Two-Stage Deep Energy Optimization in IRS-Assisted UAV-Based Edge Computing Systems

Jianqiu Wu, Zhongyi Yu, Jianxiong Guo, *Member, IEEE*, Zhiqing Tang, *Member, IEEE*, Tian Wang, *Senior Member, IEEE*, and Weijia Jia, *Fellow, IEEE*

***Abstract*—Integrating wireless-powered Mobile Edge Computing (MEC) with Unmanned Aerial Vehicles (UAVs) leverages computation offloading services for mobile devices, significantly enhancing the mobility and control of MEC networks. However, current research has not focused on customizing system designs for Terahertz (THz) communication networks. When dealing with THz communication, one must account for blockage vulnerability due to severe THz wave propagation attenuation and insufficient diffraction. The Intelligent Reflecting Surface (IRS) can effectively address these limitations in the model, enhancing spectrum efficiency and coverage capabilities while reducing blockage vulnerability in THz networks. In this paper, we introduce an upgraded MEC system that integrates IRS and UAVs into THz communication networks, focusing on a binary offloading policy for studying the computation offloading problem. Our primary objective is to optimize the energy consumption of both UAVs and User Electronic Devices, alongside refining the phase shift of the IRS reflector. The problem is a Mixed Integer Non-Linear Programming problem known as NP-hard. To tackle this challenge, we propose a two-stage deep learning-based optimization framework named Iterative Order-Preserving Policy Optimization (IOPO). Unlike exhaustive search methods, IOPO continually updates offloading decisions through an order-preserving quantization method, thereby accelerating convergence and reducing computational complexity, especially when handling complex problems with extensive solution spaces. The numerical results demonstrate that the proposed algorithm significantly improves energy efficiency and achieves near-optimal performance compared to benchmark methods.**

***Index Terms*—Mobile edge computing, deep learning, unmanned aerial vehicles, intelligent reflective surface, terahertz communications.**

I. Introduction

A MOBILE Edge Computing (MEC) network enhanced by the inclusion of Unmanned Aerial Vehicles (UAVs) stands as a fitting solution for ensuring reliable network services at target locations, leveraging their mobility and precise deployment capabilities [1], [2], [3], [4], [5], [6], [7], [8]. Yet, limited research of the present researchers have considered constructing this model under Terahertz (THz) communication, which can offer abundant bandwidth resources, and this is crucial in an era in which communicating data is growing with an explosive speed [9].

However, the high propagation loss associated with THz transmissions due to electromagnetic signal travel through the medium and water vapor's absorptive properties in the atmosphere [10] poses a significant challenge. Fortunately, the proposed intelligent reflective surface (IRS) can reconfigure wireless propagation channels by adjusting phase shifts of reflecting elements. This innovation significantly enhances communication, especially in UAV-supported THz communication systems [11], [12]. Recent studies [13], [14], [15], [16] have demonstrated that IRS can be a significant component in UAV-assisted MEC systems. Furthermore, additional research [17], [18], [19], [20], [21], [22], [23], [24] has shown that IRS plays a crucial role in augmenting wireless communication performance and increasing network transmission speed.

Despite this, task offloading allocations in an IRS-assisted multi-UAV MEC system operating within the THz network remain underexplored, with minimal research in this area. The continuous fluctuations in channel gain, user and UAV positioning, and phase shifts perpetually impact transmission speed. With constrained resources allocated to UAVs, an imperative arises for an energy-efficient offloading plan. Optimizing both task offloading decisions and phase shifts becomes vital. However, this optimization problem, referred to as a Mixed-Integer Non-Linear Programming (MINLP) problem, is complex and NP-hard.

Received 11 January 2024; revised 13 August 2024; accepted 12 September 2024. Date of publication 17 September 2024; date of current version 4 December 2024. This work was supported in part by the National Key R&D Program of China under Grant 2022YFE0201400, in part by the National Natural Science Foundation of China (NSFC) under Grant 62202055, in part by the Start-up Fund from Beijing Normal University under Grant 310432104, in part by the Start-up Fund from BNU-HKBU United International College under Grant UICR0700018-22, in part by the Project of Young Innovative Talents of Guangdong Education Department under Grant 2022KQNCX102, and in part by the Interdisciplinary Intelligence SuperComputer Center, Beijing Normal University (Zhuhai). Recommended for acceptance by S. Balasubramaniam. *(Corresponding author: Jianxiong Guo.)*

Jianqiu Wu and Zhongyi Yu are with the Guangdong Key Lab of AI and Multi-Modal Data Processing, Department of Computer Science, BNU-HKBU United International College, Zhuhai 519087, China (e-mail: jqwuhelen@qq.com; zhongyicst@gmail.com).

Jianxiong Guo and Weijia Jia are with the Advanced Institute of Natural Sciences, Beijing Normal University, Zhuhai 519087, China, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Zhuhai 519087, China (e-mail: jianxiongguo@bnu.edu.cn; jiawj@bnu.edu.cn).

Zhiqing Tang and Tian Wang are with the Advanced Institute of Natural Sciences, Beijing Normal University, Zhuhai 519087, China (e-mail: zhiqingtang@bnu.edu.cn; cs_tianwang@163.com).

Digital Object Identifier 10.1109/TMC.2024.3461719

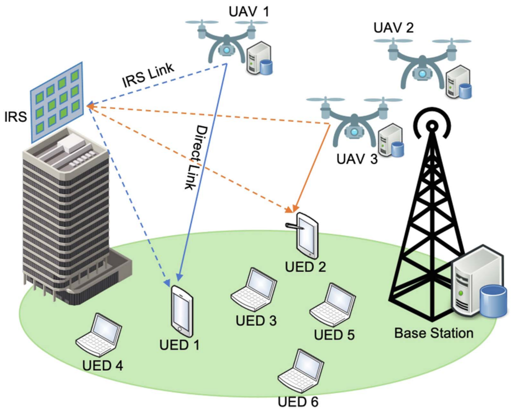


Fig. 1. The proposed IRS-assistant UAV-MEC system. User data can be directly transmitted from UEDs to UAVs or be redirected to UAVs from IRS.

Recent research has introduced optimization methods based on reinforcement learning to address this challenge. While some methods like discretizing the action space in Deep Q Network (DQN) encounter issues related to the curse of dimensionality, others like the Deep Deterministic Policy Gradient algorithm (DDPG) overcome this challenge by using neural networks to map system states to actions [25]. However, these methods adopt a single-stage approach, generating offloading decisions and optimized phases simultaneously, resulting in suboptimal solutions and requiring further training iterations.

Our proposed system (as illustrated in Fig. 1) comprises multiple User Equipment Devices (UEDs), a fleet of UAVs, and an IRS responsible for enhancing UAVs' channel capacity and improving MEC network transmission reliability. To address these challenges, we propose the *Iterative Order-preserving Policy Optimization (IOPO)* framework, a novel two-stage deep learning framework. IOPO effectively determines energy-efficient binary task offloading allocations for the MEC system and optimizes the phase shift configurations of the IRS. Compared to one-stage methods attempting to derive two variables from a joint probability space, a two-stage method first obtains a definite offloading decision and then identifies an optimal phase shift. This approach allows us to effectively approximate the theoretically optimal solution. The experiments reveal IOPO's capability to generate optimal task offloading strategies while meeting defined constraints, achieving superior optimization outcomes. Moreover, with an equal number of training iterations, IOPO produces solutions that are closer to the optimal one. Our source code can be found at https://github.com/UIC-JQ/IOPO. The contributions of this paper can be summarized as follows.

- We present a novel MEC system tailored for operation on the THz communication network. The proposed MEC system is equipped with an IRS, which is crucial in enhancing communication performance within the network. Additionally, the system is designed to accommodate multiple UAVs and users.
- In order to streamline the optimization process and improve the efficiency of the MEC system, we propose a deep learning framework named IOPO. IOPO is designed to jointly optimize offloading decisions of the multi-user multi-uav system and the phase shift of the IRS. As a result, IOPO eliminates the need to solve complex MINLP problems, which can be computationally demanding and time-consuming.
- To facilitate the generation of high-quality offloading decisions, we equip IOPO with a novel policy exploration unit called *Order-Preserving Policy Optimization (OPPO)*, specifically designed to search for improved offloading decisions. Experimental results demonstrate the effectiveness of OPPO in discovering improved offloading decisions, even in scenarios with a vast solution space. Furthermore, results show that the integration of OPPO facilitates the convergence of IOPO towards optimal offloading decisions.
- Simulation results demonstrate IOPO's impressive capability in significantly reducing energy consumption, surpassing benchmark schemes, including a strong baseline DDPG [26]. The energy cost is reduced by up to 32.8% when there are 3 UAVs and 15 users.

The rest of the paper is organized as follows. Section II provides a comprehensive review of previous studies. In Section III, we introduce the proposed MEC system model and formulate the data communication within the THz network. Section IV formulates the optimization problem aimed at minimizing the energy. The design of the proposed IOPO framework is described in Section V. Experimental settings are presented in Section VI, followed by a thorough analysis of the results in Section VII. Finally, Section VIII concludes the paper by summarizing the key findings.

II. Related Work

The integration of IRS in THz communication has been extensively studied in recent works [19], [20], [21], [22], [23]. In [19], [20], the IRS is employed to maximize the sum-rate performance of THz communications. The studies conducted in [21], [22] focus on utilizing the IRS to maintain reliable THz transmission. [23] introduces a comprehensive optimization framework that jointly optimizes the UAV trajectory, IRS phase adjustments, THz sub-band allocation, and power control. Additionally, recent works [13], [14], [16] have explored the integration of UAVs and IRS within MEC systems. These studies emphasize the importance of expanding UAV capabilities and utilizing IRS to enhance system performance.

To generate offloading allocations for MEC systems, several studies employ machine learning algorithms. [27], [28] applies deep reinforcement learning techniques to determine optimal task offloading strategies in scenarios involving single or multiple access points (APs). [29] considers factors such as channel state information, queue state information, and energy queue state and introduces a deep Q-learning network to generate offloading decisions that minimize task execution costs. Similarly, in [30], a deep Q-learning network is proposed to maximize the computational performance of energy-harvesting MEC networks. [31] proposes a deep learning based optimization approach to minimize the system energy consumption while optimizing the positions of ground vehicles and unmanned aerial vehicles along with the resource allocation in a hybrid mobile edge computing platform. Furthermore, [32] focuses on optimizing the phase shift of IRS, UAV computing resources, and sub-band allocation in a single UAV scenario. [15] introduces a dueling double deep Q networks (D3QN)-DDPG network for minimize transmission and computing delays while ensuring secure transmission. These works demonstrate the effectiveness of machine learning models in producing high-quality offloading strategies for MEC systems.

While progress has been made in existing literature, the task offloading in an IRS-assisted multi-UAV MEC system operating within the THz network remains unexplored. Specifically, [19], [20], [21], [22] primarily focuses on enhancing THz network communication with IRS. However, they do not adequately address the modeling of MEC systems within the context of THz networks. Moreover, [23], [24] introduce the utilization of IRS to improve the efficiency of MEC systems, but their systems do not tackle the optimization problems associated with task offloading. In addition, recent works [13], [14], [15], [16], [27], [28], [29], [30], [31] have utilized convex optimization techniques and deep learning models to generate offloading decisions, but these approaches are tailored to the 5G network context, failing to account for the unique characteristics of THz communication networks. Lastly, [32] investigates the allocation of network recourses and computational resources in the context of THz networks, taking into account the integration of IRS and UAVs. However, the studied system does not address the MEC task offloading problem and only involves a single UAV, thereby failing to model the complexities that arise in systems with multiple UAVs.

III. System Model

In this section, we first provide a detailed description of the components comprising the proposed MEC system and demonstrate how the MEC system operates in general. Following this, Section III-B formulates the communication and data transmission between UAVs and users within the MEC system. Lastly, Section III-C introduces the steps for computing the total energy consumed in the MEC system. The frequently used notations are shown in Table I.

A. The Proposed MEC System

Fig. 1 presents the proposed multi-UAV multi-user MEC system designed for THz communication networks. The system comprises a single IRS, U users denoted as $\mathcal{U} = \{1, 2, \ldots, U\}$, and M UAVs denoted as $\mathcal{M} = \{1, 2, \ldots, M\}$. Each user is equipped with a User Electronic Device (UED), which serves as a local computing server. Each UAV provides full-duplex communication services to users within a specific area and is equipped with an MEC server responsible for processing the tasks uploaded by users and transmitting the results through downlink transmission. We assume that the MEC server mounted on the UAV is the UAV itself. Additionally, the computation result to be downloaded to the WD is much shorter than the data offloaded to the edge server and can be neglected. The scarcity of previous studies indicates the feasibility of this approach [27], [33], [34], [35]. Compared with the UEDs, the MEC servers are designed with higher computational capacity. This empowers users to make decisions regarding task offloading, choosing between offloading their computational tasks to one of the M UAVs or executing them locally on their UEDs. Consequently, the task allocation for the entire MEC system can be represented by a $U \times (M + 1)$ matrix, where $M + 1$ signifies that users choose from M UAVs and their local UEDs. An IRS comprising K reflecting elements is set to assist the system. By manipulating the phase shifts of these reflecting elements, the IRS can reconfigure wireless propagation channels in a highly efficient manner. This reconfiguration leads to significant improvements in both the overall propagation environment and the data transmission speed of the system.

TABLE I
THE FREQUENTLY USED NOTATIONS IN THIS PAPER

Notation	Description
U	The number of users
M	The number of UAVs
T	The length of a time slot
$\boldsymbol{\beta}(n)$	The allocation matrix of users and UAVs at time slot n
$\tilde{l}_1(n)$	The location of the first reflector of IRS at time slot n
$\hat{l}_u(n)$	The location of user u at time slot n
$\bar{l}_m(n)$	The location of UAV m at time slot n
K_x	The number of reflecting elements along the X-axis
K_z	The number of reflecting elements along the Z-axis
K	Equals to $K_x \cdot K_x$, the total number of reflectors of IRS
$d_{u,m}(n)$	Euclidean distance of user u and UAV m at time slot n
$h_{u,m}(n)$	Direct channel gain between user u and UAV m at time slot n
$\hat{g}_{u,m}(n)$	The IRS assisted channel gain between user u and UAV m at time slot n
$\phi_k(n)$	The phase shift of reflector k of IRS at time slot n
$R_{u,m}(n)$	The transmission rate between user u and UAV m at time slot n
$\mathbf{\Phi}(n)$	The diagonal reflection matrix of IRS phase shifts at time slot n
B	The communication bandwidth
σ^2	The Gaussian noise
f_e, f_w	The input feature vectors represent the energy cost of users to UAVs and workload of UAVs
$\boldsymbol{\mathcal{P}}(n)$	DNN predicted probability matrix at time slot n
H	The number of quantized binary offloading decisions
$\boldsymbol{\beta}^h$	H binary offloading decisions quantized by OPPO
$\boldsymbol{\beta}^*(n)$	The one yielding the lowest energy cost among the H candidate offloading decisions generated by OPPO at time slot n

The proposed MEC system operates as follows: at a time frame n within the system time $\mathcal{N} = \{1, 2, \ldots, n, \ldots, N\}$, each user in the system has a computational task that needs to be processed. The primary objective is to utilize the available computational resources, such as UAVs and UEDs, to complete all users' tasks within an acceptable time while minimizing the total energy consumed during task processing. To achieve this objective, an offloading decision that allocates user tasks to the appropriate computational resources is required. Initially, the central server, located at the base station, collects essential information, such as the locations, computational power of users, UAVs, etc. Subsequently, the collected information is input into an offloading decision prediction model, which is discussed in detail in Section V. This model predicts an offloading allocation matrix denoted as $\boldsymbol{\beta}(n) \in \{0, 1\}^{U \times (M+1)}$, where U represents the number of users and M represents the number of UAVs. For

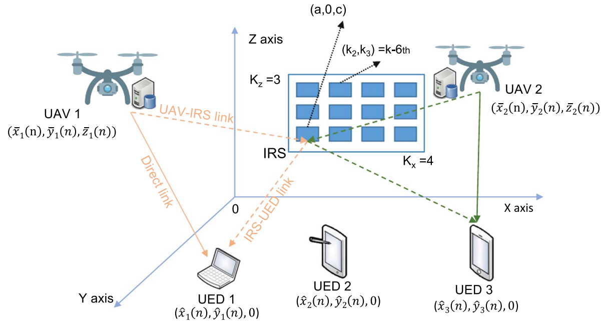

(a,0,c)
Z axis
(k_2,k_3) =k-6th
K_z =3
UAV 1
$(\bar{x}_1(n),\bar{y}_1(n),\bar{z}_1(n))$
UAV-IRS link
UAV 2
$(\bar{x}_2(n),\bar{y}_2(n),\bar{z}_2(n))$
IRS
K_x =4
Direct link
IRS-UED link
0
X axis
Y axis
UED 1
$(\hat{x}_1(n),\hat{y}_1(n),0)$
UED 2
$(\hat{x}_2(n),\hat{y}_2(n),0)$
UED 3
$(\hat{x}_3(n),\hat{y}_3(n),0)$

Fig. 2. The proposed system includes K reflectors. The first reflector serves as a reference point and is positioned at (a, 0, c).

a given user u, $\beta_{u,m}(n) = 1$ indicates that the corresponding task is offloaded to UAV m ($m \leq M$), and $\beta_{u,M+1}(n) = 1$ signifies that the task is processed locally on the user's UED. In the proposed system, we assume that when a task is offloaded to UAVs, it can only be offloaded to a single UAV at a time, prohibiting simultaneous offloading to multiple UAVs. This constraint is mathematically expressed as $\sum_{m=1}^{M} \beta_{u,m}(n) = 1$ for each user $u \in \mathcal{U}$.

B. Data Transmission in the THz Network

In this section, we elucidate the data transmission within the THz network. As depicted in Fig. 2, at time frame n, there are two approaches for transmitting user data and tasks to UAVs: (i) direct transmission of user data from UEDs to UAVs, and (ii) redirection of user data to UAVs through the IRS. Both approaches are employed simultaneously to facilitate data transmission by the users. According to the Shannon theorem, the achievable throughput $R_{u,m}(n)$ for user u to transmit data to the m-th UAV is determined as follows:

$$R_{u,m}(n) = B \log_2 \left(1 + \frac{p \left| h_{u,m}(n) + \hat{g}_{u,m}(n) \right|^2}{\sigma^2} \right), \quad (1)$$

where $h_{u,m}(n)$ denotes the channel gain for direct data transmission and $\hat{g}_{u,m}(n)$ is the channel gain of transmitting data through the IRS. We assume that when multiple UEDs upload their tasks to UAVs simultaneously, the available wireless bandwidth is equally shared among them. Given this setup and the high transmission speed of the THz bandwidth, it is reasonable to assume that the transmission time is within the channel coherence time. This assumption, commonly adopted in prior works [28], [36], [37], allows each task packet to be transmitted over a flat fading quasi-static channel. Accordingly, B represents the channel bandwidth allocated to each UED. p represents the transmission power provided by the base station and σ^2 is a Gaussian noise for modeling random noise that affects the communication.

In the case of direct data transmission, given the coordinate of user u, denoted as $\hat{l}_u(n) = (\hat{x}_u(n), \hat{y}_u(n), 0)^T$ and the coordinate of the m-th UAV, denoted as $\bar{l}_m(n) = (\bar{x}_m(n), \bar{y}_m(n), \bar{z}_m(n))^T$, the euclidean distance $d_{u,m}(n)$ between them can be formulated as: $d_{u,m}(n) =$

$$\sqrt{(\bar{x}_m(n) - \hat{x}_u(n))^2 + (\bar{y}_m(n) - \hat{y}_u(n))^2 + \bar{z}_m^2(n)}. \quad (2)$$

Given the distance $d_{u,m}(n)$, the channel gain for direct transmission $h_{u,m}(n)$ is defined as follows:

$$h_{u,m}(n) = \left(\frac{\mathcal{C}}{4\pi f d_{u,m}(n)} \right) \cdot \exp\left(\frac{-j2\pi f d_{u,m}(n)}{\mathcal{C}} + \frac{-K(f)\, d_{u,m}(n)}{2} \right), \quad (3)$$

where $\mathcal{C}$ represents the speed of light, f denotes the frequency of the sub-band, j is the imaginary unit, and $K(f)$ represents the absorption coefficient of the transmission medium.

In the context of data transmission via an IRS, the IRS acts as an intermediary that receives data from the data-sending device and subsequently reflects the data to the receiver. As depicted in Fig. 2, the IRS is situated on the X-Z plane and comprises a total of $K = K_x \cdot K_z$ reflecting elements. K_x and K_z represent the quantities of reflecting elements along the X-axis and Z-axis, respectively. The coordinates of the reflecting elements in the IRS are determined based on the position of the first reflecting element, denoted as $\tilde{l}_1 = (a, 0, c)^T$, which is located at the lower-left corner of the IRS. Accordingly, the coordinates of the k-th reflecting element ($k = k_z + (k_x - 1)K_z$), denoted as $\tilde{l}_k$, can be calculated using the following expression:

$$\tilde{l}_k = (a + (k_x - 1)\delta_x,\ 0,\ c + (k_z - 1)\delta_z)^T, \quad (4)$$

where k_x and k_z represent the indices of the reflecting element along the X-axis and Z-axis, respectively. δ_x and δ_z denote the gaps between the elements along the X-axis and Z-axis.

It is worth noting that the first element of the IRS is considered as the reference point. Hence, the distance between the IRS and communication points like UAVs or UEDs can be approximated by measuring the distance between the reference point and the corresponding point [24]. Therefore, the transmission vector from the IRS (approximated to be the first reflecting element) to the UAV m is represented as $\Delta\bar{r}_m(n) = \bar{l}_m(n) - \tilde{l}_1 = (\bar{x}_m(n) - a, \bar{y}_m(n), \bar{z}(n) - c)^T$. The difference vector between the first reflecting element and the k-th reflecting element is defined as $\Delta\tilde{r}_k = \tilde{l}_k - \tilde{l}_1 = ((k_x - 1)\delta_x, 0, (k_z - 1)\delta_z)^T$. Accordingly, for signals transmitted to the m-th UAV through the IRS, the phase difference between the signal reflected by the first reflecting element and the signal reflected by the k-th element can be formulated as follows:

$$\theta_k^m(n) = \frac{2\pi f}{\mathcal{C}} \frac{\Delta\tilde{r}_k^T}{|\Delta\tilde{r}_k|} \Delta\bar{r}_m(n)$$
$$= \frac{2\pi f}{|\Delta\tilde{r}_k|\, \mathcal{C}} \left((\bar{x}_m(n) - a)(k_x - 1)\delta_x + (\bar{z}(n) - c)(k_z - 1)\delta_z \right). \quad (5)$$

Similarly, the transmission vector from the first reflecting element of the IRS to user u can be defined as $\Delta\hat{r}_u(n) = \hat{l}_u(n) - \tilde{l}_1 = (\hat{x}_u(n) - a,\ \hat{y}_u(n),\ -c)^T$ and the phase difference between the signal sent to the user by the first reflecting element and the signal sent by the k-th element can be formulated as follows:

$$\nu_k^u(n) = \frac{2\pi f}{\mathcal{C}} \frac{\Delta \tilde{r}_k^T}{|\Delta \tilde{r}_k|} \Delta \hat{r}_u(n)$$
$$= \frac{2\pi f}{|\Delta \tilde{r}_k| \, \mathcal{C}} \left((\hat{x}_u(n) - a)(k_x - 1)\delta_x - c(k_z - 1)\delta_z \right). \quad (6)$$

The cascaded channel gain of the UAV-IRS-UED connection can be defined as:

$$g_{u,m}(n) = \left(\frac{\mathcal{C}}{8\sqrt{\pi^3} f d'_{u,m}(n)} \right) \cdot$$
$$\exp\left(\frac{-j2\pi f d'_{u,m}(n)}{\mathcal{C}} + \frac{-K(f) d'_{u,m}(n)}{2} \right). \quad (7)$$

The variable $d'_{u,m}(n)$ is defined as $\hat{d}_u(n) + \bar{d}_m(n)$. So we sum the distance between user u and the first reflector of IRS, denoted by $\hat{d}_u(n) = ||\Delta \hat{r}_u(n)||_2$, and $\bar{d}_m(n) = ||\Delta \bar{r}_m(n)||_2$, which represents the distance between UAV m and the first reflector of IRS [23]. Finally, the channel gain for UAV-IRS-UED data transmission is defined as:

$$\hat{g}_{u,m}(n) = g_{u,m}(n) \, \bar{\boldsymbol{e}}_m(n)^T \, \boldsymbol{\Phi}(n) \, \hat{\boldsymbol{e}}_u(n), \quad (8)$$

where $\bar{\boldsymbol{e}}_m(n) = (\exp(j\theta_1^m(n)), \ldots, \exp(j\theta_K^m(n)))^T$, $\hat{\boldsymbol{e}}_u(n) = (\exp(j\nu_1^u(n)), \ldots, \exp(j\nu_K^u(n)))^T$, and $\boldsymbol{\Phi}(n) = diag(\exp(j\phi_1(n)), \ldots, \exp(j\phi_K(n)))$ is diagonal matrix of IRS phase shifts, where $\phi_k(n)$ is the phase shift of the k-th reflecting element.

C. System Energy Consumption

In this section, we formulate the energy consumed in the MEC system. The energy cost within the system consists of two parts: (i) the energy consumed by processing user tasks on UEDs and (ii) the energy consumed by processing user tasks on UAVs. At a given time frame n, let us consider user u with its corresponding task denoted as $\Psi_u(n) = \{D_u(n), T_u(n), C_u(n)\}$. Here, $D_u(n)$ represents the size of the data, $T_u(n)$ represents the tolerable latency, and $C_u(n)$ represents the CPU cycles required to process the task. If the task is processed on the user's UED (i.e., $\beta_{u,M+1}(n) = 1$), the energy consumed can be defined as:

$$E_u^{local}(n) = t_u^{local}(n) \cdot p_u, \quad (9)$$

where p_u represents the energy consumed by the UED per CPU clock and $t_u^{local}(n)$ denotes the time required for processing the user's task (measured in CPU clock):

$$t_u^{local}(n) = C_u(n)/Z_u, \quad (10)$$

where Z_u refers to the CPU clock speed of the UED. It is assumed that both Z_u and p_u remain constant over time.

If user u's task is processed on UAVs (i.e., $\sum_{m\in\mathcal{M}} \beta_{u,m}(n) = 1$), the energy consumed during this process can be divided into two parts: (i) the energy consumed for uploading the task to UAVs and (ii) the energy consumed during the task processing on UAVs. The energy consumed in transmitting data from user u to UAVs is defined as follows:

$$E_u^{tran}(n) = t_u^{tran}(n) \cdot p_u^{tran}, \quad (11)$$

where p_u^{tran} represents the energy consumed per second and $t_u^{tran}(n)$ denotes the transmission time (measured in second):

$$t_u^{tran}(n) = \frac{D_u(n)}{\sum_{m\in\mathcal{M}} R_{u,m}(n) \cdot \mathbf{I}[\beta_{u,m}(n) = 1]} \quad (12)$$

where $\mathbf{I}[\beta_{u,m}(n) = 1]$ is an indicator function that takes a value of 1 if $\beta_{u,m}(n) = 1$, and a value of 0 otherwise.

Regarding the energy consumed in processing user u's task on UAVs, it can be defined as:

$$E_u^{comp}(n) = \sum_{m\in\mathcal{M}} t_{u,m}^{comp}(n) \cdot p_m \cdot \mathbf{I}[\beta_{u,m}(n) = 1], \quad (13)$$

where p_m represents the energy consumed by UAV m per CPU clock, and $t_{um}^{comp}(n)$ denotes the number of CPU clocks required to process user u's task on UAV m.

$$t_{um}^{comp}(n) = \frac{C_u(n)}{Z_m / w_m(n)}. \quad (14)$$

In this context, Z_m represents the CPU clock speed of UAV m, while $w_m(n) = \max(1, \sum_{u\in\mathcal{U}} \beta_{u,m}(n))$ denotes the workload status of UAV m. The workload refers to the current number of tasks being processed on UAV m.

Hence, the energy consumption attributed to user u can be formulated as $E_u^{total}(n) =$

$$G \cdot \left(E_u^{tran}(n) + E_u^{comp}(n) \right) + (1 - G) \cdot E_u^{local}(n), \quad (15)$$

where $G = 1 - \beta_{u,M+1}(n)$.

The overall system energy is defined as the aggregate of the energy consumed by all users within the system:

$$E^{total}(n) = \sum\nolimits_{u\in\mathcal{U}} E_u^{total}(n). \quad (16)$$

IV. Optimization Problem

In the given system time frame $n \in \mathcal{N}$, our objective is to minimize the total energy consumption $E^{total}(n)$ of all the UAVs and UEDs, while considering various constraints. To simplify the notation, we denote the coordinates of all users and UAVs in the system as $\boldsymbol{L}(n)$, the CPU clock speed of UAVs and UEDs as $\boldsymbol{Z}(n)$, and the task information of all users as $\boldsymbol{\Psi}(n)$. We rewrite the total energy consumed in the system $E^{total}(n)$ as:

$$E^{total}(n)\{\boldsymbol{\beta}, \boldsymbol{\phi} | \boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}\} = \sum_{u\in\mathcal{U}} E_u^{total}(n)\{\boldsymbol{\beta}, \boldsymbol{\phi} | \boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}\} \quad (17)$$

to highlight the dependent variables, where the '(n)' terms in $\boldsymbol{L}(n), \boldsymbol{\Psi}(n), \boldsymbol{Z}(n), \boldsymbol{\beta}(n), \boldsymbol{\phi}(n)$ are omitted for convenience. Accordingly, the optimization problem can be formulated as:

$$\mathcal{P}1: \min_{\boldsymbol{\beta}(n), \boldsymbol{\phi}(n)} E^{total}(n)\{\boldsymbol{\beta}, \boldsymbol{\phi} | \boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}\} \quad (18)$$

$$\textbf{s.t.}\ \beta_{u,m}(n) \in \{0, 1\}, \forall u \in \mathcal{U}, m \leq M + 1, \quad (18a)$$

$$\sum_{m=1}^{M+1} \beta_{u,m}(n) = 1, \quad (18b)$$

$$0 \leq \phi_k(n) \leq 2\pi, 1 \leq k \leq K, \quad (18d)$$

$$t_u^{comp}(n) + t_u^{tran}(n) + t_u^{local}(n) \leq T_u(n), \forall u \in \mathcal{U}. \quad (18f)$$

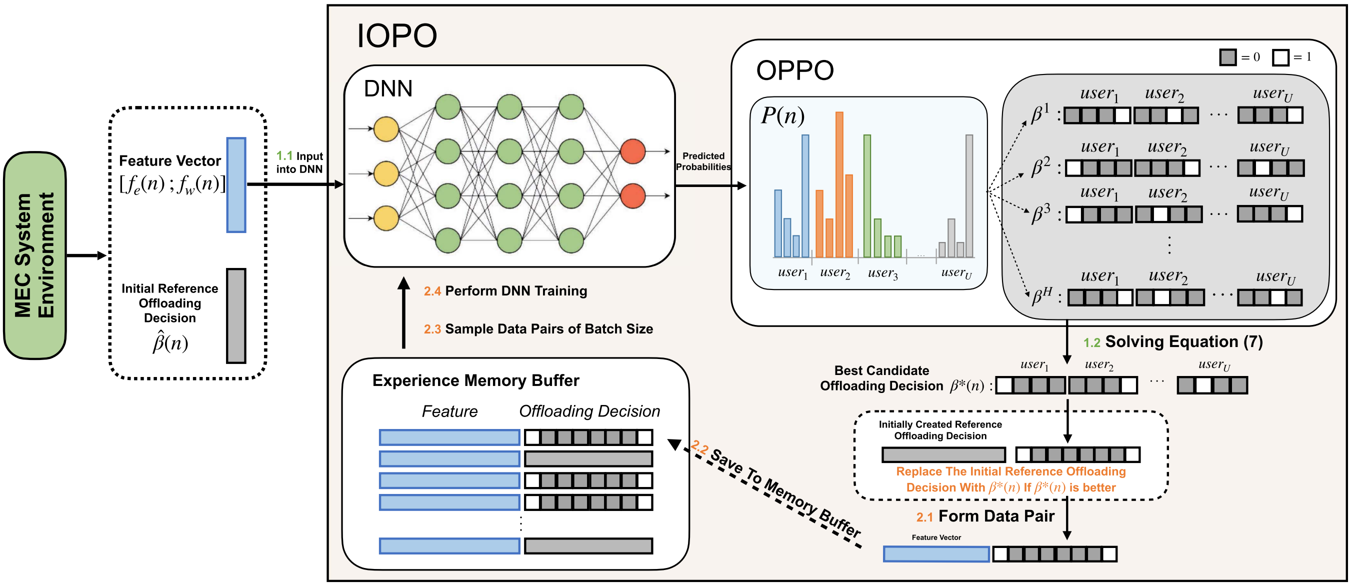

IOPO
MEC System Environment
Feature Vector $[f_e(n); f_w(n)]$
Initial Reference Offloading Decision $\hat{\beta}(n)$
1.1 Input into DNN
DNN
Predicted Probabilities
OPPO
$P(n)$
$user_1$ $user_2$ $user_3$ $user_U$
= 0 = 1
β^1 β^2 β^3 β^H
$user_1$ $user_2$ $user_U$
1.2 Solving Equation (7)
Best Candidate Offloading Decision $\beta^*(n)$
Initially Created Reference Offloading Decision
Replace The Initial Reference Offloading Decision With $\beta^*(n)$ If $\beta^*(n)$ is better
2.1 Form Data Pair
Feature Vector
2.2 Save To Memory Buffer
Experience Memory Buffer
Feature
Offloading Decision
2.3 Sample Data Pairs of Batch Size
2.4 Perform DNN Training

Fig. 3. The structure of the proposed IOPO Framework. The IOPO framework consists of two processes: offloading decision generation (Steps 1.1 and 1.2) and offloading decision update (Steps 2.1, 2.2, 2.3, and 2.4). Essential operations of the algorithm encompass generating the system feature, generating offloading decisions, evaluating offloading decisions, and updating the network.

It means that given $\{\boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}\}$, we want to find the offloading decision $\boldsymbol{\beta}(n)$ and the IRS phase $\boldsymbol{\phi}(n) = \{\phi_1(n), \phi_2(n), \ldots, \phi_K(n)\}$ such that the total energy consumed is minimized. The best offloading decision and the best IRS phase shifts are denoted as $\boldsymbol{\beta}^{\circ}(n)$ and $\boldsymbol{\phi}^{\circ}(n)$ respectively. Constraints (18a) and (18b) ensure that at the time frame n, each user is assigned only one task, which can be either allocated to one of the M UAVs or executed locally on the UED. The Constraint (18d) guarantees the angle of the k-th reflector of IRS remains within the range of 0 and 2π. Lastly, Constraint (18f) ensures that the task of user u is completed within the acceptable delay threshold $T_u(n)$.

Problem $\mathcal{P}1$ presents a formidable challenge as it belongs to the category of NP-hard mixed-integer non-linear programming (MINLP) problems. To tackle this challenge, we propose a two-stage approach. For the first step, we focus on generating the offloading decision $\boldsymbol{\beta}^*(n)$. In this study, we introduce a deep learning-based offloading decision generation model capable of generating high-quality offloading decisions within milliseconds. The intricate details of this model are elucidated in Section V-B. Once the offloading decision $\boldsymbol{\beta}^*(n)$ is obtained from the offloading decision model, the subsequent step involves identifying the phase shifts $\boldsymbol{\phi}^*(n)$ for the IRS that minimize the overall system energy consumption, given the decision $\boldsymbol{\beta}^*(n)$. The optimization of IRS phase shifts is explained in detail in Section V-E and can be formulated as:

$$\mathcal{P}2: \min_{\boldsymbol{\phi}(n)} E^{total}(n)\{\boldsymbol{\phi}|\boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}, \boldsymbol{\beta}^*\}, \quad \textbf{s.t.}\ (18d).$$

V. The IOPO Framework

A. IOPO Framework Overview

The proposed Iterative Order-Preserving Policy Optimization (IOPO) Framework, as illustrated in Fig. 3, comprises two alternating stages: (i) *offloading decision generation* and (ii) *offloading policy update*. In the offloading decision generation stage, a deep neural network (DNN) offloading decision prediction model denoted as f_θ is utilized to predict an energy-efficient task offloading allocation. For the n-th system time frame ($n \in \mathcal{N}$), the DNN takes the input feature $[f_e(n); f_w(n)]$ constructed based on the status of system environment, and outputs a probability matrix $\boldsymbol{\mathcal{P}}(n)$, representing the probabilities of different offloading allocations that each user may adopt at time n. The probability matrix is then quantized into H candidate offloading decisions within the Order-Preserving Policy Optimization (OPPO) unit. Among these candidate decisions, the one yielding the lowest system energy cost is selected as the predicted offloading decision for the current time frame, denoted as $\boldsymbol{\beta}^*(n)$. Subsequently, the generated offloading decision $\boldsymbol{\beta}^*(n)$, along with the corresponding input feature vector, are stored in the experience memory buffer for subsequent DNN training.

In the offloading decision update stage, a batch of training samples is randomly selected from the memory buffer to train the DNN f_θ, resulting in the update of DNN parameters θ. The updated DNN is then utilized to produce offloading decisions in the subsequent system time frames. Detailed descriptions of these two stages are provided in the following subsections.

B. Offloading Decision Generation

At a system time frame $n \in \mathcal{N}$, the input to DNN is a feature vector $[f_e(n); f_w(n)]$ formed by concatenating two distinct feature vectors: $f_e(n)$ and $f_w(n)$, where '$[\cdot;\cdot]$' denotes the vector concatenation operator. The first feature vector $f_e(n) \in \mathbb{R}^{(M+1)\times U}$ represents the energy costs associated with each of the U users and their $M+1$ offloading options. The second feature vector $f_w(n) \in \mathbb{R}^M$ encodes the CPU clock speed of

M UAVs. The two feature vectors are concatenated to form the DNN input feature vector, which possesses a shape of $(M+1)\times U+M$. The DNN offloading decision model f_θ with parameters θ, is a multilayer perceptron (MLP) consisting of an input layer, six hidden layers, and an output layer. The activation function employed in both the input and hidden layers is the hyperbolic tangent (Tanh) function, while the softmax function is utilized in the output layer. In order to enhance the model's generalization capability and mitigate the potential overfitting issue, a dropout layer [38] is incorporated between each pair of consecutive hidden layers.

Given the input feature $[f_e(n); f_w(n)]$, the DNN predicts a probability matrix $\boldsymbol{\mathcal{P}}(n)=\{p_{u,m}(n)\mid p_{u,m}(n)\in[0,1], u\in\mathcal{U}, m\in\{1,2,\ldots,M+1\}\}$. Each element in the matrix holds a value ranging from 0 to 1, and the matrix has a dimension of $U\times(M+1)$. The probability matrix $\boldsymbol{\mathcal{P}}(n)$ signifies the probability of different offloading allocations that each user may adopt at the system time n. Specifically, the $p_{u,m}(n)$ denotes the probability that user u offloads its task to UAV m, while $p_{u,M+1}(n)$ denotes the probability that user u is assigned to execute the task locally on its UED. This process can be mathematically formulated as follows:

$$\boldsymbol{\mathcal{P}}(n)=f_\theta\left([f_e(n); f_w(n)]\right).$$

The next step is to transform the probability matrix $\boldsymbol{\mathcal{P}}(n)$ into the offloading decision matrix $\boldsymbol{\beta}(n)$. To accomplish this, we first feed the probability matrix into a novel Order-Preserving Policy Optimization (OPPO) unit, where H candidate offloading decisions are generated based on the DNN output. Then, the candidate offloading decision with the minimum energy cost is chosen from this set of H decisions to serve as the predicted offloading matrix $\boldsymbol{\beta}^*(n)$.

The OPPO unit is derived from the order-preserving optimization method proposed in [27]. The original order-preserving algorithm generates a set of H candidate offloading decisions, where the dissimilarity between any two candidate decisions is maximized. This approach promotes diversity among the candidate solutions, thereby increasing the chance of identifying the optimal decision. However, the order-preserving method described in [27] is specifically designed for systems that consist of a single MEC infrastructure. As the proposed MEC system consists of multiple UAVs and users, the original approach is not suitable. Hence, we modify the order-preserving optimization algorithm to align with our system configuration, resulting in the modified approach referred to as OPPO. Specifically, given the DNN predicted probability matrix $\boldsymbol{\mathcal{P}}(n)\in\mathbb{R}^{U\times(M+1)}$, where U represents the number of users and M denotes the number of UAVs in the system, OPPO generates a set of H candidate offloading decisions, where the hyper-parameter H is a positive integer chosen from the range of $\{1,2,\ldots,U\times(M+1)\}$.

The first candidate offloading decision $\boldsymbol{\beta}^1$ can be obtained through the following procedure. For the u-th row of $\boldsymbol{\mathcal{P}}(n)$, we identify the index of the highest probability within that row using $z_0=\arg\max_{z\in\{1,2,\ldots,M+1\}} p_{u,z}$. Subsequently, we set β^1_{u,z_0} to 1, while assigning 0 to the remaining M elements within that row. Mathematically, this process can be expressed as follows:

$$\beta^1_{u,m}=\begin{cases}1 & m=z_0 \text{ and } p_{u,m}>\mathcal{T}_0,\\ 0 & otherwise.\end{cases}$$

where $\mathcal{T}_0=1/(M+1)$. To generate the remaining $H-1$ offloading decisions, we begin by arranging all $U\times(M+1)$ elements of $\mathcal{P}(n)$ in ascending order based on their distances from $\mathcal{T}_0$. This sorted matrix is denoted as $\mathcal{T}=\{p'_{1,1},p'_{1,2},\ldots,p'_{U,M+1}\}$. Here, the element $p'_{i,j}$ becomes the h-th threshold denoted as $\mathcal{T}_h$, where $h=(i-1)\cdot(M+1)+j$, and i and j represent the row and column indices of $p'_{i,j}$, respectively. For instance, $\mathcal{T}_1=p'_{1,1}$ corresponds to the probability element with the smallest distance to $\mathcal{T}_0$. Subsequently, the h-th offloading decision, denoted as $\boldsymbol{\beta}^h$ (where $h\in\{2,3,\ldots,H\}$), is defined according to three generation rules.

The first generation rule states that for the u-th row of $\boldsymbol{\mathcal{P}}(n)$, if $\mathcal{R}1=\{(u,z_1)\mid p_{u,z_1}>\mathcal{T}_{h-1}, z_1\in\{1,2,\ldots,M+1\}\}$ is not an empty set, then we assign $\beta^h_{u,z_1}=1$, while setting the remaining M values to 0. Mathematically, this can be expressed as:

$$\beta^h_{u,m}=\begin{cases}1 & m=z_1,\\ 0 & otherwise.\end{cases}$$

If there are multiple elements in $\mathcal{R}1$, we utilize the first (u,z_1) pair only and omit the remaining elements to meet the constraint (18b). In the case where $\mathcal{R}1$ is an empty set, we proceed to apply the second generation rule. Specifically, for the u-th row of $\boldsymbol{\mathcal{P}}(n)$, if $\mathcal{R}2=\{(u,z_2)\mid p_{u,z_2}=\mathcal{T}_{h-1}, p_{u,z_2}\le\mathcal{T}_0, z_2\in\{1,2,\ldots,M+1\}\}$ is not an empty set, we assign a value of 1 to β^h_{u,z_2} while setting the remaining elements to 0. This can be expressed mathematically as:

$$\beta^h_{u,m}=\begin{cases}1 & m=z_2,\\ 0 & otherwise.\end{cases}$$

Again, if there are multiple elements in $\mathcal{R}2$, we only utilize the first (u,z_2) pair and omit the remaining elements. Lastly, in the scenario where both $\mathcal{R}1$ and $\mathcal{R}2$ are all empty, we employ the third generation rule, whereby the task is assigned to be executed locally:

$$\beta^h_{u,m}=\begin{cases}1 & m=M+1,\\ 0 & otherwise.\end{cases}$$

Upon completion of the OPPO, we obtain a collection of H candidate offloading decisions, denoted as $\{\beta^1,\beta^2,\ldots,\beta^H\}$. Subsequently, we identify the optimal candidate offloading decision among them, which corresponds to the one that minimizes the overall system energy cost. This process can be mathematically formulated as follows:

$$\boldsymbol{\beta}^*(n)=\mathop{\arg\min}_{\boldsymbol{\beta}^i\in\{\boldsymbol{\beta}^1,\boldsymbol{\beta}^2,\ldots,\boldsymbol{\beta}^H\}} E^{total}(n)\{\boldsymbol{\beta}^i, f_{WOA}(\boldsymbol{\beta}^i)|\boldsymbol{L},\boldsymbol{\Psi},\boldsymbol{Z}\}, \tag{19}$$

where E^{total} is (17) and $f_{WOA}(\cdot)$ corresponds to the WOA method for producing optimized IRS phase shifts (introduced in Section V-E). Please be noted that, as the OPPO unit can generate H candidate offloading decisions based on the DNN output, it can also be perceived as an effective solution searching unit, in which offloading decisions with low energy costs are discovered. Throughout the execution of IOPO, OPPO continuously explores offloading decisions that are more energy-efficient. These newly discovered offloading decisions are subsequently utilized in the offloading policy update procedure to update the DNN parameters θ.

After obtaining the predicted offloading decision $\boldsymbol{\beta}^*(n)$, we employ the function $\boldsymbol{\phi}^*(n) = f_{WOA}(\boldsymbol{\beta}^*(n))$ to compute the optimized IRS phase shifts $\boldsymbol{\phi}^*(n)$. By substituting $\boldsymbol{\beta}^*(n)$ and $\boldsymbol{\phi}^*(n)$ into (17), we can evaluate the energy cost of the system. However, in order to address $\mathcal{P}1$, it is imperative for the predicted offloading decision $\boldsymbol{\beta}^*(n)$ to align with, or at least closely approximate, the optimal offloading decision $\boldsymbol{\beta}^\circ(n)$ (i.e., $\boldsymbol{\beta}^*(n) = \boldsymbol{\beta}^\circ(n)$ or $\boldsymbol{\beta}^*(n) \approx \boldsymbol{\beta}^\circ(n)$). To achieve this alignment, it is necessary to implement an offloading policy update procedure, which enables the DNN to learn to generate desired offloading decisions accurately. Furthermore, the desired offloading decisions utilized in DNN training should also be gradually improved as the IOPO executes. As a result, the offloading decisions predicted by the IOPO framework, which are derived from DNN outputs, exhibit a gradual improvement and ultimately converge towards optimal offloading decisions.

However, during the initial stage of the IOPO execution, the DNN is not yet adequately trained. As a result, the predicted offloading decision $\boldsymbol{\beta}^*(n)$ may exhibit poor quality. Learning from these low-quality offloading decisions could hinder the convergence towards optimal offloading decisions, particularly in systems with a substantial number of UAVs and users (wherein a poorly performing DNN finds it challenging to predict the optimal decision among a total of $(M+1)^U$ possible offloading decisions, with M, U denoting the number of UAVs and the number of users within the system). To address this issue and expedite the convergence process, an intuitive approach provides a favorable starting point for the DNN to learn. Hence, we introduce an initial reference offloading decision $\hat{\boldsymbol{\beta}}(n)$ with high quality (the generation of this initial reference offloading decision is elaborated in Section VI-B). At the early stages of the IOPO execution, $\hat{\boldsymbol{\beta}}(n)$ may exhibit lower energy cost compared to $\boldsymbol{\beta}^*(n)$, thereby enabling faster convergence toward the optimal offloading decisions when learning from $\hat{\boldsymbol{\beta}}(n)$. As the IOPO execution progresses, the DNN gradually improves, and the predicted offloading decision $\boldsymbol{\beta}^*(n)$ based on the DNN output can surpass the initial reference offloading decision. Consequently, we compare the predicted offloading decision $\boldsymbol{\beta}^*(n)$ with the initially provided reference offloading decision $\hat{\boldsymbol{\beta}}(n)$. If the MEC system achieves lower energy costs with $\boldsymbol{\beta}^*(n)$ compared to $\hat{\boldsymbol{\beta}}(n)$, we update the reference offloading decision to $\boldsymbol{\beta}^*(n)$ (i.e., $\hat{\boldsymbol{\beta}}(n) = \boldsymbol{\beta}^*(n)$). This ensures that the DNN can always learn from high-quality offloading decisions.

Subsequently, we maintain a memory buffer with limited capacity. At the n-th time frame, a new training data sample $([f_e(n); f_w(n)], \hat{\boldsymbol{\beta}}(n))$ is added to the memory buffer. When the memory buffer is full, the newly generated data sample replaces the oldest one.

C. Offloading Policy Update

To train the DNN offloading decision model f_θ, first, we sample a batch of data pairs, denoted by $\mathcal{B}$, from the memory buffer, where $j \in \mathcal{B}$ implies the data pair generated in j-th time frame, $([f_e(j); f_w(j)], \hat{\boldsymbol{\beta}}(j))$, is in this batch. Subsequently, the parameters θ of the DNN are updated to minimize the average Maximum Likelihood Estimation (MLE) loss. The MLE loss for pair j in the training batch $\mathcal{B}$ is defined as follows:

$$\ell(j) = -\sum_{u=1}^{U} \sum_{m=1}^{M+1} \hat{\beta}_{u,m}(j) \log\Big(p(\hat{\beta}_{u,m}(j) | \left[f_e(j); f_w(j)\right], \theta)\Big),$$

where $\hat{\beta}_{u,m}(j)$ refers to the reference allocation decision of the data pair $j \in \mathcal{B}$ and $[f_e(j); f_w(j)]$ is the input feature associates with the data pair $j \in \mathcal{B}$. The average MLE loss for the given training batch is formulated as:

$$\mathcal{L}(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum\nolimits_{j \in \mathcal{B}} \ell(j),$$

where $|\mathcal{B}|$ denotes the batch size. The parameter θ is updated using the Adam optimizer [39] and is updated every λ IOPO execution step. By minimizing $\mathcal{L}$, the IOPO-predicted offloading decisions are refined progressively and eventually align with optimal offloading decisions (demonstrated in experiment Section VII-C). With the optimal offloading allocations produced and the optimal phase shifts obtained using the WOA algorithm (introduced in Section V-E), problem $\mathcal{P}1$ can be solved. The pseudo-code of IOPO is presented in Algorithm 1.

D. Computational Complexity Analysis

As illustrated in Fig. 3, the core processes of the IOPO algorithm involve generating system features, producing offloading decisions, evaluating these decisions, and updating the network. First, the system feature, which includes the information on UAVs and UEDs, is obtained, as shown in (17). The system comprises M UAVs and U UEDs, with each UED assigned one task, resulting in U tasks and a complexity of $O(M + 2U)$. Second, the probability matrix is computed, followed by the generation of offloading decisions. The computation of the probability matrix only requires a forward pass through the network, which is dependent solely on the network size (which is simple in our structure), and can therefore be considered to have a constant time complexity [40]. The OPPO unit is responsible for generating offloading decisions. The quantization process involves a fixed number of operations, including selecting the largest element from each user u and finding the index of the highest probability. This operation requires a maximum search over $M+1$ elements for each user, resulting in a complexity of $O(U(M+1)) = O(UM)$. Next, all $U \times (M+1)$ elements are arranged in ascending order, which takes $O((U(M+1)) \log(U(M+1)))$, simplifying to $O((UM) \log(UM))$. Then, the remaining $H-1$ candidate offloading decisions are generated, each requiring $O(U(M+1)) = O(UM)$ operations, leading to a total complexity of $O((H-1)UM) = O(HUM)$. Combining all these steps, the overall time complexity for generating H candidate offloading decisions using the OPPO algorithm is $O(UM) +$

Algorithm 1: The Execution of the IOPO Framework.

```
Input : Input feature $f(n) = [f_e(n); f_w(n)]$ at each
        time frame $n$, and an initial reference
        offloading decision $\hat{\boldsymbol{\beta}}(n)$.
Output: Final Offloading decision $\hat{\boldsymbol{\beta}}(n)$ and the best
        IRS phase shifts for each time frame $n$.
Randomly initialize parameters $\theta$ of DNN $f_\theta$ and
  empty the memory buffer.;
for $n = 1, 2, \ldots, N$ do
    Compute the DNN probability matrix:
      $\boldsymbol{\mathcal{P}}(n) = f_\theta([f_e(n); f_w(n)])$;
    Feed $\boldsymbol{\mathcal{P}}(n)$ into OPPO, where $\boldsymbol{\mathcal{P}}(n)$ is quantized
      into $H$ candidate offloading decisions;
    Select the best candidate decision $\boldsymbol{\beta}^*(n)$ using
      Eqn. (19);
    Obtain the best IRS phase shifts $\boldsymbol{\phi}^*(n)$ using
      $\boldsymbol{\phi}^*(n) = f_{WOA}(\boldsymbol{\beta}^*(n))$ as shown in Sec. V-E;
    if $\boldsymbol{\beta}^*(n)$ is better than the initially provided
      reference offloading decision $\hat{\boldsymbol{\beta}}(n)$ then
        $\hat{\boldsymbol{\beta}}(n) = \boldsymbol{\beta}^*(n)$ ;
    end
    Update the memory buffer by adding $\big(f(n), \hat{\boldsymbol{\beta}}(n)\big)$;
    if $n \bmod \lambda = 0$ then
        Randomly sample a batch $\mathcal{B}$ from the memory
          buffer as $\{([f_e(j); f_w(j)], \hat{\boldsymbol{\beta}}(j)) \mid j \in \mathcal{B}\}$;
        Train the DNN on $\mathcal{B}$ and update $\theta$ using the
          Adam optimizer;
    end
end
```

$O((UM)\log(UM)) + O(HUM)$, which can be approximated as $O(HUM + (UM)\log(UM))$. Third, the evaluating complexity using WOA depends on the number of whales W and the number of evolution round E, the energy cost is computed according to (16), with a given offloading decision, the complexity is $O(U)$. Here, we must calculate the best among H candidates' offloading decisions. Thus, it is $O(HWEU)$. Without applying OPPO, we would need to consider $(M+1)^U$ offloading decisions instead of H, significantly increasing the complexity. These steps are executed sequentially to be completed in polynomial time.

Moreover, the complexity of updating the MLP network is dependent on the loop over N times, which involves operations across the network layers. Sampling a batch from the memory buffer every λ time is $O(|\mathcal{B}|)$. Training the DNN on the batch using the Adam optimizer is $O(|\mathcal{B}|\mathcal{L}\mathcal{D})$, where $\mathcal{L}$ is the number of the layers and $\mathcal{D}$ is the element of every layer of the network. The MLP backward pass can be treated as matrix multiplication with a complexity of $(N/\lambda)O(|\mathcal{B}|LM)$. Therefore, the overall time complexity can be approximated as $O(N(M+2U) + N(HUM + (UM)\log(UM)) + N(HWEU) + (N/\lambda)|\mathcal{B}|\mathcal{L}\mathcal{D})$. In this setting, with most parameters fixed, the time complexity is primarily determined by the neural network structure and the number of training iterations.

E. IRS Phase Shifts Optimization

Given the offloading decision $\boldsymbol{\beta}^*(n)$, the determination of the optimal IRS phase shifts shown as Problem $\mathcal{P}2$ is a non-convex optimization problem. To address this, we follow [32] to employ the Whale Optimization Algorithm (WOA) [41]. WOA is commonly employed to tackle optimization problems such as resource allocations in wireless networks and beyond [42]. In our approach, the WOA algorithm $\boldsymbol{\phi}^*(n) = f_{WOA}(\boldsymbol{\beta}^*(n))$ takes an offloading decision $\boldsymbol{\beta}^*(n)$ as input and produces the best IRS phase shifts $\boldsymbol{\phi}^*(n)$ through $\mathcal{E} = \{1, 2, \ldots, E\}$ evolution rounds, where the hyper-parameter E determines the total number of evolution rounds. Initially, the whale population is represented as $\boldsymbol{\phi}'(0) = \{\boldsymbol{\phi}'_1(0), \boldsymbol{\phi}'_2(0), \ldots, \boldsymbol{\phi}'_W(0)\}$, where the hyper-parameter W determines the number of whales in the environment. The j-th whale, denoted as $\boldsymbol{\phi}'_j(0)$, is a randomly generated IRS phase shift. During the t-th evolution round ($t \in \mathcal{E}$), the following operations are performed. First, we obtain the best IRS phase shift that minimizes the system energy cost. This process can be mathematically formulated as:

$$\boldsymbol{\phi}'_*(t) = \mathop{\arg\min}_{\boldsymbol{\phi}' \in \{\boldsymbol{\phi}'(t-1) \cup \boldsymbol{\phi}'_*(t-1)\}} E^{total}(n)\{\boldsymbol{\phi}' | \boldsymbol{L}, \boldsymbol{\Psi}, \boldsymbol{Z}, \boldsymbol{\beta}^*\},$$

where $E^{total}_u(n)\{\cdot\}$ is (17), $\boldsymbol{\phi}'_*(t)$ denotes the global optimal phase shifts selected in the preceding t iterations. In the case of $t = 1$, we initialize $\boldsymbol{\phi}'_*(0)$ as an empty set, since the global optimal phase shift has not been determined yet. Subsequently, the WOA algorithm employs a balanced probability of 50% to perform either a "spiral route" update or a "shrink-wrap" update. In the event that a "spiral route" update is chosen, the j-th whale within the whale population (i.e., the j-th candidate IRS phase shifts) undergoes the following update procedure:

$$\begin{aligned}
&\boldsymbol{D} = abs(\boldsymbol{\phi}'_*(t) - \boldsymbol{\phi}'_j(t-1)),\\
&\boldsymbol{\phi}'_j(t) = abs(\boldsymbol{D} \cdot e^{b \cdot l_j(t)} \cdot \cos(2\pi \cdot l_j(t)) + \boldsymbol{\phi}'_j(t-1)),
\end{aligned}$$

where $abs(\cdot)$ denotes the element-wise absolute function, b is a constant with a value of 1, and $l_j(t)$ denotes the behavior of the j-th whale during the t-th evolution, which is a random real value between $[-1, 1]$.

In the case of selecting a "shrink-wrap" update, an additional condition check is necessary to determine whether the whale engages in exploration or exploitation. Specifically, if the condition $abs(A_j(t)) < 1$ is satisfied, an *exploitation* step is performed. Conversely, if $abs(A_j(t)) \geq 1$, an *exploration* step is conducted. Here, $A_j(t) = a_j(t) \cdot (2\,r_j(t) - 1)$, where $a_j(t) = 2 \cdot (1 - \frac{t}{E})$ is a scalar that decreases as t increases, and $r_j(t)$ is a randomly generated real value in the range of $[0, 1]$.

In the *Exploitation* phase, the update rule for the j-th whale can be expressed as follows:

$$\begin{aligned}
&\boldsymbol{D} = abs(C_j(t) \cdot \boldsymbol{\phi}'_*(t) - \boldsymbol{\phi}'_j(t-1)),\\
&\boldsymbol{\phi}'_j(t) = abs(\boldsymbol{\phi}'_*(t) - A_j(t) \cdot \boldsymbol{D}),
\end{aligned}$$

where $C_j(t) = 2 \cdot r_j(t)$. In the *Exploration* phase, the update rule for the j-th whale can be defined as:

$$\boldsymbol{D} = abs(C_j(t) \cdot \boldsymbol{\phi}_j^{rand}(t) - \boldsymbol{\phi}'_j(t-1)),$$

$$\boldsymbol{\phi}'_j(t) = abs(\boldsymbol{\phi}_j^{rand}(t) - A_j(t) \cdot \boldsymbol{D}),$$

where $\boldsymbol{\phi}_j^{rand}(t)$ represents a randomly generated IRS phase shifts. Upon completing all E iterations, the resulting IRS phase shifts $\boldsymbol{\phi}'_*(E+1)$ is returned as the final output of WOA.

VI. Experimental Settings

A. Simulation Setup

In conducted experiments we inspired by [19], [32] to set users and UAVs are confined within a rectangular area measuring 800 meters in length and 600 meters in width. The locations of users and UAVs are randomly generated within the designated area, with the UAVs flying at a height of 20 meters. The CPU clock speed of MEC servers carried by UAVs, denoted as Z_m, is distributed between 0.08 and 0.4 GHz. In contrast, the CPU clock speed of UEDs Z_u ranges from 0.04 to 0.08 GHz. The transmission frequency range from 200 to 400 GHz aligns with the THz characteristics outlined in [43] and the molecular absorption coefficients for THz frequencies as indicated in reference [10]. The IRS is composed of 25 reflectors, with the first element located at (4 m, 0 m, 4 m), and $K_x = 5, K_z = 5$. The task size of each user ranges from 32 bytes to 100 KB. The time that users finish their tasks locally is set as the acceptable delay threshold. Any processing time that is longer than this threshold fails to meet Constraint (18f) and is considered as overdue.

B. The Execution of IOPO

We execute IOPO for $N = 200,000$ system time frames, during which the DNN offloading decision model f_θ is trained in a supervised manner. The initial reference offloading decision is generated using the Greedy OC method (introduced in Section VI-C) and the training interval λ is set to 10, indicating that the DNN parameters θ are updated every 10 IOPO execution steps. Furthermore, we utilize a batch size of 256, a dropout rate of 0.1 to mitigate overfitting, a memory buffer size of 1.5 times the batch size, and a learning rate of 0.001in the Adam optimizer. During the execution of IOPO, we set the number of candidate decisions generated in OPPO as $H = 20$. In order to guide OPPO towards identifying decisions that satisfy the no-overdue constraint (defined in (18f)), we introduce an overdue penalty to candidate offloading decisions involving overdue users. Each overdue user adds a penalty score of 100 to the total system energy cost. This prioritizes candidate decisions without overdue users during the selection of the best candidate offloading decision. For the WOA method, the number of whales W is set as 3, while the evolution round E is set as 5.

Following the completion of IOPO execution, we conducted a series of experiments to evaluate its performance compared to several offloading decision-generation baselines. These experiments are carried out over the last 1,000 system time frames and the average metrics (e.g., system energy costs, overdue statistics) are reported. To calculate the system energy costs of different methods, we first acquire a predicted offloading decision from each of the considered offloading decision models. Subsequently, we employ the WOA method denoted as $f_{WOA}(\cdot)$ to derive optimized IRS phase shifts. The optimized IRS phase shift and the obtained offloading decision are substituted into (17), yielding the total energy cost of different offloading decision generation methods.

C. Comparison Offloading Decision Generation Methods

We compare the performance of the proposed IOPO model with baseline offloading allocation approaches as follows:

- *Deep Deterministic Policy Gradient Algorithm (DDPG)*: A model-free reinforcement learning algorithm based on actor-critic architecture. DDPG [26] can be used to generate policies from continuous action spaces. As a strong baseline of one-stage methods, for each time frame, DDPG takes the encoded environment feature as input and then generates an output vector that contains both the offloading decision and the optimal IRS phase.
- *Greedy Selection (Greedy)*: This method utilizes a greedy approach to assign users to UAVs. Specifically, the algorithm iteratively selects the user with the longest local processing time and assigns it to the UAV with the fastest processing speed. After each assignment, the computational speeds of UAVs are updated based on their workload status. This process continues until the fastest UAV processing speed is slower than the slowest local computational speed among the remaining users. The remaining unassigned users finish the tasks locally.
- *Greedy Selection with no-overdue constraint (Greedy OC)*: Similar to the Greedy method, users are ranked based on their local processing times. However, instead of directly assigning each user to the fastest UAV, a more involved iterative process is performed. This process considers all UAVs and selects the UAV that can complete the user's task with the lowest energy cost while ensuring that the time constraints (18f) of all users on that UAV are met. If a suitable UAV cannot be found, the user is assigned to local processing.
- *Local Computing (LOCAL)*: Users independently process tasks on their UEDs without using UAV resources.
- *Optimized Random Selection (OPT RANDOM)*: Users are randomly assigned to either local processing or UAV processing. 10 offloading decisions are randomly generated, and the decision with the lowest energy cost is selected as the final offloading decision.
- *Optimized Random Edge Selection (OPT RANDOM w/o LOCAL)*: Users are randomly assigned to UAVs for task processing. In this case, no user performs tasks locally. Again, 10 offloading decisions are randomly generated, and the decision with the lowest energy cost is chosen.

VII. Experimental Results

A. Model Performance Given Different Numbers of Users

In this experiment, we assess the proposed IOPO model in systems with varying numbers of users. The number of UAVs in systems is fixed at 3. The energy costs of offloading decisions

TABLE II
OVERDUE STATISTICS GIVEN DIFFERENT NUMBERS OF USERS IN THE SYSTEM

	10 USERS		15 USERS		20 USERS	
Methods	*O* Plan%	Avg #*O* Users	*O* Plan%	Avg #*O* Users	*O* Plan%	Avg #*O* Users
Baselines						
LOCAL	0	0	0	0	0	0
GREEDY (OC)	0	0	0	0	0	0
GREEDY	81.76%	1.27	100%	12	100%	12.39
OPT RANDOM	82.46%	3.34	99.94%	8.83	100%	14.49
OPT RANDOM (W/O LOCAL)	97.94%	4.44	100%	11.91	100%	17.41
DDPG	67.90%	2.31	100%	5.48	100%	8.59
Ours						
IOPO	0.86%	1.36	0.6%	1.94	6.88%	1.66

O plan% is the proportion of offloading decisions that contain overdue users and avg #*o* users is the average number of overdue users in overdue offloading decisions.

TABLE III
ENERGY COSTS OF METHODS GIVEN DIFFERENT NUMBERS OF USERS IN THE SYSTEM (WITH OVERDUE PENALTY = 100)

Methods	**10 Users**	**15 Users**	**20 Users**
Baselines			
LOCAL	1048.77	1676.27	2062.25
GREEDY (OC)	508.64	1011.89	1384.11
GREEDY	451.66	1791.93	2030.92
OPT RANDOM	647.64	1540.31	2221.74
OPT RANDOM (W/O LOCAL)	737.47	1728.55	2343.66
DDPG	444.96	1225.47	1640.17
Ours			
IOPO	**397.72**	**823.32**	**1247.98**

TABLE IV
ENERGY COSTS OF METHODS GIVEN DIFFERENT NUMBERS OF USERS IN THE SYSTEM (WITHOUT OVERDUE PENALTY)

Methods	**10 Users**	**15 Users**	**20 Users**
Baselines			
LOCAL	1048.77	1676.27	2062.25
GREEDY (OC)	508.64	1011.89	1384.11
GREEDY	347.48	591.92	791.24
OPT RANDOM	372.08	657.75	771.82
OPT RANDOM (W/O LOCAL)	301.75	**537.17**	**601.82**
DDPG	**290.16**	677.96	781.17
Ours			
IOPO	390.18	819.38	1211.52

predicted by different offloading decision models are presented in Table III. It is observed that the predicted offloading decisions include users who fail to meet their acceptable delay threshold (i.e., fail to meet the Constraint (18f)). As the ideal offloading decisions should minimize energy costs while satisfying the no-overdue constraint (18f), we introduce an overdue penalty to offloading decisions containing overdue users. Specifically, each overdue user adds a penalty score of 100 to the overall system energy cost. By incorporating this overdue-penalized energy cost metric, we are able to evaluate the offloading decisions in terms of both energy costs and the occurrence of overdue users. The results presented in Table III demonstrate that, in comparison to the baselines, the proposed IOPO model achieves the lowest overdue-penalized energy costs across all system configurations. This highlights the effectiveness of IOPO in generating offloading decisions that not only minimize energy consumption but also adhere to the no-overdue constraint (18f).

To gain deeper insights into the overdue situations in offloading decisions generated by various methods, we present the overdue statistics in Table II. The term *O Plans%* represents the percentage of model-predicted offloading decisions that include overdue users, while *Avg #O Users* signifies the average number of overdue users within these overdue decisions. The results reveal that, except for LOCAL and GREEDY (OC), all baseline methods generate a considerable number of offloading decisions containing overdue users. Although LOCAL and GREEDY (OC) adhere to the no-overdue constraint, they fail to fully harness UAV resources to generate energy-efficient offloading decisions (as depicted in Table IV, wherein the overdue penalty is excluded from the system energy cost computation). Consequently, none of the baseline methods can be considered preferable. In contrast, the proposed IOPO framework exhibits the ability to generate offloading allocations with lower energy costs (in comparison to LOCAL and GREEDY (OC)) while significantly reducing the number of overdue users (in comparison to GREEDY, DDPG, and random methods). These findings underscore the effectiveness of the proposed methods over baselines.

B. Model Performance Given Different Numbers of UAVs

In this experiment, we evaluate IOPO in systems with varying numbers of UAVs. The number of users in the system is fixed at 20 and the overdue-penalized energy costs of different methods are reported. Table VI illustrates the overdue-penalized energy costs resulting from offloading allocations generated by different methods. Results show that IOPO consistently outperforms all baseline methods across different system configurations. This underscores IOPO's ability to yield energy-efficient offloading decisions while satisfying the overdue constraint in diverse system setups. Further insights into the overdue statistics are provided in Table V. Once again, the results affirm that IOPO surpasses the baselines GREEDY, DDPG, and RANDOM, while achieving comparable performance to LOCAL and GREEDY (OC) in meeting the no-overdue constraint (18f).

TABLE V
Overdue Statistics of Methods Given Different Numbers of UAVs

	3 UAVS		4 UAVS		5 UAVS	
Methods	O Plan%	Avg #O Users	O Plan%	Avg #O Users	O Plan%	Avg #O Users
Baselines						
Local	0	0	0	0	0	0
Greedy (OC)	0	0	0	0	0	0
Greedy	100%	12.39	100%	16.71	100%	6.07
OPT Random	100%	14.49	100%	12.49	99.90%	9.24
OPT Random (w/o Local)	100%	17.41	100%	15.56	100%	11.52
DDPG	100%	8.59	100%	5.99	100%	4.07
Ours						
IOPO	6.88%	1.66	6.24%	1.91	6.80%	1.86

O plan% denotes the proportion of offloading decisions that contain overdue users and avg #o users denotes the average number of overdue users in overdue offloading decisions. The number of users in the system is set to 20.

TABLE VI
Energy Costs of Methods Given Different Numbers of UAVs in the System (With Overdue Penalty = 100)

Methods	**3UAVs**	**4UAVs**	**5UAVs**
Baselines			
Local	2062.25	2078.15	1779.39
Greedy (OC)	1384.11	1194.84	1009.61
Greedy	2030.92	2235.64	1322.54
OPT Random	2221.74	1874.64	1646.52
OPT Random (w/o Local)	2343.66	2064.96	1800
DDPG	1640.17	1111.7	1038.98
Ours			
IOPO	**1247.98**	**1059.53**	**929.15**

The number of users in the system is set to 20.

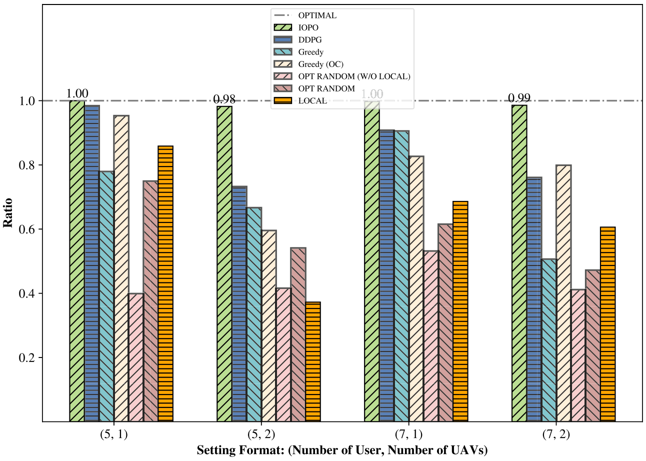


Fig. 4. Average proximity ratio of methods over the last 1,000 time frames.

C. How Good is the Predicted Offloading Decision Compared to the Optimal Decision?

In this experiment, we compare the offloading decisions predicted by IOPO with the optimal offloading decisions. Optimal offloading decisions are determined by considering all possible allocations and selecting the one that minimizes the energy cost while satisfying the no-overdue constraint. We evaluate the performance of IOPO in systems containing (5, 7) users and (1, 2) UAVs. To assess the similarity between the predicted decisions and optimal decisions, we introduce a proximity ratio. This ratio is calculated by dividing the average energy cost of optimal decisions by the average energy cost of predicted offloading decisions. An ideal scenario is indicated by a ratio of 1, signifying that the model-predicted offloading decisions perfectly match the optimal offloading decisions. A ratio smaller than 1 suggests that the energy costs of predicted offloading allocations exceed the optimal energy costs. Therefore, a ratio close to one is desirable, as it indicates a close alignment between the predicted decisions and the optimal decisions. Fig. 4 demonstrates the proximity ratio of IOPO along with 6 baselines under various system settings. Notably, IOPO consistently outperforms all comparison methods, maintaining a proximity ratio close to 1 across all (user, UAV) configurations. These results substantiate that the IOPO-predicted offloading decisions can converge to optimal offloading decisions.

It should be noted that as the number of users and UAVs in the system increases, the number of possible offloading decisions grows exponentially. For instance, in a system with 5 UAVs and 20 users, the total number of potential offloading decisions amounts to $(5+1)^{20}$. This exponential growth makes it impractical to obtain optimal allocations for complex system setups within a reasonable time. Consequently, we focus the investigations on systems with a limited number of users and UAVs. While we do not present optimal solutions for intricate system setups, we observe that increasing the total number of IOPO iterations yields a further reduction in the overall system energy cost. This finding implies that for systems encompassing only a small number of users and UAVs, the IOPO model can converge towards optimal offloading decisions with a relatively small number of IOPO iterations. Conversely, for complex systems involving a larger number of users and UAVs, IOPO necessitates a greater number of iterations to approximate the optimal solution. Therefore, when confronted with systems entailing a significant number of users and UAVs, it is recommended to employ a larger number of iteration steps to attain enhanced outcomes.

D. Ablation Study: How OPPO Affects IOPO Performance

This experiment aims to assess the impact of the proposed OPPO unit on the performance of IOPO. The experimental settings include a penalty of 100 for overdue tasks, 20 users, and

3 UAVs. The evaluation of two variants is based on the average energy cost observed over the last 1,000 system time slots. The two variants considered are IOPO with and without OPPO, taking into account scenarios with the unit disabled during the execution of IOPO and without being disabled. When OPPO is disabled, an alternative approach is needed to quantize the DNN output probability matrix into the offloading decision matrix. To address this, at the n-th time frame, given the DNN predicted probability matrix $\boldsymbol{\mathcal{P}}(n) \in \mathbb{R}^{U\times(M+1)}$, for each user $u \in \mathcal{U}$, we assign a value of 1 to the offloading choice with the largest probability and a value of 0 to the remaining M choices. The resulting offloading decision matrix $\boldsymbol{\beta}(n)$ satisfies Constraints (18a) and (18b). Formally:

$$z' = \underset{z\in\{1,2,\ldots,M+1\}}{\arg\max} \; p_{u,z},$$

$$\beta_{u,m}(n) = \begin{cases} 1 & m = z', \\ 0 & otherwise. \end{cases}$$

The energy cost of IOPO with OPPO is 1247.98, whereas without OPPO is 1408.36. This demonstrates that the inclusion of OPPO significantly reduces the overdue-penalized system energy cost when compared to the variant without OPPO.

Besides, we analyze the impact of removing OPPO on overdue cases in IOPO. Surprisingly, IOPO without OPPO outperformed IOPO with OPPO, significantly reducing overdue decisions and users. With OPPO, there was a 6.88% occurrence of overdue plans, compared to 0.94% without OPPO. Moreover, despite higher penalties, IOPO with OPPO achieved lower energy costs for overdue tasks than the variant without OPPO. The reason behind these findings can be attributed to the challenge lies in creating efficient offloading allocations using UAV computational power while adhering to the no-overdue constraint. The variant without OPPO showed limited user offloading to UAVs, while the variant with OPPO underutilized UAV capabilities. IOPO systematically improved initial decisions with OPPO, leading to more overdue cases with a slight increase in users per UAV. Still, IOPO had fewer overdues, achieving lower energy costs despite predicting more overdue cases.

During IOPO execution, OPPO continually explored improved decisions, generating 127,966 during 200,000 iterations. The DNN learned from these decisions, reducing the overdue-penalized energy cost to 1247.98 compared to 1384.57 for initial decisions. Notably, the initial offloading decisions, generated using the Greedy method with a no-overdue constraint, didn't have overdue users. The decrease in energy cost resulted from OPPO's ability to optimize task distribution between users and UAVs. In summary, results demonstrate the efficacy of OPPO in generating a substantial quantity of improved offloading decisions and reducing the system energy costs.

E. Does the Initial Reference Offloading Decision Help?

In this experiment, we study if applying initial reference offloading decisions benefits the performance of IOPO. The introduction of initial offloading decisions aims to establish a favorable starting point for training the DNN in IOPO. Without the provision of initial reference offloading decisions, the DNN may learn from suboptimal offloading decisions during the early stages of IOPO execution, thereby slowing the convergence towards optimal offloading allocations and resulting in impaired IOPO performance. This issue could become particularly pronounced when dealing with a large solution space due to the increasing difficulty in identifying high-quality offloading decisions for training the DNN. Consequently, the inclusion of initial reference offloading allocations can play a critical role in guiding the training of DNN and reducing the energy costs of IOPO-predicted offloading decisions.

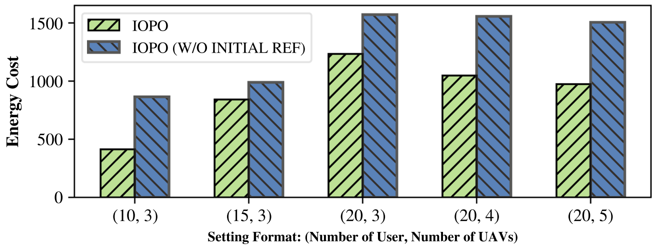


Fig. 5. IOPO performance with and without utilizing initial reference offloading decisions during the training of DNN. The overdue penalty is set to 100 in system energy cost computation.

Fig. 5 presents the average overdue-penalized energy costs over the last 1,000 system time frames. When the initial reference offloading decisions are not provided during DNN training, we set the predicted offloading decisions generated using (19) as reference to offloading decisions. Results demonstrate that, compared to the variant *IOPO (W/O INITIAL REF)*, in which initial reference offloading decisions are excluded in DNN training, *IOPO* can produce offloading decisions with lower energy costs. These findings align with the intuition and emphasize the significance of supplying high-quality initial reference decisions during DNN training to achieve reduced system energy consumption.

F. Does DNN Complexity Affect IOPO Performance?

In this experiment, we study the influence of DNN complexity on the performance of IOPO. Table VII presents the performance of IOPO equipped with two DNNs: the proposed DNN (*Ours*) and a DNN with reduced complexity (*Simplified*). Compared to *Ours*, the downgraded network consists of 1 hidden layer instead of 6 and 64 hidden units instead of 256. Results indicate that the downgraded DNN (*Simplified*) exhibits higher overdue-penalized energy cost (*Eng Cost*) in all tested settings compared to the sophisticated DNN (*Ours*). This outcome can be attributed to the subpar performance of the simplified DNN in producing high-quality probability matrices. As the offloading decisions predicted by the IOPO are derived from the DNN probability matrix, sub-optimal probability matrices generated from *Simplified* result in predicted offloading decisions that incur higher energy costs. Moreover, a reduced number of improved offloading decisions discovered by OPPO (*#Improved*) is observed in the downgraded model. These findings suggest that DNN complexity has a significant impact on the final system energy cost and the performance of OPPO searching.

TABLE VII
MODEL PERFORMANCE AND OPPO STATISTICS WITH DIFFERENT DNN COMPLEXITY (OVERDUE PENALTY IS 100 IN SYSTEM ENERGY COST)

	10 USERS 3 UAVS		15 USERS 3 UAVS		20 USERS 3 UAVS		20 USERS 4 UAVS		20 USERS 5 UAVS	
Metrics	Ours	Simplified	Ours	Simplified	Ours	Simplified	Ours	Simplified	Ours	Simplified
Eng Cost	**393.34**	424.43	**841.49**	912.33	**1233.76**	1306.16	**1047.57**	1118.58	**953.45**	1044.69
#Improved	**146505**	102555	**143939**	105877	**126177**	102803	**122078**	101471	**115477**	85720

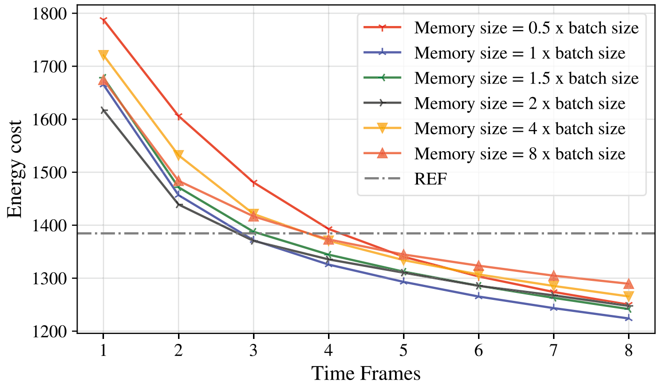


Fig. 6. Impact of memory buffer size on system energy cost. Each time frame represents the average energy costs of over 25,000 IOPO execution steps.

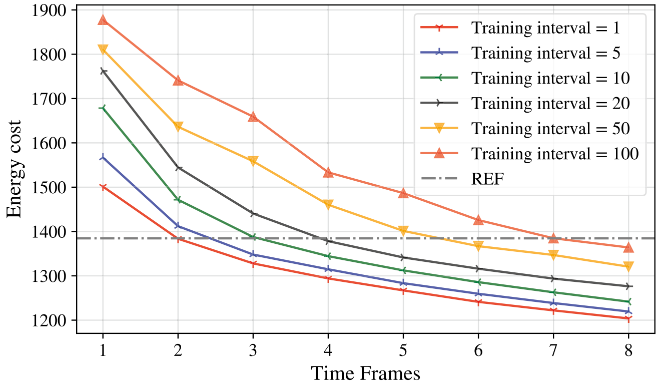


Fig. 7. Impact of Training Interval size on energy cost. Each time frame represents the average energy cost of over 25,000 IOPO execution steps.

TABLE VIII
IOPO PERFORMANCE WITH DIFFERENT MEMORY SIZES
(WITH OVERDUE PENALTY = 100)

Memory Size	**Eng Cost**	**#Improved**
0.5 batch size	1256.82	121689
1 batch size	**1232.28**	**131871**
1.5 batch size	1253.76	124038
2 batch size	1273.86	121413
4 batch size	1285.07	117858
8 batch size	1294.34	111396

G. Model Analysis: Memory Buffer Size

In this experiment, we investigate the influence of memory buffer size on the performance of IOPO. The number of users in the system is set to 20, and the number of UAVs is set to 3. Fig. 6 shows the overdue-penalized energy costs of offloading decisions predicted by IOPO during the entire IOPO execution. The REF horizontal line represents the average energy cost of the initially provided reference offloading decisions. As depicted in Fig. 6, IOPO with various memory sizes outperforms the REF offloading decisions as the iteration progresses. This improvement is attributed to the OPPO unit in IOPO, which can discover offloading decisions with low energy costs as the IOPO execution progresses. Moreover, IOPO with a memory size equal to the batch size demonstrates the lowest energy cost by the end of IOPO execution, compared to other memory size configurations. To provide a comprehensive understanding of the impact of memory size, Table VIII presents the average overdue-penalized energy costs (*Eng Cost*) over the last 1,000 system time frames and the number of IOPO-predicted offloading decisions that surpass the initially provided reference offloading decisions (*#Improved*). Results indicate that the optimal IOPO performance is achieved when the memory size aligns with the batch size, with the lowest test energy cost recorded as 1232.28 and the largest number of improved allocations discovered as 131,871. These findings highlight the significance of aligning the memory size with the size of training batches for optimal IOPO performance.

When considering alternative memory sizes, we observe slightly higher system energy costs and smaller numbers of offloading decisions discovered compared to the optimal configuration. Additionally, as the memory size becomes larger, the overall energy cost increases. This phenomenon can be attributed to the difficulty of sampling the most recently improved offloading decisions from a substantial historical pool when training the DNN. As a result, the DNN may acquire knowledge from sub-optimal historical data, leading to compromised performance and heightened energy consumption in IOPO-predicted offloading decisions.

H. Model Analysis: Training Interval

In this experiment, we examine the impact of the size of the training interval λ on the performance of IOPO. The number of users in the system is set to 20 and the number of UAVs is set to 3. Fig. 7 illustrates the overdue-penalized energy cost of IOPO-predicted decisions and REF denotes the average energy cost of the initial reference offloading decisions.

As shown in Fig. 7, IOPO with different training interval sizes (1, 5, 10) can yield offloading decisions with similar and low energy costs after IOPO execution. When the training interval size is increased to 50 and 100, the resulting decisions exhibit higher energy costs. Moreover, the energy costs of IOPO with training intervals 50 and 100 are closer to the horizontal REF line, indicating a compromised performance of the OPPO unit in discovering improved offloading decisions when the training interval is large. This is because, with large training intervals, the parameters θ of the DNN offloading decision model f_θ are updated less frequently. Consequently, the accuracy of the DNN is

TABLE IX
IOPO PERFORMANCE WITH VARIOUS TRAINING INTERVALS
(WITH OVERDUE PENALTY = 100)

Training Interval	Eng Cost	#Improved
1	**1196.84**	**144841**
5	1203.57	137763
10	1253.76	124038
20	1277.90	118099
50	1324.63	86867
100	1370.78	50734

compromised, causing the predicted offloading decisions, which rely on the DNN-output probability matrix, to be impaired.

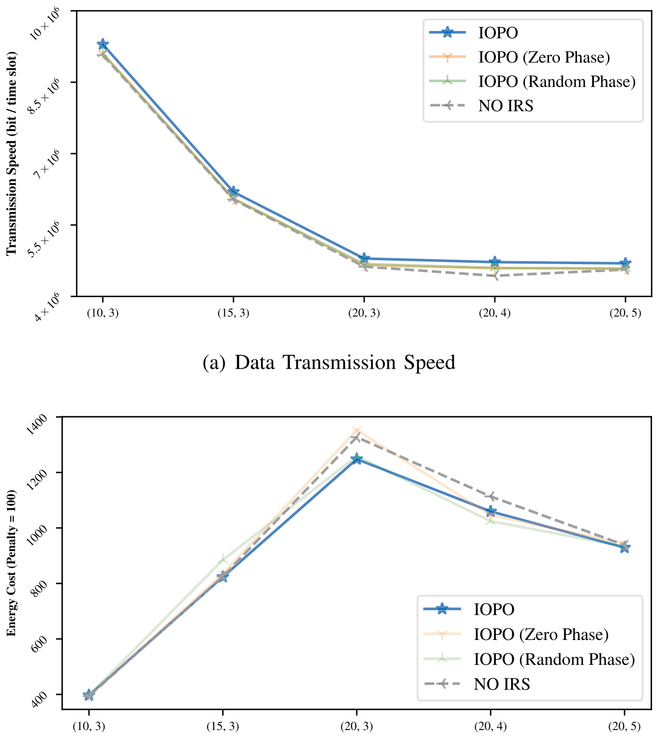
Table IX demonstrates that the lowest system energy cost achieved is 1196.84, and the largest number of improved decisions discovered is 144841, both obtained when the training interval is set to 1. This is because a small training interval facilitates the update of DNN parameters and the improvement of DNN performance. With the continual improvement of the DNN, there is a corresponding enhancement in the IOPO-predicted offloading allocations that depend on the DNN's performance. Subsequently, the DNN learns from these improved offloading decisions, leading to further enhancements in its own performance and a reduction in energy costs of IOPO-predicted decisions. However, it is important to note that using a smaller training interval may result in slower system speed due to the increased frequency of DNN parameter updates. If execution speed is a primary concern, it is reasonable to consider setting the training interval to 5 or 10, as these interval sizes yield energy costs that are close to the energy cost achieved with a training interval of 1.

I. Model Analysis: Impact of IRS

In this experiment, we explore the influence of IRS on both data transmission speed and system energy consumption. Specifically, we conduct a comparative analysis involving the proposed IOPO framework against three distinct variants: (i) NO IRS, wherein the IRS board is excluded; (ii) IOPO (ZERO PHASE), denoting a configuration where the phase shift of all IRS reflecting elements is set to 0; and (iii) IOPO (RANDOM PHASE), where the phases shift of IRS reflecting elements are randomly assigned.

As depicted in Fig. 8(a), shows that IOPO consistently achieves superior data transmission speeds when compared to all three variants. Fig. 8(b), the removal of IRS from the system is observed to result in escalated energy consumption. Results demonstrate the efficacy of the IRS in reducing system energy consumption while augmenting data transmission rates. Moreover, IOPO consistently demonstrates reduced energy costs compared to both IOPO (RANDOM PHASE) and IOPO (ZERO PHASE) configurations in scenarios involving (15 users and 3 UAVs) and (20 users and 3 UAVs), while maintaining comparable energy consumption across other settings.

The trend of the lines in Fig. 8(b) indicates an increase in energy cost up to the point of (20 users and 4 UAVs). This is because the UAVs function as MEC servers. When the number of users increases while the computing resources remain constant,



(a) Data Transmission Speed

(b) System Overdue-Penalized Energy Cost

Fig. 8. Impact of IRS on system energy and data transmission speed. The x-axis denotes (the number of users, and the number of UAVs) in the system.

the total system cost rises. The increased user demand for the same resources leads to a higher average task allocation per UAV, resulting in higher energy costs. Adding more UAVs after this point alleviates the system's computing burden and reduces energy costs.

Although the energy consumption differences might appear less significant at certain points, IOPO consistently demonstrates superior energy efficiency in most scenarios, making it a more stable optimization than other variants. The less noticeable differences are due to the high transmission speeds under the THz network. Once the speed reaches a certain threshold, further improvements have a less pronounced effect on latency. At the point of (20 users and 3 UAVs), when resources are scarce, the benefits of optimizing IRS phase shifts to enhance channel gain become more apparent. In conclusion, the results highlight the advantages of incorporating IRS and optimizing its phase shift using IOPO over simplistic configurations such as uniformly zeroed or randomly assigned phase shifts.

VIII. CONCLUSION

In this study, we investigate the task offloading problems in a multi-user multi-UAV MEC system that integrates an IRS and operates on the THz communication network. We present the modeling of the task offloading and the task processing procedure of the MEC system within the THz network and introduce IOPO, a novel deep learning-based framework designed to optimize the energy efficiency of task offloading decisions and the phase shifts of the IRS. The IOPO framework can generate satisfactory offloading decisions within milliseconds and is incorporated with a novel offloading decision-searching unit OPPO, enabling continuous search to identify improved offloading allocations. Extensive experimental results demonstrate the superiority of IOPO over baseline methods in generating energy-efficient offloading allocations and meeting task deadlines.

In the future, several directions exist to extend this work. First, the algorithm's performance can be trained and evaluated in a realistic system (e.g., real THz data transmission environments, practical UAV energy losses, and real-world computational tasks) to improve the algorithm's robustness and applicability in practical scenarios. Second, the IOPO's performance can be further enhanced by optimizing the second-stage algorithm. Third, the proposed model can be extended to multiple base stations, encompassing wider areas and more UAVs and UEDs.

References

[1] Z. Yang, S. Bi, and Y.-J. A. Zhang, "Online trajectory and resource optimization for stochastic UAV-enabled MEC systems," *IEEE Trans. Wireless Commun.*, vol. 21, no. 7, pp. 5629–5643, Jul. 2022.

[2] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey, and C. S. Hong, "Energy-efficient resource management in UAV-assisted mobile edge computing," *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 249–253, Jan. 2021.

[3] Z. Chen, H. Zheng, J. Zhang, X. Zheng, and C. Rong, "Joint computation offloading and deployment optimization in multi-UAV-enabled MEC systems," *Peer-to-Peer Netw. Appl.*, vol. 15, pp. 194–205, 2022.

[4] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. Leung, "Joint trajectory and computation offloading optimization for UAV-assisted MEC with NOMA," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2019, pp. 1–6.

[5] L. Zhang et al., "Task offloading and trajectory control for UAV-assisted mobile edge computing using deep reinforcement learning," *IEEE Access*, vol. 9, pp. 53708–53719, 2021.

[6] J. Xue, Q. Wu, and H. Zhang, "Cost optimization of UAV-MEC network calculation offloading: A multi-agent reinforcement learning method," *Ad Hoc Netw.*, vol. 136, 2022, Art. no. 102981.

[7] F. Zhou, Y. Wu, H. Sun, and Z. Chu, "UAV-enabled mobile edge computing: Offloading optimization and trajectory design," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–6.

[8] P. A. Apostolopoulos, G. Fragkos, E. E. Tsiropoulou, and S. Papavassiliou, "Data offloading in UAV-assisted multi-access edge computing systems under resource uncertainty," *IEEE Trans. Mobile Comput.*, vol. 22, no. 1, pp. 175–190, Jan. 2023.

[9] H. Elayan, O. Amin, R. M. Shubair, and M.-S. Alouini, "Terahertz communication: The opportunities of wireless technology beyond 5G," in *Proc. Int. Conf. Adv. Commun. Technol. Netw.*, 2018, pp. 1–5.

[10] A.-A. A. Boulogeorgos, E. N. Papasotiriou, and A. Alexiou, "A distance and bandwidth dependent adaptive modulation scheme for THz communications," in *Proc. IEEE 19th Int. Workshop Signal Process. Adv. Wireless Commun.*, 2018, pp. 1–5.

[11] C. Pan et al., "Intelligent reflecting surface aided MIMO broadcasting for simultaneous wireless information and power transfer," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 8, pp. 1719–1734, Aug. 2020.

[12] T. Bai, C. Pan, C. Han, and L. Hanzo, "Reconfigurable intelligent surface aided mobile edge computing," *IEEE Wireless Commun.*, vol. 28, no. 6, pp. 80–86, Dec. 2021.

[13] M. Ahmed et al., "Joint optimization of UAV-IRS placement and resource allocation for wireless powered mobile edge computing networks," *J. King Saud Univ.- Comput. Inf. Sci.*, vol. 35, no. 8, 2023, Art. no. 101646.

[14] C. Zhao, X. Pang, W. Lu, Y. Chen, N. Zhao, and A. Nallanathan, "Energy efficiency optimization of IRS-assisted UAV networks based on statistical channels," *IEEE Wireless Commun. Lett.*, vol. 12, no. 8, pp. 1419–1423, Aug. 2023.

[15] Y. Zhang, J. Li, G. Mu, and X. Chen, "Deep reinforcement learning enabled UAV-IRS-assisted secure mobile edge computing network," *Phys. Commun.*, vol. 61, 2023, Art. no. 102173.

[16] E. T. Michailidis, N. I. Miridakis, A. Michalas, E. Skondras, and D. J. Vergados, "Energy optimization in dual-RIS UAV-aided MEC-enabled internet of vehicles," *Sensors*, vol. 21, no. 13, 2021, Art. no. 4392.

[17] Q. Liu, J. Han, and Q. Liu, "Joint task offloading and resource allocation for RIS-assisted UAV for mobile edge computing networks," in *Proc. IEEE/CIC Int. Conf. Commun. China*, 2023, pp. 1–6.

[18] M. A. ElMossallamy, H. Zhang, L. Song, K. G. Seddik, Z. Han, and G. Y. Li, "Reconfigurable intelligent surfaces for wireless communications: Principles, challenges, and opportunities," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 3, pp. 990–1002, Sep. 2020.

[19] Y. Pan, K. Wang, C. Pan, H. Zhu, and J. Wang, "Sum-rate maximization for intelligent reflecting surface assisted terahertz communications," *IEEE Trans. Veh. Technol.*, vol. 71, no. 3, pp. 3320–3325, Mar. 2022.

[20] W. Chen, X. Ma, Z. Li, and N. Kuang, "Sum-rate maximization for intelligent reflecting surface based terahertz communication systems," in *Proc. IEEE/CIC Int. Conf. Commun. Workshops China*, 2019, pp. 153–157.

[21] C. Chaccour, M. N. Soorki, W. Saad, M. Bennis, and P. Popovski, "Risk-based optimization of virtual reality over terahertz reconfigurable intelligent surfaces," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[22] C. Chaccour, M. N. Soorki, W. Saad, M. Bennis, and P. Popovski, "Risk-based optimization of virtual reality over terahertz reconfigurable intelligent surfaces," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–6.

[23] Y. Pan, K. Wang, C. Pan, H. Zhu, and J. Wang, "UAV-assisted and intelligent reflecting surfaces-supported terahertz communications," *IEEE Wireless Commun. Lett.*, vol. 10, no. 6, pp. 1256–1260, Jun. 2021.

[24] S. Li, B. Duo, X. Yuan, Y.-C. Liang, and M. Di Renzo, "Reconfigurable intelligent surface assisted UAV communication: Joint trajectory design and passive beamforming," *IEEE Wireless Commun. Lett.*, vol. 9, no. 5, pp. 716–720, May 2020.

[25] S. Ahmad, S. Khan, K. S. Khan, F. Naeem, and M. Tariq, "Resource allocation for IRS-assisted networks: A deep reinforcement learning approach," *IEEE Commun. Standards Mag.*, vol. 7, no. 3, pp. 48–55, Sep. 2023.

[26] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[27] L. Huang, S. Bi, and Y.-J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 11, pp. 2581–2593, Nov. 2020.

[28] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4692–4707, Oct. 2019.

[29] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Performance optimization in mobile-edge computing via deep reinforcement learning," in *Proc. IEEE 88th Veh. Technol. Conf.*, 2018, pp. 1–6.

[30] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for IoT devices with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.

[31] 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 of Things J.*, vol. 7, no. 7, pp. 6252–6265, Jul. 2020.

[32] Y. M. Park, S. S. Hassan, Y. K. Tun, Z. Han, and C. S. Hong, "Joint resources and phase-shift optimization of MEC-enabled UAV in IRS-assisted 6G THz networks," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp.*, 2022, pp. 1–7.

[33] X. Wang et al., "Wireless powered mobile edge computing networks: A survey," *ACM Comput. Surv.*, vol. 55, no. 3, pp. 263:1–263:37, 2023.

[34] M. Wu, W. Qi, J. Park, P. Lin, L. Guo, and I. Lee, "Residual energy maximization for wireless powered mobile edge computing systems with mixed-offloading," *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 4523–4528, Apr. 2022.

[35] T. Zhu, J. Li, Z. Cai, Y. Li, and H. Gao, "Computation scheduling for wireless powered mobile edge computing networks," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 596–605.

[36] X. Cao, F. Wang, J. Xu, R. Zhang, and S. Cui, "Joint computation and communication cooperation for energy-efficient mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4188–4200, Jun. 2019.

[37] O. Maraqa, S. Al-Ahmadi, A. S. Rajasekaran, H. U. Sokun, H. Yanikomeroglu, and S. M. Sait, "Energy-efficient optimization of multi-user NOMA-assisted cooperative THz-SIMO MEC systems," *IEEE Trans. Commun.*, vol. 71, no. 6, pp. 3763–3779, Jun. 2023.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014.

[39] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017, *arXiv:1412.6980*.
[40] Z. Tang, J. Lou, and W. Jia, “Layer dependency-aware learning scheduling algorithms for containers in mobile edge computing,” *IEEE Trans. Mobile Comput.*, vol. 22, no. 6, pp. 3444–3459, Jun., 2023.
[41] S. Mirjalili and A. Lewis, “The whale optimization algorithm,” *Adv. Eng. Softw.*, vol. 95, pp. 51–67, 2016.
[42] Q.-V. Pham, S. Mirjalili, N. Kumar, M. Alazab, and W.-J. Hwang, “Whale optimization algorithm with applications to resource allocation in wireless networks,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4285–4297, Apr. 2020.
[43] H.-J. Song and T. Nagatsuma, “Present and future of terahertz communications,” *IEEE Trans. THz Sci. Technol.*, vol. 1, no. 1, pp. 256–263, Sep. 2011.

Jianqiu Wu received the MS degree from the Faculty of Engineering, Chinese University of Hong Kong, in 2018. She is currently working toward the MPhil degree with the Department of Computer Science, BNU-HKBU United International College, Zhuhai, China. She is supervised by Dr. Jianxiong Guo, and her research interests include reinforcement learning, mobile edge computing, and deep learning in wireless communications.

Zhongyi Yu received the BS degree from the Department of Computer Science, BNU-HKBU United International College, Zhuhai, China, in 2020, and the MS degree from the School of Informatics, University of Edinburgh, Edinburgh, U.K., in 2022. His research interests include reinforcement learning, natural language processing, causal inference, and efficient machine learning.

Jianxiong Guo (Member, IEEE) received the BE degree from the School of Chemistry and Chemical Engineering, South China University of Technology, Guangzhou, China, in 2015, and the PhD degree from the Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA, in 2021. He is currently an associate professor with the Advanced Institute of Natural Sciences, Beijing Normal University, and also with the Guangdong Key Lab of AI and Multi-Modal Data Processing, BNU-HKBU United International College, Zhuhai, China. He is a member of the ACM/CCF. He has published more than 80 peer-reviewed papers and been the reviewer for many famous international journals/conferences. His research interests include social networks, wireless sensor networks, combinatorial optimization, and machine learning.

Zhiqing Tang (Member, IEEE) received the BS degree from the School of Communication and Information Engineering, University of Electronic Science and Technology of China, China, in 2015, and the PhD degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China, in 2022. He is currently an assistant professor with the Advanced Institute of Natural Sciences, Beijing Normal University, China. His current research interests include edge computing, resource scheduling, and reinforcement learning.

Tian Wang (Senior Member, IEEE) received the BSc and MSc degrees in computer science from the Central South University, in 2004 and 2007, respectively, and the PhD degree in computer science from the City University of Hong Kong, in 2011. Currently, he is a professor with the Institute of Artificial Intelligence and Future Networks, Beijing Normal University. His research interests include the Internet of Things, edge computing, and mobile computing. He has more than 15000 citations, according to Google Scholar. His H-index is 71.

Weijia Jia (Fellow, IEEE) received the BSc and MSc degree from Center South University, China, in 1982 and 1984, and the master of applied science and PhD degrees from the Polytechnic Faculty of Mons, Belgium, in 1992 and 1993, respectively, all in computer science. He is currently a chair professor, director of BNU-UIC Institute of Artificial Intelligence and Future Networks, Beijing Normal University (Zhuhai) and VP for Research of BNU-HKBU United International College (UIC) and has been the Zhiyuan chair professor of Shanghai Jiao Tong University, China. He was the chair professor and the deputy director with the State Kay Laboratory of Internet of Things for Smart City, University of Macau. From 1993–1995, he joined German National Research Center for Information Science (GMD) in Bonn (St. Augustine) as a research fellow. From 1995–2013, he worked with the City University of Hong Kong as a professor. His contributions have been recognized as optimal network routing and deployment; anycast and QoS routing, sensors networking, AI (knowledge relation extractions; NLP, etc.), and edge computing. He has more than 600 publications in the prestige international journals/conferences and research books and book chapters. He has received the best product awards from the International Science & Tech. Expo (Shenzhen) in 2011–2012 and the 1st Prize of Scientific Research Awards from the Ministry of Education of China in 2017 (list 2). He has served as area editor for various prestige international journals, chair, PC member, and keynote speaker for many top international conferences. He is the distinguished member of CCF.