UAV is closer to its serving devices than the nonoptimized trajectories to provide better channel conditions and reduce the transmission latency. Besides, resource allocations and DNN model decisions are optimized according to wireless channel conditions and available resources on the UAV, to minimize the service latency. Simulation results demonstrate that the proposed algorithm can enhance the system performance significantly, compared with other benchmark schemes.

The remainder of this article is organized as follows. Section II presents the system model and problem formulation. In Section III, an efficient iterative search algorithm is proposed for service latency minimization. Simulation results are discussed in Section IV. Finally, this article is concluded in Section V.

II. Related Works

A. MEC for AI Applications

To reduce the execution latency and energy consumption of AI applications, many current works focus on deploying DNNs and running them on the network edge nearby ground IoT devices. In [10], an edge network orchestrator was designed to improve the responsiveness and analytics accuracy of the edge-based AI applications via optimally allocating the edge computation resource. The optimal selection of a deep learning algorithm between local computing and edge offloading was investigated in [11]. Particularly, for each AI task, each IoT device decides whether to process it locally or offload it to an edge network, depending on the tradeoffs between the model size, model accuracy, processing latency, battery level, and network conditions. A novel network protocol named DARE was designed in [12] to provide high-quality AI service

with edge computing, enabling ground IoT devices to dynamically adapt their application configurations and computation resource allocations on the edge server according to computation workloads and wireless channel conditions. Such dynamic configuration adaptations can significantly reduce the service latency in high dynamic edge environments. A novel QoS-guaranteed orchestration scheme for energy-efficient AI applications was proposed in [14] to minimize the intertwined costs, including accuracy loss, latency, and energy consumption, under the latency and accuracy constraints.

However, many previous works mainly focus on developing MEC systems under the coverage of terrestrial networks to improve the performance of AI applications, which makes AI applications impossible to be available anytime and anywhere. To provide low-latency AI service for remote IoT devices with limited or no available infrastructure coverage, our work investigates edge computing supported by the UAV equipped with a MEC server. Compared with the infrastructure-based MEC system, the UAV-enabled MEC system can facilitate the development and deployment of real-time AI applications in future air–ground-integrated wireless networks.

B. UAV-Enabled MEC

Currently, many research topics on UAV-enabled MEC system have been extensively investigated, including computation task offloading, latency reduction, and energy efficiency. In [22], an energy-efficient computation offloading problem with an emphasis on physical-layer security was investigated. In [15], the joint optimization of offloading and trajectory design was investigated to minimize the UAV energy consumption and task completion time, respectively. In [16], the minimization problem of the weighted sum energy consumption of the UAV and devices was investigated in the UAV-assisted MEC system, where the UAV acts as an MEC server and a relay to assist devices to compute their tasks or further relay their tasks to the access point for computing. To achieve a good tradeoff between computation and energy, computation efficiency, defined as the ratio of the total computation bits to the total energy consumption, was introduced in [17]. To enable massive connectivity in the IoT scenario, nonorthogonal-multiple-access (NOMA)-enabled MEC in multicell networks was studied in [18], in which multiple devices applied NOMA technique to offload their computation tasks to edge networks for high energy-efficient MEC. To mitigate sensor devices' energy and computing shortage issues, the UAV-enabled wireless-powered cooperative MEC system was considered in [19] and [20], where the UAV installed with an energy transmitter and a MEC server provides both energy and computing services for multiple devices. In [23], a multi-UAV-aided MEC system was proposed, where multiple UAVs act as MEC servers to provide computing services for ground IoT devices with limited local computing capabilities, and a load balancing algorithm was introduced to balance computational loads among UAVs. To minimize the sum energy consumption of UEs in a hybrid MEC network where there are ground stations, ground vehicles and UAVs, a hybrid deep learning-based online offloading algorithm was proposed by jointly optimizing the positions of ground vehicles and UAVs, resource allocation, and binary computation offloading and user association while considering the dynamic environment [24]. In [25], a UAV-enabled MEC system with stochastic computation tasks was investigated to minimize the average weighted energy consumption of devices and the UAV, subject to the constraints on task offloading, resource allocation, and UAV's trajectory scheduling. Task offloading decision, transmission bit allocation, and UAV trajectory have been jointly optimized in [26] to reduce the overall energy consumption in the UAV-enhanced edge system. In a multi-UAV aided MEC system, the total power minimization problem via jointly optimizing user association, power control, computation capacity allocation, and UAV location planning, was considered in [27]. To improve the transmission efficiency and minimize the response delay, a joint communication and computation optimization model was established for a UAV-enabled MEC network, which includes a centralized MEC-enabled top-UAV and a swarm of distributed bottom-UAVs [28].

Based on the extensive overview of existing works, we find that few efforts have been devoted to solve the service latency minimization problem for latency-critical AI applications in a UAV-enabled MEC system, which motivates the work in this article. In particular, our work clearly differs from the aforementioned works in the following aspects.

1) Many previous works mainly focus on developing MEC systems in typical cellular networks to improve the performance of AI applications, which makes AI applications impossible to be available anytime and anywhere. AI applications often not only focus on energy consumption and latency but also on the model accuracy. The optimization models of previous works in the UAV-enabled MEC system often cannot be directly applied to AI applications with intertwined performance indicators regarding accuracy, latency and energy consumption. Therefore, we investigate the service latency minimization problem in an air–ground-integrated MEC network, while ensuring accuracy requirements and energy limitations.
2) Different from the existing works on joint optimization of offloading decision, resource allocation, and trajectory control, we focus on joint optimization of AI model decision, resource allocation, and trajectory control. The decision making for AI model selection indicates that the UAV selects the proper AI model for each IoT device from a series of AI model candidates configured on the UAV under different QoS constraints and network conditions. However, in the existing works, the decision making usually refers to binary or partial offloading variable, which means that each IoT device decides to execute the task itself or offload the task to the UAV, or each IoT device can execute local computing and offload part of its tasks to the MEC server on the UAV.

III. System Model and Problem Formulation

As shown in Fig. 1, an air–ground-integrated wireless network is considered, which consists of I ground IoT devices

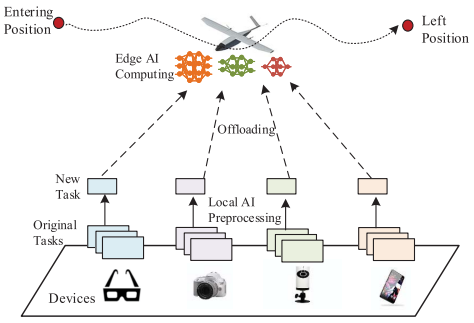

Fig. 1. Overview of the UAV-enabled MEC architecture.

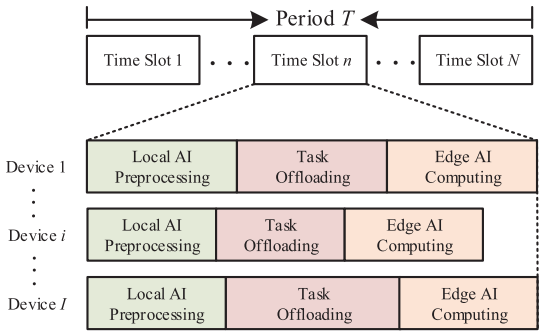

Fig. 2. Illustration of time sequence for local AI preprocessing, task offloading, and edge AI computing.

denoted as $\mathcal{I} = \{1, 2, \ldots, I\}$ and a single fixed-wing UAV equipped with a MEC server. The UAV deploys J well-trained DNN models $\mathcal{J} = \{1, 2, \ldots, J\}$ with different model input sizes to satisfy various QoS requirements of IoT devices, where a smaller DNN model can reduce the processing time and energy consumption at the cost of accuracy, and a larger AI model can increase the accuracy at the cost of longer processing latency and higher energy consumption. The UAV will choose the most appropriate DNN model for each IoT device for task processing based on network conditions and QoS requirements, such as accuracy, latency, and energy consumption. During an appointed flying period T, the UAV flies from its initial positions to the appointed final positions over ground IoT devices, and can provide powerful computation resource for ground IoT devices. For a given period T, IoT device i is continuously covered by the UAV and is always associated with the UAV. For convenience, we divide the period T into N time slots with equal length τ ($\tau = T/N$).

An AI application can be speech synthesis, speech recognition, image and video recognition, or text analysis. In this article, we take image recognition usually processed by DNNs as an example for analysis. We assume that each IoT device has an AI application to be accomplished with the help of the edge computing in UAV. We model an image recognition application as a chain of independent tasks, where each task is scheduled for task processing on a slot-by-slot basis.[1] Similar to many previous works [12], [29], [30], the UAV allocates available computation resources for different IoT devices by creating multiple virtual machines for independently processing different tasks. We introduce a binary variable $a_{ij}[n] = \{0, 1\}$ to distinguish DNN model selection decisions at time slot n. Specifically, $a_{ij}[n] = 1$ indicates that IoT device i decides to select the AI model j to execute task computing at time slot n, otherwise, $a_{ij}[n] = 0$. At time slot n, each task can only select one DNN model, i.e.,

$$\sum_{j \in \mathcal{J}} a_{ij}[n] = 1 \quad \forall i, n. \tag{1}$$

As the decision maker, the UAV collects the task processing requests from its associated IoT devices on the ground and decides on the task processing schedule and resource allocation according to the dynamic network environment, such as task processing requests and available network resources. The UAV forwards its scheduling decisions to the associated IoT devices for execution decisions. According to scheduling decisions and resource allocations, each IoT device will preprocess and offload for task computing at each time slot. To reduce the communication overhead of task offloading, each IoT device should first preprocess the task before offloading. After receiving the preprocessed tasks from IoT devices, the UAV can process them in parallel. Specifically, we assume that in each time slot, IoT device i starts task offloading only after the local preprocessing procedure has completed, and starts edge computing only after the offloading procedure has finished. Besides, we assume that the slot length τ is sufficiently small so that resources allocated to each IoT device can only be released at the end of each time slot. Therefore, in our work, we ignore queue delay for the tasks. Furthermore, computation results usually are smaller than task-input bits [27], [31]. Thus, the downloading time for computation results from the UAV to IoT devices is practically negligible. In this article, the detailed procedures of each task of each IoT device at each time slot include three phases, i.e., local preprocessing at IoT device i, task offloading from IoT device i to the UAV, and edge computing at the UAV, as illustrated in Fig. 2. Let $T_{ij}^r[n]$, $T_{ij}^t[n]$, and $T_{ij}^c[n]$ denote the duration of the local preprocessing phase, task offloading phase, and edge computing phase, respectively. Accordingly, for task execution of each task at time slot n, we have the following constraint as:

$$a_{ij}[n]\left(T_{ij}^r[n] + T_{ij}^t[n] + T_{ij}^c[n]\right) \leq \tau \quad \forall i, j, n. \tag{2}$$

A. Local Preprocessing

In this article, we take the image recognition application as an example. As mentioned above, each task is required to locally preprocess before task offloading. Similar to prior works [30], we assume that each IoT device uses the typical bilinear interpolation method for local preprocessing, i.e., task resizing. Let s_j^2 (in pixels) denote the resolution of the new task, where s_j is both the weight and height of the task after local preprocessing. Then, the total data size of the new task, same as the input size of the DNN model j, can be expressed as σs_j^2, where σ is the data size for each pixel. For

[1] We assume that each IoT device needs to accomplish the processing of a task in each time slot for convenience. In practice, our proposed approach also can be adapted to the system where the number of tasks and the number of time slots are different. When the time slot length τ is much shorter than the processing time, the task processing procedure will go through multiple time slots. When the time slot length τ is much longer than the processing time, the processing of multiple tasks can be accomplished in one time slot.

the bilinear interpolation method, its computational complexity is proportional to the resolution of the task. Let C_i represent the required CPU cycles per bit. Then, the computation latency (in second) and energy consumption (in joule) of local preprocessing at time slot n are, respectively, given by

$$T_{ij}^{r}[n] = \frac{C_i \sigma s_j^2}{f_i[n]} \quad \forall n, i, j \tag{3}$$

$$E_{ij}^{r}[n] = kC_i \sigma s_j^2 f_i^2[n] \quad \forall n, i, j \tag{4}$$

where $f_i[n]$ denotes the local computing resource of IoT device i, which is constrained by $0 \le f_i[n] \le f_i^{\max}$, where $f_i^{\max}$ is the maximum allowable computing resource of IoT device i. k is the effective switched capacitance determined by the corresponding device and the original task.

It is worth mentioning that although we take the image recognition application as an example for analysis, the above-mentioned latency and energy consumption models can also be adapted to other AI applications where the computational complexity is proportional to the task size and only the total data size of task is different.

B. Task Offloading

After local preprocessing, IoT devices will offload their unaffordable computing tasks to its associated UAV MEC server for task computing through the wireless channel. We assume that the UAV flies at a fixed height h above the ground. Suppose that the entering and left positions of the UAV are determined, whose coordinates are denoted as $\mathbf{q}_0$ and $\mathbf{q}_F$, respectively. The UAV trajectory is discretized into N line segments, which can be represented by the $(N+1)$ waypoints during the flying period T. The flying trajectory of the UAV at time slot n can be denoted as $\mathbf{q}[n] = (x[n], y[n])$. Then, the UAV's flying speed $\mathbf{v}[n]$ can be calculated by

$$\mathbf{v}[n] = \frac{\mathbf{q}[n+1] - \mathbf{q}[n]}{\tau} \quad \forall n. \tag{5}$$

The flying speed of the fixed-wing UAV at time slot n has a minimum speed $V_{\min}$ requirement to remain aloft, while cannot exceed its maximum speed $V_{\max}$. Hence, we have

$$V_{\min}^2 \le \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{6}$$

where $\|.\|$ denotes the Euclidean norm.

Suppose that the position of IoT device i is known, and its horizon coordinate is given by $\mathbf{u}_i = (x_i, y_i)$. According to the Euclidean formula, we can obtain the distance between the UAV and IoT device i at time slot n as

$$d_i[n] = \sqrt{h^2 + \|\mathbf{q}[n] - \mathbf{u}_i\|^2} \quad \forall i, n. \tag{7}$$

The slot length τ is sufficiently small so that $d_i[n]$ is approximately unchanged, and the channel gain is approximately unchanged within each time slot.

In the air–ground-integrated wireless network, due to the UAV's high altitude, the Line of Sight (LoS) link is much more dominant than other channel impairments, such as small-scale or shadowing fading. Therefore, referring to the existing works [31], we consider the wireless channel between IoT device i and the UAV as the free-space pathloss model. Moreover, the Doppler effect caused by the UAV mobility is considered to be well compensated at the UAV for simplicity. At time slot t, the LoS channel power gain from IoT device i to the UAV can be modeled as

$$h_i[n] = \beta_0 (d_i[n])^{-2} \quad \forall i, n \tag{8}$$

where β_0 is the channel power gain at the reference distance $d_0 = 1$ m.

Considering LTE or 5G-oriented systems, the orthogonal-frequency-division-multiple access (OFDMA) scheme is implemented for computation offloading in each time slot, so that interference among IoT devices can be avoided during the offloading process. Let $w_i[n]$ denote the communication resource allocated to IoT device i for task transmission at time slot n, which is constrained by the available communication resource limitation W, i.e.,

$$\sum_{i \in \mathcal{I}} w_i[n] \le W \quad \forall n. \tag{9}$$

When offloading, the transmission rate[2] between IoT device i and the UAV can be calculated by

$$R_i[n] = w_i[n] \log_2 \left(1 + \frac{h_i[n] p_i}{w_i[n] N_0}\right) \tag{10}$$

where N_0 represents the noise power spectral density of the UAV, and p_i is the transmit power of IoT device i.

The time required to offload a task and transmission energy consumption of IoT device i is expressed as

$$T_{ij}^{t}[n] = \frac{\sigma s_j^2}{R_i[n]} \quad \forall n, i, j \tag{11}$$

$$E_{ij}^{t}[n] = \frac{\sigma s_j^2}{R_i[n]} p_i[n] \quad \forall n, i, j. \tag{12}$$

Since the battery life of IoT device is relatively short and the battery level of IoT device is much lower than that of the UAV, this article only focuses on the energy consumption of the IoT device and does not focus on the energy consumption of the UAV during the flying period T. Note that since the size of computation results is much smaller than input data size [27], [31], the energy consumption for computation results transmitting back from the UAV to IoT device i is neglected. The total energy consumption of IoT device i mainly includes the preprocessing energy consumption caused by local preprocessing and transmission energy consumption due to task offloading. The total energy consumption of IoT device i during the period T is constrained by

$$\sum_{n \in \mathcal{N}} \sum_{j \in \mathcal{J}} a_{ij}[n] \left(E_{ij}^{r}[n] + E_{ij}^{t}[n]\right) \le E_i^{\max} \quad \forall i \tag{13}$$

where $E_i^{\max}$ is the residual energy of each device.

[2]When the time slot length τ is much shorter than the processing time, the task processing procedure may go through multiple time slots. In this case, we can directly adjust the constant value on the right-hand side of the inequation (2) to the sum of multiple time slots. In particular, when the transmission procedure goes through multiple time slots, we can use the average data rate of multiple time slots to evaluate the data transmission latency.

C. Edge Computing

Relatively, the UAV has limited computation resource compared to the cloud. Therefore, efforts for efficient computation resource allocation are necessary. Let $f_e^{\max}$ (in CPU cycle/s) denote the total available computation resource of the UAV, which will be allocated to all IoT devices for parallel task computing. Denote $f_i^e[n]$ as the computation resource allocated to IoT device i, which is constrained by the maximal computation capacity of the UAV, i.e.,

$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} a_{ij}[n] f_i^e[n] \le f_e^{\max} \quad \forall n. \tag{14}$$

Define C_e as the required CPU cycles per bit at the UAV. Then, the edge computing latency of IoT device i can be expressed as

$$T_{ij}^c[n] = a_{ij}[n]\frac{C_e \sigma s_j^2}{f_i^e[n]} \quad \forall n, i, j. \tag{15}$$

We assume that a task will not be processed until the UAV receives its entire input data. Moreover, we neglect the latency caused by signaling interaction between the UAV and IoT device i for task execution. Therefore, for each task, its end-to-end service latency mainly consists of the preprocessing latency due to resizing the task, transmission latency due to offloading, and computing latency due to the running DNN model. Thus, one has

$$T_{ij}[n] = T_{ij}^r[n] + T_{ij}^t[n] + T_{ij}^c[n] \quad \forall n, i, j. \tag{16}$$

D. Learning Accuracy

For an AI task, the learning accuracy is one of the most critical issues that affects the Quality of Experience (QoE) of IoT devices. Consider an example of an autonomous driving vehicle. Video data sensed or collected by the vehicle are processed in real time to detect nearby objects to avoid crashing with other vehicles. In order to ensure that no objects are missed in the video data, the target detection result should be as accurate as possible. The existing accuracy models are derived based on the performance measurements obtained from real experiments [10], [32], which generally increases with the input size under a fixed DNN model at the cost of higher computation energy consumption and longer processing time. According to the existing work [10], the learning accuracy highly depends on the input size of the DNN model, and the accuracy function with respect to the DNN model input size s_j^2 (in pixels) and model selection variable $a_{ij}[n]$ is modeled as a monotone nondecreasing function, i.e., $\phi(s_j^2, a_{ij}[n]) = a_{ij}[n](1 - 1.578\mathrm{e}^{-6.5\times10^{-3}s_j})$. According to the accuracy function, selecting the larger input size usually results in a better accuracy value.

In this article, we take object recognition usually processed by DNNs as an example for analysis. The learning accuracy of each task is defined as the ratio between the number of correctly recognized objects and that of total objects in a frame. For convenience, we consider that the computing result of the selected DNN model is unsatisfactory when the accuracy function value $\phi(s_j^2, a_{ij}[n])$ is lower than a predefined threshold A, otherwise, it is satisfactory. Due to the different types of services and application scenarios, multiple DNN models with different input sizes are available on the UAV. The model decision is undesired when a DNN model with an unsatisfactory accuracy function value is used by IoT device i at time slot n. To quantify the accuracy performance level of the system within the required period T, we define the accuracy level indicator of the system as the ratio of the total number of the undesired model decisions to the number of model decisions, i.e.,

$$\chi = \frac{\pi\left(\phi\left(s_j^2, a_{ij}[n]\right) \le A\right)}{\pi\left(\phi\left(s_j^2, a_{ij}[n]\right)\right)} \tag{17}$$

where $\pi(\cdot)$ denotes the number of model decisions. For example, there are three DNN models available on the UAV, corresponding to each DNN model, the input sizes are set as 100×100, 200×200, and 300×300 pixels, respectively. The larger the input data of the DNN model, the higher the model accuracy [10]. According to the accuracy function ϕ, the accuracy value of a DNN model with input size greater than or equal to 200×200 pixels is greater than 0.5. Suppose that selecting a DNN model with the accuracy value below the predefined threshold $A = 0.5$ is undesired. Each task is scheduled for processing on a slot-by-slot basis, and the number of model decisions of four IoT devices in four time slots is 48. Suppose that the total number of the undesired model decisions of all IoT devices is 12. The accuracy level indicator value of the system can be computed as $\chi = (12/48) = (1/4)$.

The accuracy performance indicator value χ reflects the overall learning accuracy level of the system and the service satisfaction of IoT devices. Specifically, the smaller the χ, the higher the overall learning accuracy performance of the system, which is due to the fact that IoT devices need to select larger size and more satisfactory DNN models to increase the learning accuracy at the cost of longer processing time and higher energy consumption.

E. Problem Formulation

Based on the above analysis, we investigate the joint optimization problem of the DNN model decision, resource allocation, and UAV's flying trajectory control in an air–ground-integrated wireless network. The AI-related computing task is latency sensitive and energy consuming, generally requiring high accuracy. Therefore, we focus on minimizing the total service latency of all tasks under learning accuracy requirement constraints, while keeping the tolerable device energy consumption via jointly optimizing the model decision $\mathbf{a} = \{a_{ij}[n]\}$, computation resource allocation $\mathbf{f} = \{f_i[n], f_i^e[n]\}$, communication resource allocation $\mathbf{w} = \{w_i[n]\}$, and UAV trajectory control $\mathbf{q} = \{q[n]\}$. Moreover, we assume that the channel gains between the UAV and all IoT devices are known. This latency minimization problem can be formulated as

$$\mathbf{P}: \min_{\mathbf{a},\mathbf{f},\mathbf{t},\mathbf{q}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} a_{ij}[n] T_{ij}[n]$$

$$\text{s.t.} \quad a_{ij}[n] \in \{0, 1\} \quad \forall n, i, j \tag{18a}$$

$$\sum_{j\in\mathcal{J}} a_{ij}[n] = 1 \quad \forall i, n \tag{18b}$$

$$\chi \le \chi_{\text{th}} \tag{18c}$$
$$V_{\min}^2 \le \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{18d}$$
$$\mathbf{q}[1] = \mathbf{q}_0, \mathbf{q}[N+1] = \mathbf{q}_F \tag{18e}$$
$$0 \le f_i[n] \le f_i^{\max} \quad \forall n, i \tag{18f}$$
$$\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} a_{ij}[n] f_i^e[n] \le f_{\text{e}}^{\max} \quad \forall n \tag{18g}$$
$$\sum_{i\in\mathcal{I}} w_i[n] \le W \quad \forall n \tag{18h}$$
$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}} a_{ij}[n]\left(E_{ij}^r[n] + E_{ij}^t[n]\right) \le E_i^{\max} \quad \forall i \tag{18i}$$
$$a_{ij}[n]\left(T_{ij}^r[n] + T_{ij}^t[n] + T_{ij}^c[n]\right) \le \tau \quad \forall i, j, n. \tag{18j}$$

Constraint (18a) denotes the binary model selection variables, and constraint (18b) reflects that only DNN model can be selected by each task. As mentioned above, the indicator value χ reflects the overall learning accuracy level of the system and indicates the service satisfaction of IoT devices. The learning accuracy requirement of the system is given in (18c), which denotes the accuracy performance indicator value of the system is less than or equal to the predefined threshold χ_{th} to ensure user service experience. The flying velocity constraint of UAV is shown in (18d). The entering and left positions of UAV during period T are given in (18e)–(18g), respectively, and state the constraints of maximal computation capacity of IoT device i and the UAV. Constraint (18h) is the communication resource allocation limitation. Constraint (18i) denotes that the energy constraint of IoT device i. Constraint (18j) denotes the service latency requirements.

It can be seen that problem P is a nonconvex MINLP problem, which is hard to solve directly. This is because the model decision is binary while the communication resource allocation, computation resource allocation, and flying trajectory control of UAV are continuous. Different variables exist nonlinear coupling. Furthermore, since the objective function and the constraints are nonconvex, problem P is a nonconvex optimization problem. To tackle problem P, we decompose the original problem into different subproblems and propose an alternative optimization algorithm to solve them iteratively.

IV. Proposed Latency Minimization Algorithm

In the proposed iterative optimization algorithm, the DNN model decision, communication resource allocation, computation resource allocation, and UAV's trajectory control are alternatively optimized. First, we optimize the DNN model decisions by given the communication resource allocation, computation resource allocation, and UAV's trajectory control. Then, we optimize the communication resource and computation resource allocation under the given DNN model decision and UAV's trajectory control. Finally, we optimize the UAV's trajectory control by given the DNN model decision, communication resource, and computation resource allocation.

A. Model Decision Optimization

It can be seen that the model decision in problem P is an integer programming process for the given communication resource, computation resource, and the flying trajectory of UAV. The DNN model decision in the original problem P can be reformulated as $SP1$

$$\textbf{SP1}: \min_{\mathbf{a}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} a_{ij}[n] T_{ij}[n]$$
$$\text{s.t.} \quad (18a)-(18c), (18g), (18i), (18j). \tag{19a}$$

In problem $SP1$, only binary variable remains, which is a standard integer linear programming problem. Therefore, we can utilize the existing optimization algorithms to solve this problem, such as branch-and-bound [33] and cutting plane methods [34].

B. Communication and Computation Resource Allocation Optimization

For any given DNN model decision and UAV trajectory control, the communication resource and computation resource allocation optimization in P can be reformulated as

$$\textbf{SP2}: \min_{\mathbf{f},\mathbf{w}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} a_{ij}[n]\left(\frac{C_i\sigma s_j^2}{f_i[n]} + \frac{C_e\sigma s_j^2}{f_i^e[n]} + \frac{\sigma s_j^2}{w_i[n]\log_2\left(1+\frac{h_i[n]p_i}{w_i[n]N_0}\right)}\right)$$
$$\text{s.t.} \quad (18f)-(18j). \tag{20a}$$

Lemma 1: $SP2$ is a convex optimization problem.

Proof: We define the function $\phi(x) = x\ln(1+[b/ax]), x \ge 0, a > 0, b > 0$, and its first-order derivative is given by

$$\phi'(x) = \ln\left(1+\frac{b}{ax}\right) - \frac{b}{b+ax} = \ln(y) + \frac{1}{y} - 1 \tag{21}$$

where $y = 1 + (b/ax) \ge 1$. By defining the function $\varphi(y) = \ln(y) + (1/y)$, we have $\varphi'(y) = (1/y) - (1/y^2) \ge 0$, and $\varphi(1) = 1$. Thus, we have $\varphi(y) \ge 1$ and $\phi'(x) \ge 0$. Besides, the second-order derivative of $\phi(x)$ is given by

$$\phi''(x) = -\frac{b^2}{x(b+ax)^2} < 0 \tag{22}$$

which indicates that $\phi(x)$ is concave with respect to x. Thus, $R_i[n]$ is concave with respect to $w_i[n]$ and $[1/(R_i[n])]$ is a convex function of $w_i[n]$. Accordingly, the objective function of $SP2$ is convex with respect to $w_i[n]$ and the constraints of (18j) are convex.

In addition, in problem $SP2$, $[(C_i\sigma s_j^2)/(f_i[n])]$ and $[(C_e\sigma s_j^2)/(f_i^e[n])]$ are, respectively, convex with respect to $f_i[n]$ and $f_i^e[n]$. According to [35], the nonnegative sum of multiple convex functions is still convex. Thus, the objective function of problem $SP2$ is convex with respect to $\mathbf{f} = \{f_i[n], f_i^e[n]\}$ and $\mathbf{w} = \{w_i[n]\}$. Constraint (18j) is convex with respect to $f_i[n]$. Therefore, the constraints of problem $SP2$ are convex sets. Furthermore, problem $SP2$ is a convex optimization problem, which can be solved by utilizing typical convex optimization algorithms, such as the Lagrange duality method or interior point method [35]. ■

C. Trajectory Optimization

Intuitively, due to the agility and flexible mobility, the UAV can fly over the regions closer to all ground IoT devices, which provides better channel conditions for computation task offloading. According to the path-loss channel model in (8), decreasing the distance between the device and the UAV may decrease the path-loss exponent factor, which will increase the transmission rate. Therefore, the UAV's trajectory needs to be optimized accordingly to provide better channel conditions for computation task offloading.

Under given offloading and model decision, and allocated resources, we can optimize the trajectory control of the UAV, for which the problem can be formulated as

$$\mathbf{SP3}: \min_{\mathbf{q}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \frac{a_{ij}[n]\sigma s_j^2}{w_i[n]\log_2\left(1+\frac{\beta_0 p_i/(w_i[n]N_0)}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)}$$

$$\text{s.t.}\quad V_{\min}^2 \le \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{23a}$$

$$\mathbf{q}[1]=\mathbf{q}_0, \mathbf{q}[N+1]=\mathbf{q}_F \tag{23b}$$

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}\left(E_{ij}^r[n]+\frac{a_{ij}[n]\sigma s_j^2 p_i}{w_i[n]\log_2\left(1+\frac{\beta_0 p_i/(w_i[n]N_0)}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)}\right) \le E_i^{\max} \quad \forall i \tag{23c}$$

$$T_{ij}^r[n]+\frac{a_{ij}[n]\sigma s_j^2}{w_i[n]\log_2\left(1+\frac{\beta_0 p_i/(w_i[n]N_0)}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)}+T_{ij}^c[n] \le \tau \quad \forall i,j,n. \tag{23d}$$

To analyze the convexity of problem $SP3$, we transform the original problem $SP3$ into the following equivalence problem $SP3'$ on the basis of the same constraints as problem $SP3$:

$$\mathbf{SP3}': \max_{\mathbf{q}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \frac{\log_2\left(1+\frac{C_{ij}[n]}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)}{D_{ij}[n]}$$

$$\text{s.t.}\quad V_{\min}^2 \le \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{24a}$$

$$\mathbf{q}[1]=\mathbf{q}_0, \mathbf{q}[N+1]=\mathbf{q}_F \tag{24b}$$

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}\frac{p_i}{D_{ij}[n]\log_2\left(1+\frac{C_{ij}[n]}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)} \le E_i^{\max}-\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}E_{ij}^r[n] \quad \forall i \tag{24c}$$

$$\frac{1}{D_{ij}[n]\log_2\left(1+\frac{C_{ij}[n]}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right)} \le \left(\tau-T_{ij}^r-T_{ij}^c\right) \quad \forall i,j,n \tag{24d}$$

where $C_{ij}[n] = [(\beta_0 p_i)/(w_i[n]N_0)]$, $D_{ij}[n] = a_{ij}[n][(\sigma s_j^2)/(w_i[n])]$.

Although other variables and the trajectory of the UAV could be decoupled, problem $SP3'$ is still nonconvex. The objective function and constraints (24c) and (24d) are neither convex or concave with respect to the UAV's flying trajectory $\mathbf{q}[n]$. Thus, it is a challenge to solve the nonconvex problem $SP3'$.

To tackle the problem, we adopt the successive convex optimization (SCA) method [17], [36] to approximate the nonconvex function to a convex function in an iterative manner, to obtain a local optimal solution for problem $SP3$. Thus, we define the following function as:

$$S_i[n]=\log_2\left(1+\frac{C_{ij}[n]}{h^2+\|\mathbf{q}[n]-\mathbf{u}_i\|^2}\right). \tag{25}$$

We can find that $S_i[n]$ is a convex function with respect to $\|\mathbf{q}[n]-\mathbf{u}_i\|^2$. Thus, it can be globally lower bounded by its first-order Tayler expansion with $\|\mathbf{q}[n]-\mathbf{u}_i\|^2$ at any point [35]. Define $L_i^k[n]=\|\mathbf{q}^k[n]-\mathbf{u}_i\|$ as the horizontal distance between the UAV and IoT device i at the kth iteration. Given the flying trajectory at the kth iteration, the lower bound of $S_i[n]$ can be obtained by

$$\hat{S}_i[n]=S_i^k[n]+\nabla S_i^k[n]\left(\|\mathbf{q}[n]-\mathbf{u}_i\|-L_i^k[n]\right) \tag{26}$$

where $S_i^k[n]$ and $\nabla S_i^k[n]$ are the $S_i[n]$ at the kth iteration and the first-order derivative of $S_i^k[n]$ with respect to $L_i^k[n]$, which are calculated by

$$S_i^k[n]=\log_2\left(1+\frac{C_{ij}[n]}{h^2+\left(L_i^k[n]\right)^2}\right) \tag{27}$$

$$\nabla S_i^k[n]=\frac{-C_{ij}[n]\ln 2}{\left(h^2+\left(L_i^k[n]\right)^2\right)\left(h^2+\left(L_i^k[n]\right)^2+C_{ij}[n]\right)}. \tag{28}$$

Similarly, in constraints (24c) and (24d), we adopt the SCA method to relax the constraints. As a result, problem $SP3'$ can be rewritten as the following approximate problem $SP3''$:

$$\mathbf{SP3}'': \max_{\mathbf{q}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \frac{\hat{S}_i[n]}{D_{ij}[n]}$$

$$\text{s.t.}\quad V_{\min}^2 \le \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{29a}$$

$$\mathbf{q}[1]=\mathbf{q}_0, \mathbf{q}[N+1]=\mathbf{q}_F \tag{29b}$$

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}\frac{p_i[n]}{D_{ij}[n]\hat{S}_i[n]} \le \left(E_i^{\max}-\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}}E_{ij}^r[n]\right) \quad \forall i \tag{29c}$$

$$\frac{1}{D_{ij}[n]\hat{S}_i[n]} \le \left(\tau-T_{ij}^r-T_{ij}^c\right) \quad \forall i,j,n. \tag{29d}$$

Since the constraint $\|\mathbf{v}[n]\|^2$ is a convex and differentiable function with $\|\mathbf{v}[n]\|$, for any given $\|\mathbf{v}[n]\|^k$ at the kth iteration, the first-order Taylor expansion can be obtained by

$$\|\mathbf{v}[n]\|^2 \ge \left(\|\mathbf{v}[n]\|^k\right)^2+2\|\mathbf{v}[n]\|^k\left(\|\mathbf{v}[n]\|-\|\mathbf{v}[n]\|^k\right) \quad \forall n \tag{30}$$

where $(\|\mathbf{v}[n]\|^k)^2+2\|\mathbf{v}[n]\|^k(\|\mathbf{v}[n]\|-\|\mathbf{v}[n]\|^k)$ is affine with respect to $\|\mathbf{v}[n]\|$. Thus, we recast problem $SP3''$ as

$$\mathbf{SP3}''': \max_{\mathbf{q}} \sum_{n\in\mathcal{N}}\sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \frac{\hat{S}_i[n]}{D_{ij}[n]}$$

$$\text{s.t.}\quad \|\mathbf{v}[n]\|^2 \le V_{\max}^2 \quad \forall n \tag{31a}$$

$$V_{\min}^2 \le \left(\|\mathbf{v}[n]\|^k\right)^2+2\|\mathbf{v}[n]\|^k \times\left(\|\mathbf{v}[n]\|-\|\mathbf{v}[n]\|^k\right) \quad \forall n \tag{31b}$$

$$\mathbf{q}[1]=\mathbf{q}_0, \mathbf{q}[N+1]=\mathbf{q}_F \tag{31c}$$

Algorithm 1 Joint Optimization of Model Decision, Resource Allocation, and Trajectory Control

Input: Initialized parameters: user coordinates $\mathbf{u}_i$, UAV's trajectory $\mathbf{q}^0[n]$, communication resource $\mathbf{w}^0[n]$ and computation resource allocation $\mathbf{f}^0[n]$, computation density $\mathbf{C}_i$,$\mathbf{C}_e$, model size $\mathbf{s}$, tolerance error ξ_1 and ξ_2.
Output: model decision $\mathbf{a}^{k,*}[n]$, computation resource allocation $\mathbf{f}^{k,*}[n]$, communication resource allocation $\mathbf{w}^{k,*}[n]$, trajectory of the UAV $\mathbf{q}^{k,*}[n]$.
1: Initialization, set iterative number $k = 1$;
2: **repeat**
3: Solve $SP1$ to obtain $\mathbf{a}^{k,*}[n]$ for given $\mathbf{f}^k[n]$, $\mathbf{w}^k[n]$, $\mathbf{q}^k[n]$;
4: Solve $SP2$ by using standard convex optimization techniques or CVX solver to obtain $\mathbf{f}^{k,*}[n]$, $\mathbf{w}^{k,*}[n]$ for given $\mathbf{q}^k[n]$, $\mathbf{a}^{k,*}[n]$;
5: Initialization, set iterative number $l = 1$.
6: **repeat**
7: Solve $SP3'''$ by using standard convex optimization techniques or CVX solver to obtain $\mathbf{q}^{k,*}[n]$ for given $\mathbf{a}^{k,*}[n]$, $\mathbf{f}^{k,*}[n]$, $\mathbf{w}^{k,*}[n]$;
8: **if** $\sum_{n\in N} \|\mathbf{q}_i^{l,*}[n] - \mathbf{q}_i^{l-1,*}[n]\| \le \xi_1$ **then**
9: $\mathbf{q}_i^{k,*}[n] = \mathbf{q}_i^{l,*}[n]$;
10: break;
11: **end if**
12: $l = l + 1$;
13: **until**
14: $k = k + 1$;
15: Calculate the objective function $f(\mathbf{a}^{k,*}[n], \mathbf{f}^{k,*}[n], \mathbf{w}^{k,*}[n], \mathbf{q}^{k,*}[n])$;
16: **until** $k \ge K$ or $|f(\mathbf{a}^{k,*}[n], \mathbf{f}^{k,*}[n], \mathbf{w}^{k,*}[n], \mathbf{q}^{k,*}[n]) - f(\mathbf{a}^{k-1,*}[n], \mathbf{f}^{k-1,*}[n], \mathbf{w}^{k-1,*}[n], \mathbf{q}^{k-1,*}[n])| \le \xi_2$.

$$\sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}} \frac{p_i}{D_{ij}[n]\hat{S}_i[n]} \le \left(E_i^{\max} - \sum_{n\in\mathcal{N}}\sum_{j\in\mathcal{J}} E_{ij}^r[n]\right) \quad \forall i \tag{31d}$$

$$\frac{1}{D_{ij}[n]\hat{S}_i[n]} \le \left(\tau - T_{ij}^r - T_{ij}^c\right) \quad \forall i, j, n. \tag{31e}$$

With convex objective function and constraints, problem $SP3'''$ is a convex optimization problem, which can be solved by standard convex optimization techniques, such as the CVX solver [37].

Therefore, the joint optimization of model decision, resource allocation, and trajectory control is detailed in Algorithm 1. According to Algorithm 1, the proposed iterative optimization algorithm solves the joint optimization of model decision, resource allocation, and trajectory control of the UAV. Algorithm 1 is comprised of three subproblems, each of them is convex or approximate convex and they can be solved in an alternating manner.

D. Convergence Analysis

To prove the feasibility of Algorithm 1, we discuss the convergence of the algorithm in this section. It is worth noting that for the trajectory control subproblem, we only optimally solve its approximate problem. In order to show the convergence properties of Algorithm 1, we have the following analysis.

Define $\Omega(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^k)$ as the objective value of the original problem P at the kth iteration. In step 3 of Algorithm 1, since $\mathbf{a}^k$ is suboptimal model decisions of problem $SP1$ with the fixed $\mathbf{f}^{k-1}$, $\mathbf{w}^{k-1}$, $\mathbf{q}^{k-1}$, we have

$$\Omega\left(\mathbf{a}^k, \mathbf{f}^{k-1}, \mathbf{w}^{k-1}, \mathbf{q}^{k-1}\right) \le \Omega\left(\mathbf{a}^{k-1}, \mathbf{f}^{k-1}, \mathbf{w}^{k-1}, \mathbf{q}^{k-1}\right). \tag{32}$$

In step 4 of Algorithm 1, for given $\mathbf{a}^k$ and $\mathbf{q}^{k-1}$, $\mathbf{f}^k$ and $\mathbf{w}^k$ are the optimal computation resource allocation and communication resource allocation of problem $SP2$, and we have

$$\Omega\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^{k-1}\right) \le \Omega\left(\mathbf{a}^k, \mathbf{f}^{k-1}, \mathbf{w}^{k-1}, \mathbf{q}^{k-1}\right). \tag{33}$$

In step 7 of Algorithm 1, since $\hat{S}_i[n]$ is the lower bounded by its the first-order Taylor expansion as shown in (26), the objective value of convex problem $SP3''$ is a lower bound of that of problem $SP3'$. Define the objective function of problem $SP3''$ as $\Theta(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^k)$. Thus, for given $\mathbf{a}^k$, $\mathbf{f}^k$, and $\mathbf{w}^k$, we have

$$\Theta\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^k\right) \ge \Theta\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^{k-1}\right). \tag{34}$$

As can be seen from the above inequality, the object value of problem $SP3$ is nonincreasing after each iteration. Although only an approximate optimization problem is solved for UAV's trajectory control, the objective value of the original problem P is still nonincreasing after each iteration. Thus, for given $\mathbf{a}^k$, $\mathbf{f}^k$, and $\mathbf{w}^k$, it follows that:

$$\Omega\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^k\right) \le \Omega\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^{k-1}\right). \tag{35}$$

Based on the above analysis, we obtain

$$\Omega\left(\mathbf{a}^k, \mathbf{f}^k, \mathbf{w}^k, \mathbf{q}^k\right) \le \Omega\left(\mathbf{a}^{k-1}, \mathbf{f}^{k-1}, \mathbf{w}^{k-1}, \mathbf{q}^{k-1}\right) \tag{36}$$

which indicates that the objective function of problem (18) is nonincreasing after each iteration in Algorithm 1.

Therefore, in each iteration, the objective value of problem P is monotonically nonincreasing and can converge to a local optimal solution.

E. Computation Complexity

According to [35], the computational complexity of an algorithm mostly depends on the number of decision variables. In this article, P is divided into three subproblems and its solution is found by iteratively solving $SP1$, $SP2$, and $SP3$. Thus, the computation complexity of P is equal to the total computational complexity of $SP1$, $SP2$, and $SP3'''$ multiplied by the total number of iterations.

$SP1$ is a standard integer linear programming problem, which can be solved by the binary cut-and-branch method [34] with the complexity of $O(n_1 \log n_1)$, where n_1 is the number of variables. $SP1$ includes (IJN) decision variables and its computation complexity of $SP1$ can be denoted as $O((IJN)\log(IJN))$. When the interior point method [35] is adopted to solve $SP2$ and $SP3'''$, the computational complexity of the optimal solution for a convex problem is given as $O(n_2^{3.5}\log(1/\varepsilon))$, where n_2 is the number of variables and ε is the given solution accuracy. $SP2$ includes

TABLE I
SIMULATION PARAMETERS

Sym.	Parameters	Value
W	Communication bandwidth	20 MHz
$V_{\max}$	UAV's maximum speed	100 m/s
$V_{\min}$	UAV's minimum speed	20 m/s
β_0	Channel power gain	-50 dB
p_i	Transmit power of device i	0.3 W
N_0	Noise power density	-174 dBm/Hz
C_i	Local computation density	[1000,2000] CPU cycles/bit
C_e	Edge computation density	1000 CPU cycles/bit
$f_i^{\max}$	Computing capacity of device i	[1,2] GHz
$f_e^{\max}$	UAV's Computing capacity	14 GHz
$E_i^{\max}$	Device residual energy	10 J
k	Effective switched capability	10^{-28}

$(2IN)$ decision variables and the corresponding complexity is $O((2IN)^{3.5}\log(1/\varepsilon))$. There are $(2(N-1))$ decision variables in $SP3'''$, and its complexity is $O((2N)^{3.5}\log(1/\varepsilon))$. Suppose that the number of iterations in the outer and inner loops is K and L, respectively, the total computation complexity for the proposed Algorithm 1 can be calculated as $O(K((IJN)\log(IJN)+((2IN)^{3.5}+L(2N)^{3.5})\log(1/\varepsilon)))$. It could be found that the proposed algorithm runs in a polynomial time and is of high complexity, making its implementation challenged especially when the scale of the network is extremely large. However, this issue could be alleviated by setting larger threshold parameters ξ_1 and ξ_2 to reduce the number of iterations properly while resulting in the computation accuracy reduction. Therefore, there exists a tradeoff between the computation efficiency and accuracy of the proposed algorithm.

V. SIMULATION RESULTS

We consider a UAV-enabled MEC system, where six IoT devices are randomly distributed within an area of 2000 m $\times$ 2000 m. The entering horizontal position and the left position of the UAV are set as [0, 0] and [2000, 2000], respectively. The flying trajectory of the UAV is always sampled every time slot. We assume that the UAV flies at a fixed altitude $h = 2000$ m. There are three DNN models deployed on the UAV, and corresponding to each DNN model, the input sizes are set as 100×100, 300×300, and 600×600 pixels, respectively. The data size for each pixel is set as $\sigma = 24$ bits. The larger the input data of the DNN model, the higher the model accuracy [10]. For simplicity, in our simulation, we assume that the model accuracy of the selected model is undesired when a DNN model with input size less than 300×300 pixels is selected. We also assume that the total number of undesired model decisions accounts for 1/3 or 1/4 of the number of model decisions, i.e., the learning accuracy level of the system is set as $\chi = 1/4$ or $\chi = 1/3$. According to (17), the smaller the χ, the higher the overall learning accuracy level of the system and the higher user service satisfaction. The remaining parameters are summarized in Table I.

A. Convergence

To ensure the feasibility of the proposed iterative algorithm, we first need to verify its convergence properties. Fig. 3 shows

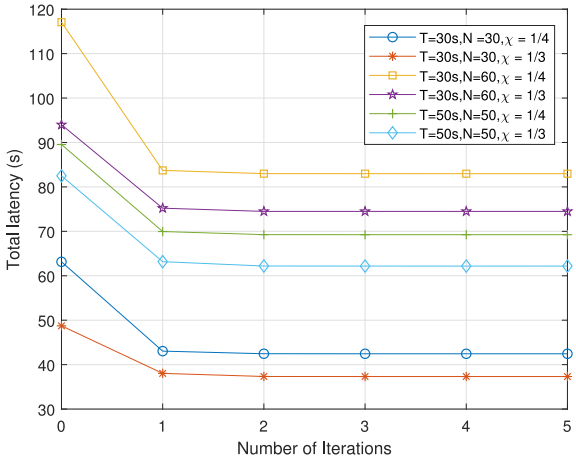

Fig. 3. Convergence behavior of the proposed algorithm under different parameter settings.

the convergence of the proposed algorithm under different parameter settings. The flying period T is set as 30 and 50 s, and the number of time periods N is set to 30 and 50, respectively, and the length of each time slot τ is 1 s. When the flying period T is set as 30 s and the number of time slots is 60, the length of each time slot τ is 0.5 s. Before the iteration starts, we run Algorithm 1 based on initial communication resource and computation resource allocation and initial trajectory. It can be observed that the objective function values can rapidly converge to a constant less than five iterations ($\xi_1, \xi_2 = 0.01$) for different learning accuracy constraints and different time slot numbers and lengths, which shows the effectiveness of the proposed algorithm. Intuitively, the objective function values increase as the accuracy indicator values reduce, which is because IoT devices need to select larger size DNN models to guarantee the learning accuracy requirements. In our simulation, each task is scheduled for task processing on a slot-by-slot basis. Decreasing the time slot length τ under the given the time period T or increasing the time period T under the given the time slot length τ can increase the number of time slots, increasing in the objective function values.

B. Trajectory Comparisons

Fig. 4 illustrates different UAV trajectories under different schemes and time slot lengths. In the fixed straight UAV trajectory scenario, the UAV flies straight with the constant flying speed from the entering position to the left position with a straight trajectory. In the fixed arc trajectory scheme, the fly trajectory of the UAV is an arc, in which the two ends of the arc are the entering position and the left position, respectively. The optimized trajectory is obtained by using our proposed Algorithm 1.

Fig. 4(a) shows two optimized trajectories at the different flying periods with time slot length equal to 1 s. For the optimized UAV trajectories, the UAV always flies from the entering position with the maximum possible speed in the first 13 time slots and flies to the left position in the last 13 time slots. After the first 13 time slots, the UAV flies close around a specific domain with the maximum possible duration. Here,

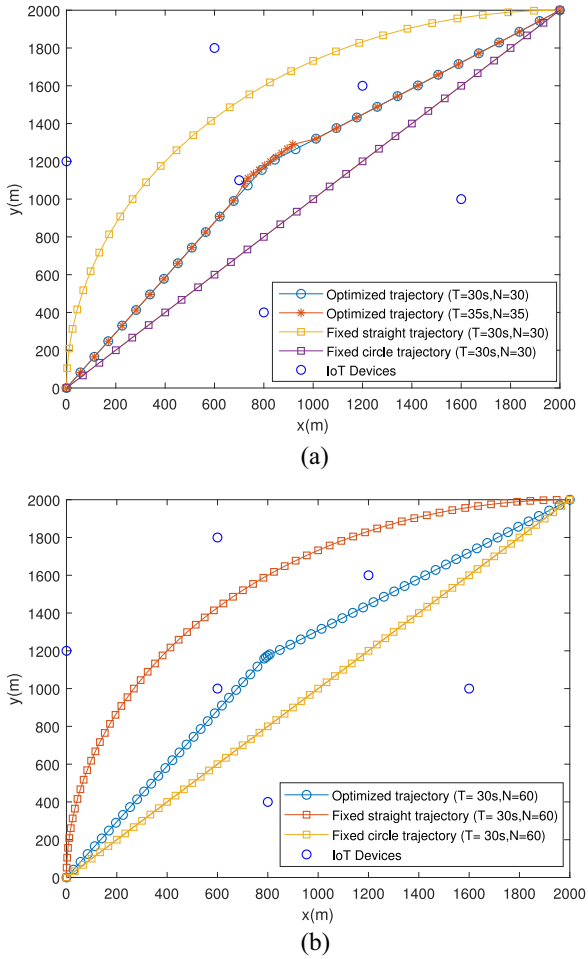

Fig. 4. Different UAV trajectories under different schemes. (a) Different UAV trajectories with $\tau = (T/N) = 1$ s. (b) Different UAV trajectories with $\tau = (T/N) = 0.5$ s.

we name this domain as the optimal way-point domain. It also can be seen that the value of T significantly affects the maximum possible duration of the UAV to fly close around the particular domain. The larger the flying period T, the longer time for the UAV to fly close around the specific domain. Specifically, when $T = 35$ s, the UAV has nine time slots flying around the fixed domain.

Fig. 4(b) shows UAV trajectories with the time slot length of 0.5 s in the same flying period. The only difference between the two fixed trajectory scenarios in Fig. 4(a) and the two fixed trajectory scenarios in Fig. 4(b) is the movement distance of the UAV in each time slot. For the optimized UAV trajectory, the UAV always flies from the entering position with the maximum possible speed in the first 26 time slots and flies to the left position in the last 26 time slots. After the first 26 time slots, the UAV flies close around the optimal way-point domain with the maximum possible duration. In comparison, when $T = 30$ s, the optimized trajectory in Fig. 4(b) is more refined than that in Fig. 4(a).

C. Performance Comparisons

To give a more detailed illustration, Fig. 5 illustrates the average service latency with and without the trajectory optimization at different periods, time slot lengths, and maximum UAV speeds. In contrast, the optimized UAV trajectories are closer to all IoT devices than the nonoptimized trajectories (i.e., the fixed arc trajectory and fixed straight trajectory), reducing the average service latency in each time slot due to better channel quality. Besides, the average service reduces as the maximum UAV speed increases. This is because the faster the UAV speed, the shorter the time it takes for the UAV to reach and leave the optimal way-point domain, and the longer the duration for the UAV to flying around the optimal way-point domain.

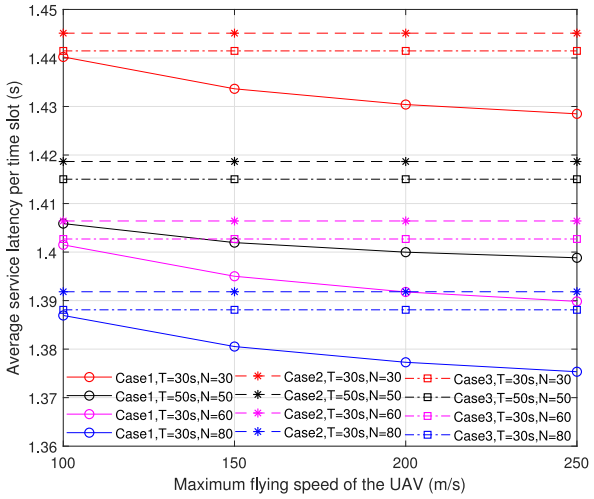

Fig. 5. Average service latency per time slot versus different periods, time slot lengths, and maximum flying speeds (Case1: optimized trajectory, Case2: fixed arc trajectory, and Case3: fixed straight trajectory).

In Fig. 5, the number of time slots of $T = 30$ and $T = 50$ is set as 30 and 50, and the length of each time slot is set as $\tau = 1$ s, respectively. It can be seen that the average service latency in each time slot would reduce as the number of time slots increases, which is due to the fact that the UAV would have more freedom to fly closer to its serving devices for better channel conditions. Specifically, when the period T increases, the UAV would be located in the optimal way-point domain with a longer duration and would service all ground IoT devices in better channel conditions, leading to decreased latency performance. Given the flying period $T = 30$ s, the number of time slots is set to 60 and 80, and the length of each time slot is 0.5 and 0.375 s, respectively. It can be seen that in the same flying period $T = 30$ s, the average service decreases with the increase of the number of time slots. This is because the UAV would fly close around the optimal way-point domain with a more refined trajectory and would service all ground IoT devices in better channel conditions, improving the service latency performance.

Figs. 6 and 7 compare the total latency of different schemes with respect to the maximal available computation and communication resources and accuracy requirements. In the previous works [8], [38], the equal resource allocation policy is considered, which allocates equal communication resource or computation resource to multiple devices. In our simulation, we use the equal resource allocation policy as a baseline for performance comparison. The optimal CPU

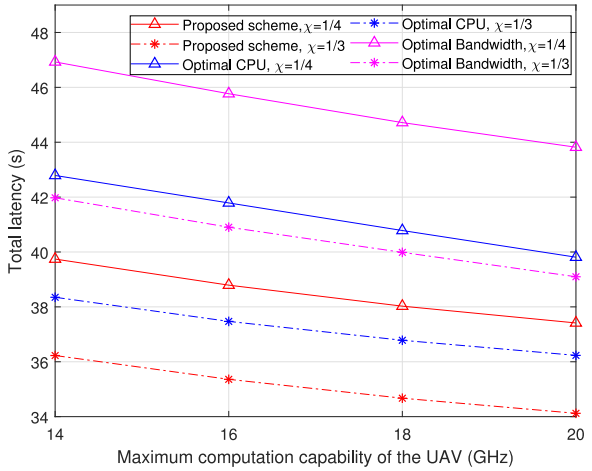

Fig. 6. Total service latency of different schemes versus maximal available computation resources of the UAV and accuracy requirements ($T = 30$ s and $N = 30$).

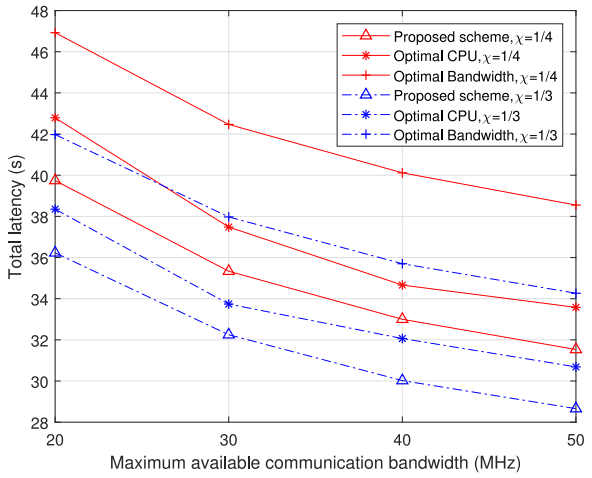

Fig. 7. Total service latency of different schemes versus maximal available communication resources and accuracy requirements ($T = 30$ s and $N = 30$).

scheme indicates that communication resources allocated to each IoT device are equal while computation resources are allocated optimally. The optimal bandwidth scheme denotes that computation resources allocated to each IoT device are equal while communication bandwidth resources are allocated optimally. Our proposed scheme uses Algorithm 1 to optimize computation and communication resources. Compared with the optimal CPU scheme and bandwidth scheme, the proposed scheme has the best system performance, which shows the efficiency of the proposed Algorithm 1 for service latency minimization.

Intuitively, according to Fig. 6, when the maximal computation capacity of the UAV increases, IoT devices can reduce more computing latency while guaranteeing their accuracy constraints. From Fig. 7, it can also be observed that the total service latency for all tasks can be further reduced if the available bandwidth increases, which is because the increased bandwidth for each device to achieve higher data transmission rate according to constraint (10). According to Figs. 6 and 7, the total service latency increases as the learning accuracy of

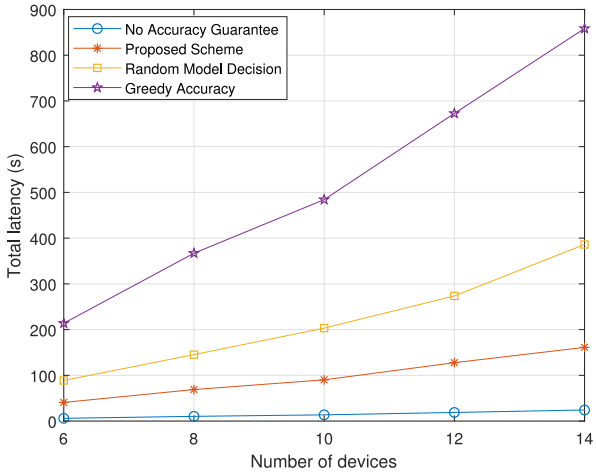

Fig. 8. Total service latency comparisons of different schemes under different device numbers ($T = 30$ s, $N = 30$, and $\chi = 1/4$).

the system, which is due to the fact that the larger models can increase the accuracy at the cost of longer processing latency. We can also observe that compared with the large accuracy indicator, the small accuracy indicator requires more computation and communication resources to achieve the same system performance. For example, in Fig. 7, when the service latency is about 32 s, for the accuracy level indicator value $\chi = 1/3$, the maximal available communication bandwidth is 30 MHz. For the accuracy indicator $\chi = 1/4$, the maximal available communication bandwidth needs to be increased to 45 MHz.

Finally, we compare the system performance of the proposed scheme with that of the following three benchmark schemes, i.e.: 1) *No Accuracy Guarantee:* it aims to minimize the total service latency of all devices while guaranteeing their energy consumption and latency constraints, regardless of the learning accuracy requirements; 2) *Random Model Decisions:* it aims to minimize the total service latency of all devices, regardless of the learning accuracy requirements, latency, and devices' energy consumption constraints; and 3) *Greedy Accuracy:* all devices select the largest DNN model all the time to maximize the learning accuracy of the system, while the energy and latency constraints are ignored.

According to Fig. 8, the service latency of different schemes increases significantly as the number of devices increases due to serious computation and communication resource contentions. Our proposed scheme always selects the larger DNN models to ensure the accuracy performance and achieve lower service latency. For the *No Accuracy Guarantee* scheme, it selects the smallest DNN model all the time to achieve the lowest service latency while sacrificing the accuracy performance, which provides an achievable lower bound of latency performance. *Greedy Accuracy* scheme selects the largest DNN model all the time to achieve the highest desirable system accuracy at a cost of long latency, which presents an achievable upper bound of latency performance. In contrast, the service latency of our scheme is significantly lower than that of *Random Model Decisions* scheme and *Greedy Accuracy* scheme and higher than that of *No Accuracy Guarantee* scheme.

VI. Conclusion and Future Work

In this article, we investigated the latency minimization problem in the air–ground-integrated wireless networks by jointly optimizing the model decision, computation and communication resource allocation, and UAV trajectory control. We have proposed an iterative optimization algorithm to tackle the problem and proved that the algorithm has good convergence. Simulation results show that by optimized the model decision, resource allocation, and UAV trajectory, the proposed algorithm could provide outstanding service performance guarantees under energy consumption, latency, and accuracy requirement constraints.

Our investigation also reveals that independently processing a task with high accuracy requirements on the IoT devices or the UAV is limited by their computing capability and energy consumption. In the future, to further improve the performance of the air–ground-integrated MEC system, we will focus on investigating UAV swarm collaboration, including how to dynamically form UAV swarm to provide continuous coverage of a given area and on-demand collaboration, and how to reasonably split a large-scale AI model into multiple independent model segments and then deploy them to UAV swarm for collaborative computing.

References

[1] S. S. Mousavi, M. Schukat, and E. Howley, "Deep reinforcement learning: An overview," in *Proc. SAI Intell. Syst. Conf.*, 2016, pp. 426–440.

[2] P. Jain, J. Manweiler, and R. R. Choudhury, "Low bandwidth offload for mobile AR," in *Proc. 12th Int. Conf. Emerg. Netw. Exp. Technol.*, 2016, pp. 237–251.

[3] T. Y.-H. Chen, L. Ravindranath, S. Deng, P. Bahl, and H. Balakrishnan, "GLIMPSE: Continuous, real-time object recognition on mobile devices," in *Proc. 13th ACM Conf. Embedded Netw. Sens. Syst.*, 2015, pp. 155–168.

[4] P. Jain, J. Manweiler, and R. R. Choudhury, "OverLay: Practical mobile augmented reality," in *Proc. 13th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2015, pp. 331–344.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[6] 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.

[7] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.

[8] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[9] X. Ran, H. Chen, Z. Liu, and J. Chen, "Delivering deep learning to mobile devices via offloading," in *Proc. Workshop Virtual Reality Augmented Reality Netw.*, 2017, pp. 42–47.

[10] Q. Liu, S. Huang, J. Opadere, and T. Han, "An edge network orchestrator for mobile augmented reality," in *Proc. IEEE INFOCOM*, Honolulu, HI, USA, 2018, pp. 756–764.

[11] X. Ran, H. Chen, X. Zhu, Z. Liu, and J. Chen, "DeepDecision: A mobile deep learning framework for edge video analytics," in *Proc. IEEE INFOCOM*, Honolulu, HI, USA, 2018, pp. 1421–1429.

[12] Q. Liu and T. Han, "DARE: Dynamic adaptive mobile augmented reality with edge computing," in *Proc. IEEE 26th ICNP*, Cambridge, U.K., 2018, pp. 1–11.

[13] T. Tan and G. Cao, "FastVA: Deep learning video analytics through edge processing and NPU in mobile," in *Proc. IEEE INFOCOM*, Toronto, ON, Canada, 2020, pp. 1947–1956.

[14] J. Ahn, J. Lee, D. Niyato, and H.-S. Park, "Novel QoS-guaranteed orchestration scheme for energy-efficient mobile augmented reality applications in multi-access edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13631–13645, Nov. 2020.

[15] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, "Completion time and energy optimization in the UAV-enabled mobile-edge computing system," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7808–7822, Aug. 2020.

[16] X. Hu, K.-K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4738–4752, Oct. 2019.

[17] J. Zhang *et al.*, "Computation-efficient offloading and trajectory scheduling for multi-UAV assisted mobile edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2114–2125, Feb. 2020.

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

[19] Y. Liu, K. Xiong, Q. Ni, P. Fan, and K. B. Letaief, "UAV-assisted wireless powered cooperative mobile edge computing: Joint offloading, CPU control, and trajectory optimization," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 2777–2790, Apr. 2020.

[20] L. Ji and S. Guo, "Energy-efficient cooperative resource allocation in wireless powered mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4744–4754, Jun. 2019.

[21] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan./Feb. 2018.

[22] T. Bai, J. Wang, Y. Ren, and L. Hanzo, "Energy-efficient computation offloading for secure UAV-edge-computing systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 6074–6087, Jun. 2019.

[23] L. Yang, H. Yao, J. Wang, C. Jiang, A. Benslimane, and Y. Liu, "Multi-UAV-enabled load-balance mobile-edge computing for IoT networks," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6898–6908, Aug. 2020.

[24] F. Jiang, K. Wang, L. Dong, C. Pan, W. Xu, and K. Yang, "Deep-learning-based joint resource scheduling algorithms for hybrid MEC networks," *IEEE Internet Things J.*, vol. 7, no. 7, p. 6252, Jul. 2020.

[25] J. Zhang *et al.*, "Stochastic computation offloading and trajectory scheduling for UAV-assisted mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3688–3699, Apr. 2019.

[26] H. Guo and J. Liu, "UAV-enhanced intelligent offloading for Internet of Things at the edge," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2737–2746, Apr. 2020.

[27] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge computing networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4576–4589, Sep. 2019.

[28] Q. Zhang, J. Chen, L. Ji, Z. Feng, Z. Han, and Z. Chen, "Response delay optimization in mobile edge computing enabled UAV swarm," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3280–3295, Mar. 2020.

[29] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," ETSI, Sophia Antipolis, France, Rep., vol. 11, no. 11, 2015, pp. 1–16. [Online]. Available: http://www.etsi.org/images/files/ ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf

[30] Y. He, J. Ren, G. Yu, and Y. Cai, "Optimizing the learning accuracy in mobile augmented reality systems with CNN," in *Proc. ICC*, Dublin, Ireland, 2020, pp. 1–6.

[31] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Sep. 2018.

[32] W. Sun, J. Liu, and Y. Yue, "AI-enhanced offloading in edge computing: When machine learning meets industrial IoT," *IEEE Netw.*, vol. 33, no. 5, pp. 68–74, Sep./Oct. 2019.

[33] P. M. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Comput.*, vol. C-26, no. 9, pp. 917–922, Sep. 1977.

[34] A. Lodi, "Mixed integer programming computation," in *50 Years of Integer Programming 1958-2008*. Heidelberg, Germany: Springer, 2010, pp. 619–645.

[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[36] G. Scutari, F. Facchinei, and L. Lampariello, "Parallel and distributed methods for constrained nonconvex optimization—Part I: Theory," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 1929–1944, Apr. 2017.

[37] M. Grant and S. Boyd, *CVX: MATLAB Software for Disciplined Convex Programming, Version 2.1*, CVX Res., Inc., Austin, TX, USA, 2014.

[38] M. Alsenwi, Y. K. Tun, S. R. Pandey, N. N. Ei, and C. S. Hong, "UAV-assisted multi-access edge computing system: An energy-efficient resource management framework," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Barcelona, Spain, 2020, pp. 214–219.

Cailian Deng received the B.E. degree in communication engineering from Southwest Jiaotong University, Emei, China, in 2017. She is currently pursuing the Ph.D. degree with the Key Laboratory of Information Coding and Transmission, School of Information Science and Technology, Southwest Jiaotong University, Chengdu, China.

Her research interests include unmanned aerial vehicle communications, network resource management, edge intelligence, mobile-edge computing, Internet of Things.

Xuming Fang (Senior Member, IEEE) received the B.E. degree in electrical engineering, the M.E. degree in computer engineering, and the Ph.D. degree in communication engineering from Southwest Jiaotong University, Chengdu, China, in 1984, 1989, and 1999, respectively.

He was a Faculty Member with the Department of Electrical Engineering, Tongji University, Shanghai, China, in September 1984. He then joined the School of Information Science and Technology, Southwest Jiaotong University, where he has been a Professor since 2001, and the Chair of the Department of Communication Engineering since 2006. He held visiting positions with the Institute of Railway Technology, Technical University of Berlin, Berlin, Germany, in 1998 and 1999, and the Center for Advanced Telecommunication Systems and Services, University of Texas at Dallas, Richardson, TX, USA, in 2000 and 2001. He has, to his credit, around 200 high-quality research papers in journals and conference publications. He has authored or coauthored five books or textbooks. His research interests include wireless broadband access control, radio resource management, multihop relay networks, and broadband wireless access for high-speed railway.

Dr. Fang is the Chair of the IEEE Vehicular Technology Society of Chengdu Chapter and an Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

Xianbin Wang (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2001.

He is a Professor and a Tier-1 Canada Research Chair with Western University, London, ON, Canada. Prior to joining Western University, he was with Communications Research Centre Canada (CRC), Ottawa, ON, Canada, as a Research Scientist/Senior Research Scientist from July 2002 to December 2007. From January 2001 to July 2002, he was a System Designer with STMicroelectronics, Geneva, Switzerland. His current research interests include 5G/6G technologies, Internet of Things, communications security, machine learning, and intelligent communications. He has over 500 highly cited journal and conference papers, in addition to 30 granted and pending patents and several standard contributions.

Dr. Wang has received many awards and recognitions, including the Canada Research Chair, the CRC President's Excellence Award, the Canadian Federal Government Public Service Award, the Ontario Early Researcher Award, and six IEEE Best Paper Awards. He currently serves/has served as an editor-in-chief, associate editor-in-chief, and editor/associate editor for over ten journals. He was involved in many IEEE conferences, including GLOBECOM, ICC, VTC, PIMRC, WCNC, CCECE, and CWIT, in different roles, such as the General Chair, the Symposium Chair, the Tutorial Instructor, the Track Chair, the Session Chair, the TPC Co-Chair, and a Keynote Speaker. He has been nominated as an IEEE Distinguished Lecturer several times during the last ten years. He is currently serving as the Chair of IEEE London Section and the ComSoc Signal Processing and Computing for Communications Technical Committee. He is a Fellow of the Canadian Academy of Engineering and the Engineering Institute of Canada and an IEEE Distinguished Lecturer.