GENERAL INSTRUCTION

- **Authors: Please check and confirm whether the name of the corresponding author is correct as set.**
- **Authors: When you submit your corrections, please either annotate the IEEE Proof PDF or send a list of corrections. Do not send new source files as we do not reconvert them at this production stage.**
- **Authors: Carefully check the page proofs (and coordinate with all authors); additional changes or updates WILL NOT be accepted after the article is published online/print in its final form. Please check author names and affiliations, funding, as well as the overall article for any errors prior to sending in your author proof corrections. Your article has been peer reviewed, accepted as final, and sent in to IEEE. No text changes have been made to the main part of the article as dictated by the editorial level of service for your publication.**
- **Authors: Per IEEE policy, one complimentary proof will be sent to only the Corresponding Author. The Corresponding Author is responsible for uploading one set of corrected proofs to the IEEE Author Gateway.**

QUERIES

Q1. Author: Please confirm or add details for any funding or financial support for the research of this article.

Q2. Author: Refs. [21] and [31] are identical. We have deleted [31]. Please check and confirm whether the renumbered references are correct as set.

DRL-Based Joint Optimization of Wireless Charging and Computation Offloading for Multi-Access Edge Computing

Xinyuan Zhu, Fei Hao, Lianbo Ma, Changqing Luo, *Member, IEEE*, Geyong Min, and Laurence T. Yang

***Abstract*—Wireless-powered multi-access edge computing (WP-MEC), as a promising computing paradigm with the great potential for breaking through the power limitations of wireless devices, is facing the challenges of reliable task offloading and charging power allocation. Towards this end, we formulate a joint optimization problem of wireless charging and computation offloading in socially-aware D2D-assisted WP-MEC to maximize the utility, characterized by wireless devices' residual energy and the strength of social relationship. To address this problem, we propose a deep reinforcement learning (DRL)-based approach with hybrid actor-critic networks including three actor networks and one critic network as well as with Proximal Policy Optimization (PPO) updating policy. Further, to prevent the policy collapse, we adopt the PPO-clip algorithm which limits the update steps to enhance the stability of algorithm. The experimental results show that the proposed algorithm can achieved superior convergence performance and, meanwhile, improves the average utility efficiently compared to other baseline approaches.**

***Index Terms*—Multi-access edge computing, wireless charging, computation offloading, deep reinforcement learning, proximal policy optimization.**

I. Introduction

A. Motivation

To cope with the issue of network congestion arising from the surge in network traffic, Multi-access Edge Computing (MEC) [1], [2] offers a solution by alleviating network burden. By offloading computation-intensive tasks from resource-limited edge devices to neighboring MEC servers, MEC effectively reduces both computational latency and energy consumption [3]. In addition, portable edge devices are also limited by battery life and computing power [4], and with the advancement of wireless communication technology and Wireless Power Transmission (WPT) [5], [6], edge wireless devices can be charged wirelessly without the necessity for battery replacement. To provide wireless devices with continuous energy supply and enhance the computing capacity, more and more studies have considered combining WPT with MEC, leading to a new framework called wireless powered multi-access edge computing (WP-MEC) [7], [8].

WP-MEC enables devices to process data without wired charging and can be used in networks requiring low latency and high computing power, such as driverless cars, smart sensor networks, and real-time data analysis applications. However, WP-MEC still encounters several challenges. In WP-MEC, it is very challenging to make offloading decisions and power allocation to reduce the energy consumption of the system and improve the network efficiency [9], [10]. And as abundant personal private data (such as personal images, personal physical health information and so on) transmitting and handling in WP-MEC networks, it is important to guarantee the reliable and efficient offloading as well as reasonable wireless charging allocation in WP-MEC network [11]. Unreasonable task offloading decisions and charging allocation will result in personal privacy disclosure, excessive energy consumption or lead to improper task execution. Socially-aware D2D task offloading schedule is an effective solution for privacy protection, reducing energy and processing resource requirements by mapping the social relationship in social domain to the relationship between devices in MEC. Therefore, it is significant to combine socially-aware D2D communication with wireless charging to select an appropriate offloading strategy and charging scheme in WP-MEC [12], [13].

What's more, socially aware D2D-Assisted wireless powered MEC combines social relationships, D2D communication, and wireless charging technologies to optimize resource allocation and computing power. What's more, it has certain practical application significance, and can be applied to UAV networks, intelligent transportation system, personal health monitoring, virtual reality or augmented reality applications and other realistic scenarios. For example, in an intelligent transportation system, the vehicle can select the best D2D communication partner based on the owner's social relationship, conduct real-time data processing, road condition prediction and task offloading,

Received 31 January 2024; revised 18 March 2025; accepted 20 March 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62477029 and Grant 61702317 and in part by the Ministry of Education Humanities and Social Sciences Research Youth Fund Project under Q1 Grant 22YJCZH046. *(Corresponding author: Fei Hao.)*

Xinyuan Zhu and Fei Hao are with the School of Computer Science, Shaanxi Normal University, Xi'an 3710119, China (e-mail: feehao@gmail.com).

Lianbo Ma is with the School of Software, Northeastern University, Shenyang 100819, China.

Changqing Luo is with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23084 USA.

Geyong Min is with the Department of Computer Science, University of Exeter, EX4 4QF Exeter, U.K.

Laurence T. Yang is with the School of Computer Science and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China, and also with the Department of Computer Science, St. Francis Xavier University, Antigonish B2G 2W5, Canada.

Digital Object Identifier 10.1109/TSC.2025.3556614

and use wireless charging technology to charge the vehicle equipment during driving [14].

Meanwhile, effectively addressing the above joint optimization problem is also challenging. Due to the advantages of DRL, more and more researchers pay attention to using DRL to address the problems in MEC [3], [15], [16], [17]. Most of the existing studies consider a single action space like discrete action [18] or continuous action [19]. Some focus on the discrete offloading decisions or selections by using Q-learning based algorithm [20], and others work out the continuous problems like resource allocation and power allocation by using the deep deterministic policy gradient (DDPG) based algorithms [3], [21]. However, these existing approaches focus on single discrete or continuous action, which are unsuitable for solving most MEC problems that require hybrid actions. And these unsuitable approaches will cause higher algorithm complexity for action transformation even the failure of action generation. Although some research works focus on the hybrid action space [3], [22], [23], these existing approaches have poor stability and convergence for the action transformation and policy updates. PPO or other methods can further optimize the algorithmic framework they used, which will make algorithm more stable and easier to converge by limiting the magnitude of policy updates.

B. Our Contributions

In order to deal with the above challenges, in this paper, we elaborate and model reliable partial task offloading under wireless charging MEC networks and jointly optimize the computation offloading and wireless charging in WP-MEC by using DRL method which is based on the PPO update policy. The major contributions of this paper are summarized as follows.

- *Problem Formulation:* We jointly optimize the task offloading, task allocation, and channel power allocation to maximize the utility of social relationships and device residual energy in the MEC network. To solve the above joint optimization problem by using DRL-based method, we further define state space, action space, and reward function, which is formalized as a Markov Decision Process (MDP).
- *DRL-based method using hybrid PPO:* Since our action space is consist of hybrid actions, including one discrete task offloading action and two continuous actions: task assignment and the power of channels allocation, our DRL-based hybrid actor-critic framework includes one critic network and three actor networks: offloading decisions network, task assignment network and power allocation network, which are updated and improved by using the hybrid proximal policy optimization (PPO). On the basis of traditional hybrid PPO methods, our method introduces adaptive learning rate and dynamic importance sampling to overcome the limitation of convergence, stability and efficiency of traditional hybrid PPO method.
- *Simulation experiments and performance evaluation:* To assess the efficacy of our proposed algorithm, we conduct extensive experiments to evaluate its performance. We carry out convergence analysis by comparing the reward of our approach with the other DRL methods and under different network parameters. Then we conduct experiments under different methods and different number of three kinds of devices in MEC with parameters changed in MEC. Our experimental results demonstrate that our approach outperforms compared with other baseline approaches in terms of reward and average utility.

C. Outline

The remainder of this paper is organized as follows. Section II presents the related work. The system model and optimization problem of this paper are illustrated in Section III. In Section IV, the DRL-based approach that are settled with the joint optimization problem is described. The simulation experiments and evaluation results of the proposed approach are described in V. Finally, Section VI makes a conclusion of this work and provides the directions for future research.

II. Related Work

In recent years, the task offloading strategy selection is widely investigated for energy conservation and improving network efficiency. It follows that DRL is more and more widely used to solve such problems effectively.

A. Wireless Charging and Task Offloading in MEC Network

Recently, most of the MEC network devices involved in research has been limited by its own power. Therefore, it is necessary to consider the energy consumption and its own power as well as wireless charging when offloading tasks [13], [24].

Wang et al. [25] focused on a wireless powered multi-user MEC system and each user relying on the collected energy depends on the collected energy for performing computing tasks. For minimizing the energy consumption, the authors jointly optimized energy beam forming and task offloading. Furthermore, aiming at minimizing the energy consumption in delay constraints in MEC, Malik et al. [9] proposed a combined wireless charging solution based on computational offloading. Wu et al. [26] elaborated a WP-MEC system and proposed a user collaboration approach to improve the performance of active devices, where nearby available devices help remotely perform the computing tasks of the user by using the wireless energy harvested from the energy transmitter. In [27], the study investigated the concept of secure offloading in a WP-MEC system. In this system, energy-constrained users are charged using wireless power transfer, and the harvested energy is harnessed to offload their computing tasks to the MEC server, even in the presence of multiple eavesdroppers. Wang et al. [28] proposed a unified MEC-WPT design, each user node rely on collecting energy computing tasks and users can perform their own tasks locally or offload tasks to MEC completely or partially based on TDMA protocols. Zheng et al. [29] studied a wireless-powered MEC network to minimize total computation delay (TCD), which is decomposed into optimizing WPT and transmission durations and offloading decisions, and a worst-WD-adjusting (WDA) algorithm and a DRL-based model efficiently achieve near-optimal solutions, demonstrating effectiveness in fast-fading networks.

Although there are many researches above that have focused on the joint optimization problem of wireless charging and the computational offloading in MEC networks such as [9], [28], our proposed optimization problem differs from the above works in terms of partial offloading, D2D offloading strategy, or social relationships in MEC networks. Existing studies about joint optimization of wireless charging and task offloading mostly focus on the separative optimization instead of synchronous optimization of wireless charging and task offloading in different devices. What's more, the social awareness is integrated into our proposed network system, which is rarely considered in current research works, such as [27], [29] and so on. And the existing methods that are used for solving the joint problem are not as effective as the approach we proposed based on the DRL and PPO.

B. The Application of DRL in MEC

With the gradual application of deep learning and reinforcement learning [30], due to its advantages of independent decision-making, strong adaptability, handling complex tasks and exploring unknown areas, DRL has been more widely applied to solve joint optimization problems in MEC systems [19], [21], [31].

Shang et al. [15] investigated the optimization of joint computation offloading and resource allocation in NOMA-MEC systems, aiming at minimizing the computation overhead, then a DRL approach was proposed to solve their problem. Jiao et al. [18] optimized the task completion time and energy consumption in the proposed MEC-enabled Industrial Internet of Things (IIoT) system, a time-energy tradeoff online offloading algorithm using DRL was proposed based on stochastic strategy, cross-mutation technology and a feasible sub-optimal offloading method. Research [4] proposed computational offloading using reinforcement learning (CORL) schemes to minimize the latency and energy consumption of IoMT's medical devices in processing sensor data, and framing the problem as a combination of latency and energy cost minimization to meet the constraints of lack of limited battery capacity and service latency deadline constraints. In [16], Huang et al. considered a WP-MEC network with binary offloading strategy, and proposed a DRL-based online offloading framework, which can adapt to task offload decision and wireless resource allocation according to the wireless channels. In [21], the problems of dynamic caching, computing offloading and resource allocation in MEC system was studied. A dynamic scheduling strategy based on DRL and DDPG was proposed to minimize the long-term average of the cost. Deng et al. [32] proposed an autonomous partial offloading system for delay-sensitive computing tasks in multi-user IIoT-MEC systems with the goal of providing the shortest latency offloading service, and proposed DRL-based offloading methods to optimize latency. Zhang et al. [33] optimized offloading and resource allocation in a WPT-enabled IoT network with multiple HAPs to maximize computation rate, and a DRL-based algorithm and Lagrangian duality method achieve efficient, near-optimal solutions with fast convergence.

TABLE I
COMPARISON OF EXISTING RESEARCH WORKS

Works	Wireless Powered	Partial Offloading	D2D	DRL	Hybrid Action Space
[9], [25], [27], [28]	✓	✓	×	×	×
[26]	✓	✓	✓	×	×
[15], [21], [31]	×	×	×	✓	✓
[18], [4]	×	×	×	✓	×
[16]	✓	×	×	✓	×
[32]	×	✓	×	✓	×
[3]	×	✓	×	✓	✓
[35]	×	×	✓	✓	✓
[29], [33]	✓	✓	×	✓	×
ours	✓	✓	✓	✓	✓

According to the above researched, the DRL are widely used in the MEC networks for settling the problem like task offloading [14], [34], resource allocation and caching, but few studies have combined wireless charging with these above problem [15], [16], [18]. In addition, most of the problems concerned by the above researches focus on a single action space (discrete or continuous) by using traditional deep reinforcement learning methods [29] or the optimization problem is divided into several sub-problems and solved by multiple methods [33]. However, for the hybrid and high-dimensional action space, it is difficult to use the traditional deep reinforcement learning method to solve it.

C. Summary of Related Works

Then we summarize the difference of our work and the existing researches which is shown in Table I. And we can conclude that our work comprehensively considers many factors in wireless powered MEC including partial offloading, D2D offloading strategy based on device's social relationships reflected by humans and time-varying channels. In view of the above factors, we use PPO algorithm based on deep reinforcement learning to solve our proposed optimization problem. Note that our problems with the particularity of discrete and continuous action space, the traditional DRL methods that consider single action space can not solve our problem effectively [16], [18], [32]. Although the methods proposed in [3] and [22] include the discrete and continuous action space, but the PPO used in our approach is proved being superior to Hybrid Advantage Actor-Critic and Hybrid Actor-Critic in convergence and stability.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we will illustrate our system model, communication model, computational model as well as social relationship model. Next, we propose and formulate the joint optimization problem addressed in this section.

A. System Model

We consider a system consisting of user devices, nearby auxiliary devices, and MEC servers, which is centered around MEC servers equipped with wireless charging device. The set of user devices is described as $I = \{l_1, l_2, \ldots, l_i, \ldots, l_I\}$,

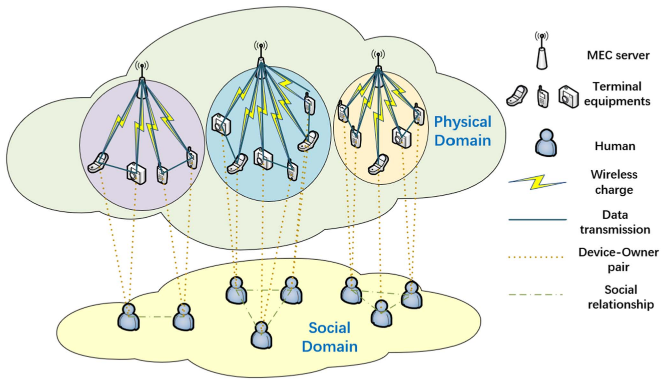


Fig. 1. An illustration of system model.

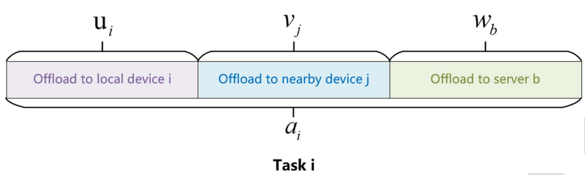


Fig. 2. Description of task partitioning.

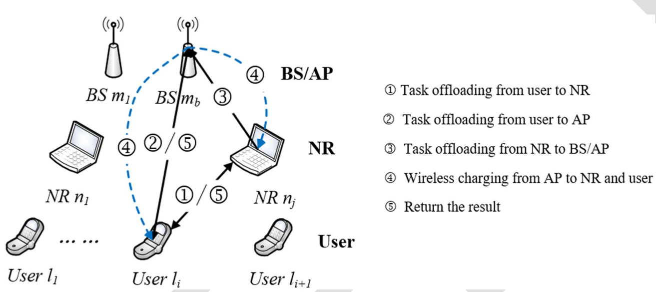


Fig. 3. Process of task offloading and wireless charging in network.

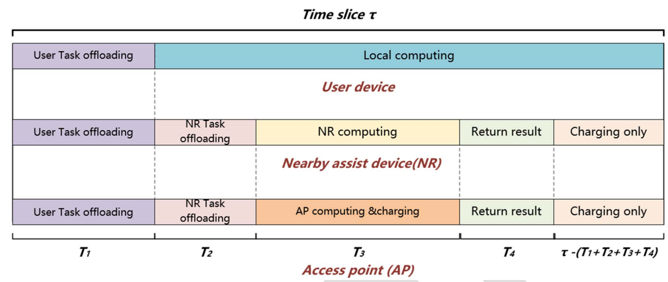


Fig. 4. An illustration of time slice on three devices.

TABLE II
NOTATION TABLE

Notation	Description
τ	time slice
W	the bandwidth of the channels
N_0	the channel noise
$P_i^U(t)$	the transmission power of the user device i at time frame t
h_0	the antenna gain
θ	path loss
f_c	the carrier frequency
$P_j^N(t)$	the transmission power of the nearby auxiliary device j at time frame t
$P_b^B(t)$	the transmission power of the MEC server b at time frame t
ε	the capability of the user devices
P_U	the execution power of user devices
σ	the capability of the nearby auxiliary devices
P_N	the execution power of the nearby auxiliary devices
φ	the capability of MEC servers
P_B	the execution power of MEC servers
$k^U(t)$	the residual energy of the user devices
$k^N(t)$	the residual energy of the nearby auxiliary devices
$P_c(t)$	the power of the charging equipment for supplying energy
$\boldsymbol{\omega}_i(t)$	the social relationship vector of the user device i
ρ	the weight value of the social relationship strength and the remaining energy of the MEC devices.
P_{max}^U	the maximum transmission power of the user devices
P_{max}^N	the maximum transmission power of the nearby auxiliary devices
P_{max}^B	the maximum transmission power of the MEC servers
k_{max}^U	the maximum energy storage of the user devices
k_{max}^N	the maximum energy storage of the nearby auxiliary devices

$N = \{n_1, n_2, \ldots, n_j, \ldots, n_J\}$ is the set of nearby auxiliary devices, and the $M = \{m_1, m_2, \ldots, m_b, \ldots, m_B\}$ is the set of MEC servers, where the I, J, B represent the number of user devices, nearby auxiliary devices and MEC servers, respectively. We construct the system model, as shown in Fig. 1. In this system, the relationships of devices in physical domain are mapped by the relationships of humans in the social domain.

Specifically, we take user device i as an example for explaining our system model. In Fig. 3, the process of task offloading, wireless charging and returning the task execution result are completed through the execution process ①– ⑤. The process ①, ②, ③ illustrate the task partial offloading from user devices to nearby devices and MEC servers, ④ is the charging process during task execution, and the execution results are returned in process ⑤. The tasks in the network are generate on user devices and the arrival of tasks in the system follows a Poisson process based on the research [36]. What's more, the tasks under this system are partially offloaded to MEC server, nearby auxiliary device and user device. Therefore, when selecting an offloading strategy for the task of size a_i generated by user device, the task can be divided into three parts for partial offloading. The task size of three parts can be represented as $a_i =< u_i, v_j, w_b >$, where a_i is the task size and u_i, v_j, w_b are assigned to the size of tasks offloaded to user device i, nearby auxiliary device j and MEC server b respectively, which is shown as Fig. 2. Note that, a_i=u_i+v_j+w_b.

According to the above network model, given the time constraint of τ, the time span of task transmission, task execution, wireless charging and return of execution results should not exceed time slice τ [9], [37]. The time slice diagram of user device, nearby auxiliary device and MEC server is shown in Fig. 4. The task offloading process and the wireless charging process are performed simultaneously on three devices. We suppose that the process of wireless charging and task offloading from the user to MEC servers are implemented simultaneously over orthogonal frequency bands, so that the two transmitted signals do not interfere with each other [37], [38].

To enhance the readability of this paper, the primary notations used throughout the paper are listed in Table II.

B. Communication Model

Based on the above system model, the data transmission process includes the data uplink process for offloading tasks and the data downlink process for returning results. The uplink and downlink transmission rates can be calculated according to the Shannon formula as shown below. Additionally, the co-channel interference among user devices is ignored, while assuming that each user device is allocated with an orthogonal spectrum [34].

1) Uplink Transmission Rate: The transmission rate of the channel between the user device i and the nearby auxiliary device j is expressed as follows.

$$r_{i,j}^{UN} = Wlog_2\left(1 + \frac{P_i^U(t)h_{i,j}^{UN}(t)}{N_0}\right) \tag{1}$$

where W represents the bandwidth of the channels in MEC network; N_0 is the channel noise and the transmission power of user device i at time slice t is expressed as $P_i^U(t)$; $h_{i,j}^{UN}(t)$ is the wireless channel gain which is follow the free space path loss model, expressed as $h_{i,j}^{UN} = h_0(\frac{3*10^8}{4\pi f_c d_{i,j}^{UN}})^\theta$ [18]. And the antenna gain is represented as h_0, θ is the path loss, f_c denotes the carrier frequency and the distance between user device i and the nearby auxiliary device j is expressed as $d_{i,j}^{UN}$ [16].

Furthermore, the transmission rate of the channel between user device i and the MEC server b can be calculated as follows.

$$r_{i,b}^{UB} = Wlog_2\left(1 + \frac{P_i^U(t)h_{i,b}^{UB}(t)}{N_0}\right) \tag{2}$$

where the $h_{i,b}^{UB}(t)$ denotes the wireless channel gain of the channel between user device i and the MEC server b. Similarly, $h_{i,b}^{UB} = h_0(\frac{3*10^8}{4\pi f_c d_{i,b}^{UB}})^\theta$ where the $d_{i,b}^{UB}$ represents the distance between user device i and the MEC server b.

And the Formula 3 expresses the transmission rate of the channel connecting the nearby auxiliary device j with the MEC server b.

$$r_{j,b}^{NB} = Wlog_2\left(1 + \frac{P_j^N(t)h_{j,b}^{NB}(t)}{N_0}\right) \tag{3}$$

where the $P_j^N(t)$ is the transmission power of the nearby auxiliary device j at the time slice t; and $h_{j,b}^{NB} = h_0(\frac{3*10^8}{4\pi f_c d_{j,b}^{NB}})^\theta$, the distance between the nearby auxiliary device j and the MEC server b is $d_{j,b}^{NB}$.

2) Downlink Transmission Rate: The MEC servers and the nearby auxiliary devices return the execution results through the downlink channels. The transmission rate of the downlink channels can be calculated as follows respectively.

The transmission rate of the downlink channel between MEC server b and the user device i is represented as follows.

$$r_{b,i}^{BU} = Wlog_2\left(1 + \frac{P_b^B(t)h_{b,i}^{BU}(t)}{N_0}\right) \tag{4}$$

where the $P_b^B(t)$ denotes the transmission power of the MEC server b at time slice t; and $h_{b,i}^{BU} = h_0(\frac{3*10^8}{4\pi f_c d_{i,b}^{UB}})^\theta$.

The downlink channel's transmission rate between the nearby auxiliary device j and user device i is as follows:

$$r_{j,i}^{NU} = Wlog_2\left(1 + \frac{P_j^N(t)h_{j,i}^{NU}(t)}{N_0}\right) \tag{5}$$

where $h_{j,i}^{NU} = h_0(\frac{3*10^8}{4\pi f_c d_{i,j}^{UN}})^\theta$, which follows the free space path loss.

C. Computational Model

According to the system model and the partition of time slice, the computational model consists of task transmission, task execution, execution result return and wireless charging.

1) Task Transmission: For the task a_i generated by user device i, the task will be divided into three parts u_i, v_j, w_b which will perform locally, be offloaded to the nearby auxiliary device and MEC server respectively. And the task executed on the nearby auxiliary devices and edge servers will be transmitted by the uplink channels.

Additionally, for the w_b offloads to MEC server b, the transmission process of this part of task is divided into two parts: 1) w_b^U transmits from user devices to MEC servers directly; 2) and the another part w_b^N is transmitted from user devices to the nearby auxiliary devices and then from the nearby auxiliary devices to MEC servers. And it satisfies the constraint: $w_b = w_b^U + w_b^N$.

Therefore, the delay and energy consumed by the process of transmitting w_b^U from user device to MEC server can be expressed as follows.

$$t_{i,b}^{UB} = \frac{w_b^U}{r_{i,b}^{UB}} \tag{6}$$

$$e_{i,b}^{UB} = P_i^U(t)t_{i,b}^{UB} \tag{7}$$

The $P_i^U(t)$ is the transmission power of user device i at time t. And the delay and energy required by the transmission of task w_b^N and the task v_j are described as follows.

$$t_{i,j}^{UN} = \frac{v_j + w_b^N}{r_{i,j}^{UN}} \tag{8}$$

$$e_{i,j}^{UN} = P_i^U(t)t_{i,j}^{UN} \tag{9}$$

The delay and energy consumed during the transmission from the nearby device to MEC server of task w_b^N are as follows.

$$t_{j,b}^{NB} = \frac{w_b^N}{r_{j,b}^{NB}} \tag{10}$$

$$e_{j,b}^{NB} = P_i^U(t)t_{j,b}^{NB} \tag{11}$$

According to the above, for the task a_i generated by the user device, the energy consumption of the transmission is:

$$e_i^s = e_{i,b}^{UB} + e_{i,j}^{UN} + e_{j,b}^{NB} \tag{12}$$

2) Task Execution: After the task transmission is completed, the devices start to execute the task. Tasks are executed by the user devices, the nearby auxiliary device and MEC servers simultaneously.

The consumption of delay and energy of execution on user device i are:

$$t_i^U = \frac{u_i}{\varepsilon} \tag{13}$$

$$e_i^U = P_U t_i^U \tag{14}$$

where ε is the capability of user devices, and P_U is the execution power of user devices.

The delay and energy consumed by the execution of the nearby auxiliary device are calculated as follows:

$$t_j^N = \frac{v_j}{\sigma} \tag{15}$$

$$e_j^N = P_N t_j^N \tag{16}$$

where the capability of the nearby auxiliary devices is denoted as σ, and the P_N is the execution power of the nearby auxiliary devices.

Moreover, the delay and energy consumption of MEC server's execution process can be described as:

$$t_b^B = \frac{w_b}{\varphi} \tag{17}$$

$$e_b^B = P_B t_b^B \tag{18}$$

where φ is the capability of MEC servers, and the execution power of MEC servers is denoted as P_B.

Therefore, for task a_i generated by user device i, the energy consumed by the execution process is:

$$e_i^{exe} = e_i^U + e_j^N + e_b^B \tag{19}$$

3) Execution Results Return: After the task is executed, the nearby auxiliary device and MEC server return the execution results back to user device through the downlink.

The delay and energy consumption of returning results from nearby auxiliary device are shown as follows:

$$t_{j,i}^{NU} = \frac{v_j'}{r_{j,i}^{NU}}, \tag{20}$$

$$e_{j,i}^{NU} = P_j^N(t) t_{j,i}^{NU}. \tag{21}$$

We assume that the execute result is proportional to the task size based on [26], which can be formulated as $v_j' = \mu v_j$, where μ is the proportional coefficient of task size.

Analogously, the delay and energy consumed by the MEC server for transmitting the execution result can be described as follows:

$$t_{b,i}^{BU} = \frac{w_b'}{r_{b,i}^{BU}} \tag{22}$$

$$e_{b,i}^{BU} = P_b^B(t) t_{b,i}^{BU} \tag{23}$$

where the return results w_b' is proportional to the execution task size of MEC server, which can be expressed as $w_b' = \mu w_b$.

Therefore, we calculate the consumption of delay and energy for transmitting the result as follows:

$$e_i^r = e_{j,i}^{NU} + e_{b,i}^{BU} \tag{24}$$

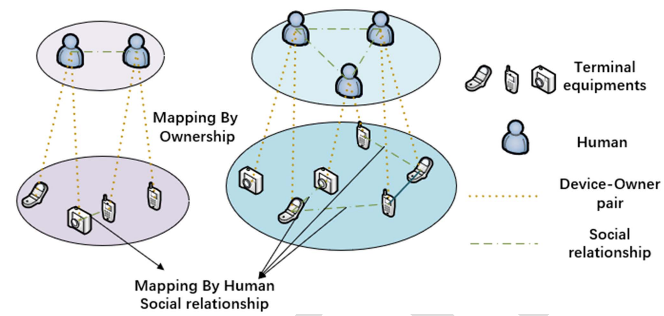


Fig. 5. An illustration of social relationship.

4) Wireless Charging: During task execution, the AP device placed on the MEC server charges user devices and the nearby auxiliary devices. We define that the residual electricity of user devices can be denoted as $k^U(t) = (k_1^U(t), k_2^U(t), \ldots, k_i^U(t), \ldots)$, and $k^N(t) = (k_1^N(t), k_2^N(t), \ldots, k_j^N(t), \ldots)$ shows the residual electricity of the nearby auxiliary devices; where $k_i^U(t)$ is the residual electricity of user device i and $k_j^N(t)$ is the residual electricity of the nearby auxiliary device j. Then we can get the time for wireless charging according to the time slice illustration of Fig. 4.

$$t_c = \tau - (T_1 + T_2 + T_4) \tag{25}$$

and T_1, T_2, T_4 are as follows:

$$T_1 = \max\{t_{i,b}^{UB}, t_{i,j}^{UN}\} \tag{26}$$

$$T_2 = t_{j,b}^{NB} \tag{27}$$

$$T_4 = \max\{t_{j,i}^{NU}, t_{b,i}^{BU}\} \tag{28}$$

The electricity produced by the AP device can be described as:

$$e_c = P_c(t) t_c \tag{29}$$

where $P_c(t)$ is the power of the charging equipment for supplying energy.

Thus, we can obtain the sum of residual electricity of the user device and the nearby auxiliary device.

$$E_c = k_i^U(t) + k_j^N(t) + e_c \tag{30}$$

Moreover, the energy consumed by MEC server for transmitting the electricity can be calculated as follows:

$$e_c t = P_b^B(t) t_c \tag{31}$$

D. Social Relationship Model

In this system, we establish a correlation between the social relationships among devices and those among users in the real world [11], [39]. As illustrated in the network model depicted in Fig. 5, we utilize the ownership of devices by users to reflect the relationships in the social domain and the physical interactions of devices. Specifically, the social relationships between devices are modeled based on the social connections between the users who own these devices.

Additionally, the social relationships between users are characterized by similarities in their behaviors. This behavioral similarity is quantified by analyzing the likelihood of users selecting similar content and engaging in comparable activities. For example, on a movie streaming platform, if user A and user B both frequently watch action movies, and also follow similar movie critics, the system can measure their similarity based on these shared preferences. By examining patterns in content preferences and interaction histories, we can assess the degree of similarity and then the social connections between users, which is described in previous research [40]. Consequently, the strength of social relationships can be derived using this method. Thus, the construction of the social relationship matrix is defined as follows:

$$\Psi(t) = (\boldsymbol{\omega}_1(t), \boldsymbol{\omega}_2(t), \ldots, \boldsymbol{\omega}_i(t), \ldots), \tag{32}$$

where $\boldsymbol{\omega}_i(t)$ represents the social relationship vector of the user devices, and $\boldsymbol{\omega}_i(t) = (\omega_{i,1}(t), \omega_{i,2}(t), \ldots, \omega_{i,j}(t), \ldots)^T$ where the $\omega_{i,j}(t)$ represents the quantification of the social relationship between the user device i and the auxiliary device j at time t.

In particular, $\omega_{i,j}(t) \in [0, 1]$, and $\omega_{i,j}(t) = 0$ indicates that there is no social relationship between the user device i and the auxiliary device j at time t, and therefore the task offloading and transmission link cannot be established between the two devices. In other words, when two users have a stronger social relationship, they are more inclined to establish a D2D link in order to offload tasks.

E. Problem Formulation

According to the above communication, computation and social models, in this MEC system, we are aiming to choose a task offloading strategy with higher trust and less energy consumption, while charging user devices and auxiliary devices as much as possible during the task offloading process. In this system, the energy consumed in the task offloading process and power transmission process is shown as follows.

$$E_i(t) = e_i^s + e_i^{exe} + e_i^r + e_{ct} \tag{33}$$

And we define the utility function as follows:

$$Q_{i,j,b}(t) = \Sigma \rho \omega_{i,j}(t) + (1-\rho)(E_c - E_i(t)) \tag{34}$$

where $\rho \in [0, 1]$, and ρ represents the weight value of the social relationship strength and the remaining energy of the MEC devices.

Based on this, we formulate this problem as:

$$\max_{\{a_i, P^U(t), P^N(t), P^B(t), k^U, k^N\}} Q \tag{35}$$

s.t.

$$C1 : 0 \leq P_i^U(t) \leq P_{max}^U$$
$$C2 : 0 \leq P_j^N(t) \leq P_{max}^N$$
$$C3 : 0 \leq P_b^B(t) \leq P_{max}^B$$
$$C4 : \sum_{m=1}^{4} T_m \leq \tau$$
$$C5 : t_i^U + T_1 \leq \tau$$
$$C6 : k_i^U(t) - e_{i,b}^{U,B} - e_{i,j}^{U,N} - e_i^U \geq 0$$
$$C7 : k_j^N(t) - e_{j,b}^{N,B} - e_j^N - e_{j,i}^{NU} \geq 0$$
$$C8 : E_c \leq k_{max}^U + k_{max}^N$$

In above (35), C1, C2, C3 constrain the power of user devices, the nearby auxiliary devices and MEC servers respectively, which guarantees that the power of different devices varies in a reasonable range. The P_{max}^U, P_{max}^N and P_{max}^B represent the maximum power of user devices, nearby auxiliary devices and MEC servers respectively. Then, C4 and C5 are the time constraints based on the time slice partitioning which is shown as Fig. 4, illustrating that the sum of transmission time, execution time, charging time on different devices is less than the length of a time slice τ. Finally, C6, C7 and C8 ensure the effective range of the remaining power of the user devices and the nearby auxiliary devices, where C6, C7 make sure that the residual power of devices is sufficient to support kinds of energy consumption, and C8 guarantees that the obtained energy by charging from APs is not excessive for maximum battery capacity k_{max}^U and k_{max}^N of the devices.

IV. Optimization Algorithm of Task Offloading and Wireless Charging Based on Deep Reinforcement Learning

The formulated optimization problem mainly includes two parts, i.e., task offloading with its distribution and wireless power distribution.

1) For the task offloading and distribution, we consider the partial offloading in MEC system, and the task a_i generated at time t is divided into three parts, expressed as $a_i =< u_i, v_j, w_b >$; corresponding to the tasks that offload to the user device i, the nearby auxiliary device j and the MEC server b respectively. Therefore, we are intent to optimize the problem (35) by finding an appropriate task offloading and distribution strategy.
2) For the wireless power distribution, we can make the devices in the system consume less energy and charge more through rational distribution of wireless channel power so that the devices in the system can keep their residual energy as much as possible.

A. Problem Modeling and Algorithm Overview

To solve the formulated optimization problem, we leverage DRL to mathematically model the problem and construct the algorithm framework.

1) Problem Model: Under the framework of DRL, the interaction process between the agent and the environment is roughly described as follows: first, the agent makes an action decision according to the current environment state and decides to take an action; then, the environment will give reward according to the agent's action, and the action influences the next state; finally, the agent generates a large amount of states, actions and rewards in the process of continuous interaction with the environment, which is oriented to the actual task goal [41]. Based on these data, the DRL algorithm can enable the agent to make more correct decisions, which is to learn the strategy. And the optimal problem can be formulated as Markov Decision Process (MDP) [19]. MDP encapsulates the essence of learning through trial and error, encompassing three fundamental components: actions, states, and rewards. These elements can be described as follows:

- *State space:* In the above problem, we define the state space as consisting of the channel gain and the remaining energy of user devices and the nearby auxiliary devices in the network which can be expressed as: $s_t = \{H_t, K_t\}$. The H_t and K_t are described as follows:

$$H_t = \{h_{i,j}^{UN}(t), h_{i,b}^{UB}(t), h_{j,b}^{NB}(t), h_{b,i}^{BU}(t), h_{j,i}^{NU}(t)\} \tag{36}$$

$$K_t = \{k_1^U(t), \dots, k_i^U(t), \dots, k_1^N(t), \dots, k_j^N(t)\} \tag{37}$$

- *Action space:* The action space consists of task assignment and offloading decisions and wireless channel power allocation decisions, expressed as $a_t = \{a^d(t), a^c(t), p(t)\}$. $a^d(t)$ represents discrete task offloading decisions, while $a^c(t)$ and $p(t)$ represents continuous task allocation decisions and power allocation decisions, respectively. Finally, the hybrid action space is formed by the aforementioned three parts.
- *Reward:* We utilize $r_t(s,a)$ to denote the reward function, which should be proportional to the value of the utility function for the above optimization problem. Therefore, we define the reward function $r_t(s,a)$ as:

$$r_t(s,a) = \sum \rho\omega_{i,j}(t) + (1-\rho)(E_c - E_i(t)) \tag{38}$$

2) Hybrid Actor-Critic Network: Furthermore, we construct the overall framework of the algorithm based on hybrid Actor-Critic network. Hybrid Actor-Critic (HAC) is a DRL algorithm that aims to learn policies that can handle both continuous and discrete action spaces efficiently [3], [15], [22]. In HAC, the actor networks propose actions for the environment, while the critic network assesses the quality of these actions. The model predicts the next state based on the current state and action, generating synthetic transitions that are used to refine both the actor and critic networks. The actor networks update their parameters using the policy gradient method to maximize the expected cumulative reward, while the critic network estimates the value function and provides feedback to enhance the actors' performance.

The actor network uses a policy-based approach to optimize policies directly and maximize cumulative rewards by iteratively updating the policies. Considering that the policy gradient has the disadvantages of low sampling efficiency and unstable training process, we adopt hybird PPO to solve these problems more effectively [42], [43]. Besides, hybird PPO uses the advantage function to estimate the advantage value of each state-action pair, which is used to measure the advantage or value of taking a specific action relative to the average behavior in a given state. The advantage function which is the policy loss of actor network is expressed as: $\hat{A}_\pi(s,a) = Q_\pi(s,a) - v_\pi(s)$. In the above formula, $\hat{A}_\pi(s,a)$ represents the advantage value of taking action a in state s, $Q_\pi(s,a)$ is the state value function after taking action a and finally the $v_\pi(s)$ is the value function of the current state s.

Accordingly, the critic network employs a value-based approach to acquire a deterministic strategy that makes decisions by relying on the numerical value function. In particular, the state value function is expressed as $v_\pi(s) = \mathbb{E}[G_t|s_t = s]$ where the $G_t = r_t + \gamma_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ [44], where $\gamma \in [0,1]$ and γ presents discount factor, determines the present value of future rewards. If $\gamma = 0$, we're thinking about maximizing the immediate return right now. As γ grows, it becomes more focused on future returns. Besides, the action value function is defined as $Q_\pi(s,a) = \mathbb{E}[G_t|s_t = s, a_t = a]$, and the value-based method typically involves optimizing the action-value function $Q_\pi(s,a)$, where the optimal strategy is derived by selecting the action that corresponds to the highest value in the $Q_\pi(s,a)$ function.

In addition,the Generalized Advantage Estimation(GAE) is used for estimating the value of advantage function which combines Monte Carlo estimation and TD(λ) methods. The method of GAE is to make multi-step estimates of the advantage function and combine these multi-step estimates with discount factor. $\hat{A}_t^{GAE(\gamma,\lambda)} = (1-\lambda)(\hat{A}_t^1 + \lambda\hat{A}_t^2 + \lambda^2\hat{A}_t^3 + \cdots) = \sum_{l=0}^{\infty}(\gamma\lambda)^l\delta_{t+1}^v$, where GAE's parameter λ controls the trade-off between Monte Carlo estimates and TD(λ) methods. When $\lambda = 0$, GAE is equivalent to using only Monte Carlo estimation; when $\lambda = 1$, GAE is equivalent to using only the TD(λ) method. By adjusting the value of λ, a compromise can be made between bias and variance. Moreover, the formula above is calculated based on the 1-step estimation, 2-step estimation and infinite step estimation of the advantage function, which are shown respectively as follows: $\hat{A}_t^1 = \delta_t^v = -v(s_t) + r_t + \gamma v(s_{t+1})$, $\hat{A}_t^2 = \delta_t^v + \gamma\delta_{t+1}^v = -v(s_t) + r_t + \gamma r_{t+1} + \gamma^2 v(s_{t+2})$ and $\hat{A}_t^\infty = \sum_{l=0}^{\infty}\gamma^l\delta_{t+l}^v = -v(s_t) + \sum_{l=0}^{\infty}\gamma^l r_{t+l}$.

3) Algorithm Overview: The algorithm is based on the hybrid actor-critic and the overall framework of algorithm is depicted in Fig. 6. The actor network is consist of three modules, which are offloading decision module, task allocation module and power allocation module. Thus, the discrete strategy and two continuous strategies are parameterised respectively as $\pi_\eta(a_t^d|s_t)$, $\pi_\chi(a_t^c|s_t)$, $\pi_\xi(p_t|s_t)$, and the critic value function is $V_\omega(s_t)$. The methodology used in this paper consists of two phases. In the first stage, action generation takes place. The states obtained by the environment are transferred to the actor networks and the critic network. Then, the discrete actor network generates the offloading decision, and continuous actor networks determine the task assignment and power assignment, three actions constitute a hybrid action to feedback the environment. Nextly, the environment calculates the immediate reward r_t according to the feedback of the action and stores it in the form of (s_t, a_t, r_t, s_{t+1}). In the second stage, policy updates are performed, and the algorithm uses a set of samples from the buffer to train actor networks and critic network. The algorithm of the general process is summarized as follows.

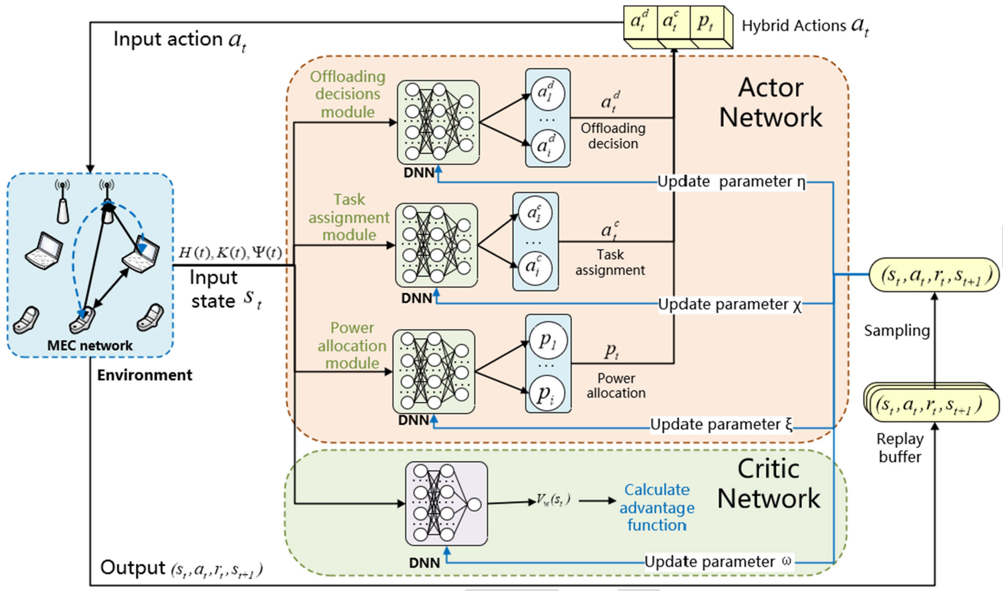


Fig. 6. The overall framework of algorithm.

Algorithm 1: Proposed Approach.

Initialize:

1: Initialize the parameters of the DNNs including embedding parameter η, χ, ξ, ω and the weight and bias of DNNs;
2: **for** $Iteration = 1 : N$ **do**
3: Collect current state of environment s_t;
4: Three actor networks generate actions a_t^d, a_t^c, p_t respectively, and the three form a hybrid action a_t;
5: Calculate the value of reward and advantage function;
6: Store the transition $\{s_t, a_t, r_t, s_{t+1}\}$ into the replay buffer;
7: **end for**
8: **for** Iteration of updating actor networks **do**
9: Calculate $r_t(\eta)$, $r_t(\chi)$, $r_t(\xi)$ and the objective functions of three actor networks $L^{CLIP}(\eta)$, $L^{CLIP}(\chi)$, $L^{CLIP}(\xi)$, respectively;
10: Update η, χ, ξ by maximizing the object functions of three actor networks;
11: **end for**
12: **for** Iteration of updating critic network **do**
13: Calculate the loss function $L_{critic}(\omega)$ of the critic network;
14: Update ω by minimizing the loss function of critic network;
15: **end for**

B. Hybrid Action Generation Module

We can get the wireless channel gain H_t and the remaining energy K_t at time t according to the network and devices state. The environmental state will be entered into the task allocation module, the offloading decision module and the power allocation module respectively. Next, we will discuss the above three modules separately.

1) Offloading Decision Module: In above MEC network, the task generated by user devices will be executed locally, directly to MEC server, D2D and D2D-assisted manners. To describe the nearby auxiliary devices and the MEC servers involved under the above four offloading strategies, we represent the task offloading decision generated by the DNN characterized by the embedded parameter η as follows.

$$a_{l,t}^d = \{(d_{l,1}^N, \ldots, d_{l,J}^N), (d_{l,1}^B, \ldots, d_{l,B}^B) | d_{l,j}^N, d_{l,b}^B \in \{0, 1\}\} \tag{39}$$

$$a_t^d = \{a_{1,t}^d, \ldots, a_{l,t}^d, \ldots | l = 1, \ldots, L\} \tag{40}$$

where the $d_{l,j}^N$ and $d_{l,b}^B$ indicate whether the task generated by user device l will be offloaded to the nearby auxiliary device j and MEC server b or not. At a certain time slice t, if $d_{l,j}^N = 1$ and $d_{l,b}^B = 1$, it means that the task offloading decision generated by DNN is offloading the task generated by user device l to the nearby auxiliary device j and MEC server b partially. Therefore, the discrete decision at time slice t $\pi_\eta(a_t^d|s_t)$ can be calculated by the offloading strategies of every task at time slice t, which can be described as follows.

$$\pi_\eta(a_t^d|s_t) = \prod_{l=0}^{l=L} \pi_\eta(a_{l,t}^d|s_t) \tag{41}$$

2) Task Allocation Module: In particular, we consider a task partial offloading, the task generated by user device will be divided into three parts which will execute locally, offload to nearby auxiliary device and MEC server. Hence, the task allocation module makes the decision which determines the task allocation proportion to three kinds of devices. Then, we represent the task allocation decision as follows.

$$a_{l,t}^c = \{u_{l,t}, v_{l,t}, w_{l,t} | u_{l,t}, v_{l,t}, w_{l,t} \in [0,1], l = 1, \ldots, L\} \tag{42}$$

$$a_t^c = \{a_{1,t}^c, \ldots, a_{l,t}^c, \ldots | l = 1, \ldots, L\} \tag{43}$$

where $u_{l,t}$, $v_{l,t}$, and $w_{l,t}$ represent the proportion of tasks assigned to user devices, the nearby auxiliary devices, and MEC server devices, respectively. What's more, $u_{l,t}$, $v_{l,t}$ and $w_{l,t}$ satisfy equation $u_{l,t} + v_{l,t} + w_{l,t} = 1$. The continuous decision $\pi_\chi(a_t^c|s_t)$ generated by DNN characterized by the embedding parameter χ is shown as follows.

$$\pi_\chi(a_t^c|s_t) = \prod_{l=0}^{l=L} \pi_\chi(a_{l,t}^c|s_t) \tag{44}$$

3) Power Allocation Module: For reducing the energy consumption and enhancing the efficiency of task offloading, we dynamically assign the power of channels between devices. Then, the power allocation decision can be represented as follows:

$$p_{l,t} = \{(p_{l,1}^U, \ldots, p_{l,1}^N, \ldots, p_{l,1}^B, \ldots) | l = 1, \ldots, L\} \tag{45}$$

$$p_t = \{p_{1,t}, \ldots, p_{l,t}, \ldots | l = 1, \ldots, L\} \tag{46}$$

where $p_{l,i}^U \in [0, P_{max}^U]$, $p_{l,j}^N \in [0, P_{max}^N]$, $p_{l,b}^B \in [0, P_{max}^B]$ and $p_{l,i}^U$, $p_{l,j}^N$, $p_{l,b}^B$ are corresponding to the channel power of the user device, the nearby auxiliary device and MEC server. We represent the power allocation decision $\pi_\xi(p_t|s_t)$ generated at time t by a DNN characterized by the embedded parameter ξ as follows.

$$\pi_\xi(p_t|s_t) = \prod_{l=0}^{l=L} \pi_\xi(p_{l,t}|s_t) \tag{47}$$

C. Policy Update Module

The hybrid Proximal Policy Optimization (PPO) method is adopt for updating actor and critic networks, hybrid PPO aims to optimize the decision-making capability of agents by limiting the magnitude of policy updates. The core idea of this algorithm is to introduce a clipped loss function to control the changes between the old and new policies, thereby enhancing the stability and robustness of policy updates. In the implementation process, hybrid PPO first collects experience data through interaction with the environment, including states, actions, rewards, and next states. Subsequently, it employs Generalized Advantage Estimation (GAE) to compute the advantage function, allowing for a smoother evaluation of the relative effectiveness of each action against the current policy. Hybrid PPO executes multiple optimization steps on each data batch; in this study, we utilize the Adam optimizer to adjust the learning rate adaptively. Furthermore, by systematically tuning hyperparameters such as learning rate, batch size, and clipping range through experimental methods, our approach based on hybrid PPO achieves good convergence and performance outcomes. By limiting the update steps using PPO-clip mechanism, this approach prevents policy collapse, improves algorithm stability, and enhances sample efficiency by incorporating efficient sampling techniques, compared with other DRL models [45].

1) Actor Network Update: As shown in Fig. 6, after receiving the mixed actions generated by the actor network, which consist of discrete and continuous actions, the environment obtains the reward value of the corresponding action strategy. In addition, the three actor networks update their network parameters with their own independent valid sampling values by maximize the objective functions respectively.

To maximize the value of the reward function, we need to make the advantage of the actor networks output as large as possible. So training the actor networks require updating the corresponding parameters by maximizing the following objective function.

For the discrete actor network, we have the object function as follows:

$$L^{CLIP}(\eta) = \hat{\mathbb{E}}_t[\min(r_t(\eta)\hat{A}_t, clip(r_t(\eta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{48}$$

where the $r_t(\eta)$ and the *clip* function are as follows:

$$r_t(\eta) = \frac{\pi_\eta(a_t^d|s_t)}{\pi_{\eta_{old}}(a_t^d|s_t)} \tag{49}$$

$$clip(r_t(\eta), 1-\epsilon, 1+\epsilon) = \begin{cases} 1-\epsilon, & \text{if } r_t(\eta) \le 1-\epsilon; \\ r_t(\eta), & \text{if } 1-\epsilon < r_t(\eta) < 1+\epsilon; \\ 1+\epsilon, & \text{if } r_t(\eta) \ge 1+\epsilon. \end{cases} \tag{50}$$

In a similar way, for the two continuous actor network, they have their objective function respectively as follows.

$$L^{CLIP}(\chi) = \hat{\mathbb{E}}_t[\min(r_t(\chi)\hat{A}_t, clip(r_t(\chi), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{51}$$

where

$$r_t(\chi) = \frac{\pi_\chi(a_t^c|s_t)}{\pi_{\chi_{old}}(a_t^c|s_t)} \tag{52}$$

$$L^{CLIP}(\xi) = \hat{\mathbb{E}}_t[\min(r_t(\xi)\hat{A}_t, clip(r_t(\xi), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{53}$$

where

$$r_t(\xi) = \frac{\pi_\xi(p_t|s_t)}{\pi_{\xi_{old}}(p_t|s_t)} \tag{54}$$

2) Critic Network Update: The main purpose of the Critic network is to evaluate the current strategy and provide a numerical feedback signal to the actor network to guide it to choose better actions. Based on the current state and actions taken, the critic network calculates the expected reward function value of the current policy and transmits it to the actor network as a feedback signal.

After the actor network is updated, the critic network is then updated based on the data sampled from the replay buffer. And dynamic importance sampling method is adopted to improve the efficiency of sampling. The critic network is updated by using the computed long term discounted rewards G_t and the critic network's prediction of the current state reward value $V_\omega(s_t)$ to do the difference as the loss function to train the DNN. We update the critic network by minimizing the mean square loss function, which is expressed as follows:

$$L_{critic}(\omega) = (V_\omega(s_t) - G_t)^2 \tag{55}$$

In conclusion, hybrid PPO extends traditional Proximal Policy Optimization (PPO) by integrating value function optimization strategies to enhance policy learning in reinforcement learning. By combining multiple optimization techniques, hybrid PPO significantly improves training stability and sample efficiency while accelerating convergence in complex tasks, ultimately yielding higher-quality policies and superior performance in real-world applications. However, unlike traditional hybrid PPO, which applies fixed learning rates separately to discrete and continuous policies, our approach incorporates an adaptive learning rate adjustment mechanism based on gradient variance. This dynamic adjustment enhances training stability and efficiency, allowing for more effective policy updates in highly dynamic environments. Furthermore, while traditional hybrid PPO relies on standard importance sampling to correct for off-policy updates, our method introduces a dynamic importance sampling strategy that adjusts sampling weights based on action relevance and transition confidence. This improves training efficiency by prioritizing impactful experiences and reducing variance in policy gradient estimation. Building upon the hybrid PPO framework, our approach integrates adaptive learning rate tuning and enhanced importance sampling, addressing the limitations of traditional hybrid PPO and achieving more robust and efficient reinforcement learning performance.

V. Simulation and Performance Evaluation

This section primarily conducts simulation experiments to evaluate the effectiveness of the proposed algorithm by comparing it against other existing baseline methods.

A. Experimental Setup

The series of simulation experiments are conducted mainly on the computer with Intel Core i5-1135G7 CPU 2.40GHz 2.42GHz, and the operation of Windows 11 system, as well as with the assistance of computer cluster equipped with the 16 core CPU, 32GB of memory and a GPU. We implement the above algorithm using PyTorch in PyCharm2021.3.

1) Parameter Setting: Before conducting the experiment, we initialize the parameters which will be used in the algorithm. Moreover, we consider a circular area that the devices in the MEC system are distributed with the radius of 100m. Specifically, the MEC severs are located in the center of the circular area with the range 10m, and in the range 10m to 50m of the area, the nearby auxiliary devices are randomly distributed, and the user devices are distributed in the annular area with a radius of 50–100 m [40]. Besides, the related parameters are initialized and assigned as Table III based on [3], [15], [18], [32].

2) Comparison System: For the problems with hybrid action space, there are many approaches and models with DRL that can solve the discrete and continuous action at the same time. Moreover, we choose following baseline approaches to compare with our approach, which can proving the advantages of our approach.

- *A3C:* Asynchronous Advantage Actor-Critic(A3C) [35] is an improved version based on the Actor-Critic algorithm, which uses asynchronous parallel training to improve the efficiency and performance of the algorithm. A3C is trained using multiple parallel worker threads, each with a separate instance of the environment in which they interact with the model. These worker threads speed up the training process by asynchronously updating the parameters of the model.
- *A2C:* Advantage Actor Critic(A2C) [31] is a reinforcement learning algorithm based on policy gradient, which combines the idea of Actor-Critic algorithm and advantage function. It is a synchronous update algorithm, which is updating the parameters of the Actor and Critic network at each time step.
- *TRPO:* Trust Region Policy Optimization (TRPO) optimizes policies by maximizing expected rewards while constraining policy updates to remain within a trust region, defined by a KL divergence limit. It formulates the optimization as a constrained problem and utilizes a conjugate gradient method to find optimal policy parameters.
- *LOC:* The tasks generated by the user devices will be executed locally.
- *MEC:* In this approach, the tasks generated by the user devices will be transmitted to the MEC severs and executed on MEC servers.
- *Our proposed approach:* In our proposed approach, we use the main idea of Proximal Policy Optimization (PPO) to solve the problem of hybrid action space including continuous and discrete. We adopt the PPO-clip method to update the actor networks. PPO-clip introduces a clipping mechanism to limit the size of the policy updates which will address the limitation of the original PPO algorithm.

TABLE III
The Parameters Initialization

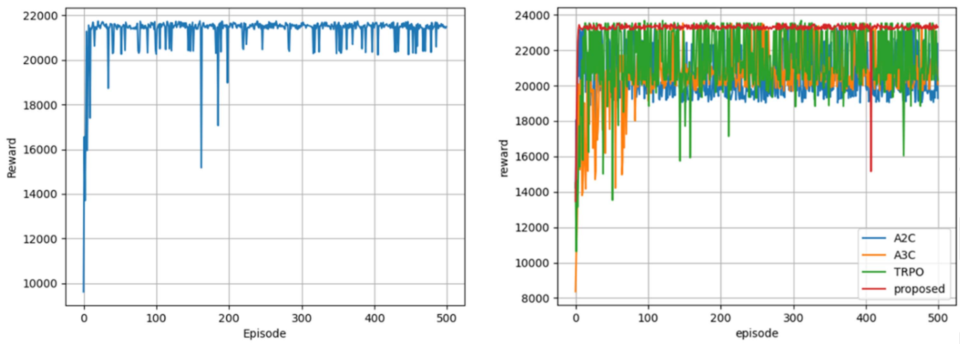
	Parameters	Values
1	W	$10\ MHz$
2	h_0	4.11
3	f_c	$915\ MHz$
4	θ	2.8
5	N_0	$-174\ dBm/Hz$
6	P^U_{max}	$18\ dBm$
7	P^N_{max}	$18\ dBm$
8	P^B_{max}	$70\ dBm$
9	ε	$1\ GHz/s$
10	σ	$1\ GHz/s$
11	ψ	$5\ GHz/s$
12	k^U_{max}	$20000\ J$
13	k^N_{max}	$20000\ J$
14	Batch size	64
15	γ	0.64
16	ϵ	0.1
17	τ	$1\ s$

B. Simulation Results and Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm under different experimental parameters, and compare the experimental results with different approaches.

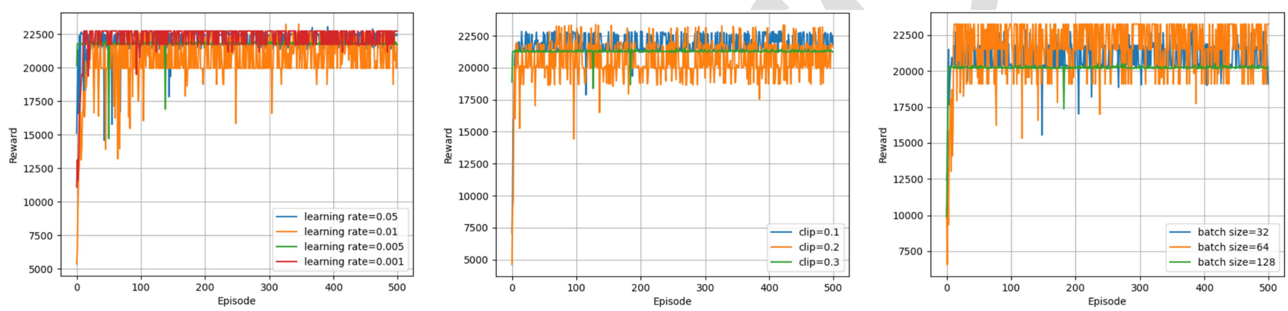
1) Convergence Evaluation: In this section, we analyze and experiment the convergence of our method different parameters and different methods, which is shown in Fig. 7.

Convergence Evaluation: First, we carry out the experiment on the convergence of reward function values under 500 iterations with the learning rate is 0.05, PPO-clip is $\epsilon = 0.1$ and the episode of update parameters as well as the batch size



(a) The convergence result of our approach (b) The comparison of convergence results

Fig. 7. Convergence results.



(a) learning rate (b) ppo-clip (c) batch size

Fig. 8. The performance under different parameters of proposed algorithm.

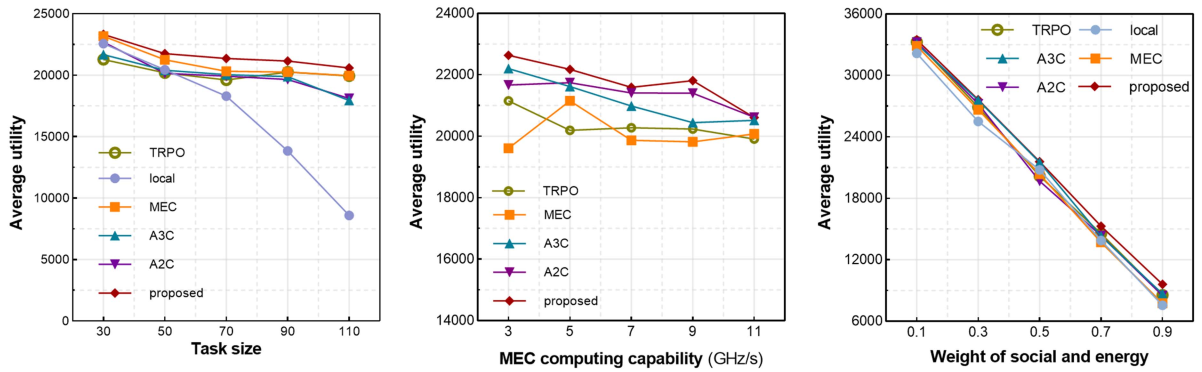
is 100. The result shows the convergence of our approach as Fig. 7(a). The results indicate that the value of reward is increased within a hundred iterations, and the value of the reward function fluctuates in the range of 20 000 to 22 000 gradually. And the results converge gradually after 20 iterations. What's more, the volatility in algorithmic results after convergence can be caused by several factors. First, the randomness in strategy updates in reinforcement learning can make policies unstable, leading to performance fluctuations. Second, the diversity and randomness of training samples, including random sampling in experience replay, contribute to result variability. Additionally, inherent randomness in the environment, such as random rewards or state changes, can cause inconsistencies. Finally, the complexity of the training process and the interaction between different networks in a hybrid structure can also lead to variations in experimental outcomes.

Convergence of Different Methods: Next, for contrasting the convergence of our approach and other methods, we carry out the experiment including four approaches: our approach, A2C, TRPO and A3C that can solve the hybrid action space problems. We compare how the value of the reward function changes as the number of iterations increases under the three methods. Note that the parameters of three kinds of methods are set to the same. The experimental results are shown in Fig. 7(b). Obviously, the experimental results show that our approach converges better than A2C, A3C and TRPO, and the algorithm we adopt converge to a larger value of reward in 100 iterations.

Third, to evaluate the comprehensiveness and completeness of the algorithm results, we also conduct experiments on the convergence performance of the algorithm under different network parameters, such as learning rate, PPO-clip coefficient and update episodes. We implement a comparative experiment on the changes of the reward function value with the number of iterations under different parameters, and the experimental results are shown below.

Convergence under Different Learning Rate: Fig. 8(a) depicts the effect of the learning rate on the change in the value of the reward function. Under different learning rates, the reward function increases rapidly and converges gradually with the increase of the number of iterations. According to the results, we can find that the reward converges best and in the range of 21 000 to 22 500 when the learning rate is set as 0.05, and the reward converges worst in range 20 000 to 22 500 when the learning rate is 0.01. In addition, when the learning rate is set to 0.05, the reward converges earlier than that while learning rate is 0.01, 0.005 and 0.001 which indicates that our algorithm shows better convergence when the learning rate is set to 0.05.

Convergence under Different PPO-clip: Moreover, the performance of our approach under different PPO-clip parameters ϵ is described as Fig. 8(b). The experimental result represents that under different values of PPO-clip parameter the reward increases and then converges with the increase of iterations. What's more, it is obvious that the reward converges to a lager value when $\epsilon = 0.1$ but the last to achieve convergence.



(a) Task size vs Average utility function

(b) The capability of MEC vs Average utility function

(c) Weight of energy and social relationship vs Average utility function

Fig. 9. The effect of different variable values on average utility function.

However, when $\epsilon = 0.3$, it reaches to convergence rapidly but the reward converges to a smaller value. At the same time, when $\epsilon = 0.3$, the performance of the algorithm is better than $\epsilon = 0.1$ in the convergence speed.

Convergence under Different Batch Size: Finally, we can get the variation of the reward function value under different batch sizes from the Fig. 8(c). The reward increases and converges gradually, while it performs better while the batch size is 32 which means we take 32 samples from the replay buffer for a network update. Our approach perform worse while the batch size is 64. We can draw an important conclusion from the above experiments: the parameters in the network will greatly influence the performance of our approach, and we can adjust the different parameter values to make our algorithm perform better.

2) Performance Evaluation: In this section, we carry out several experiments comparing our approach with other methods under different circumstance.

First of all, we conducted experiments by adjusting the variables involved in the optimization problem. We compared the average utility function values corresponding to different variable values under different methods, such as the input task size, the computing capability of the MEC servers, the weight value between energy consumption and social relations in the optimization problem, which is shown as Fig. 9.

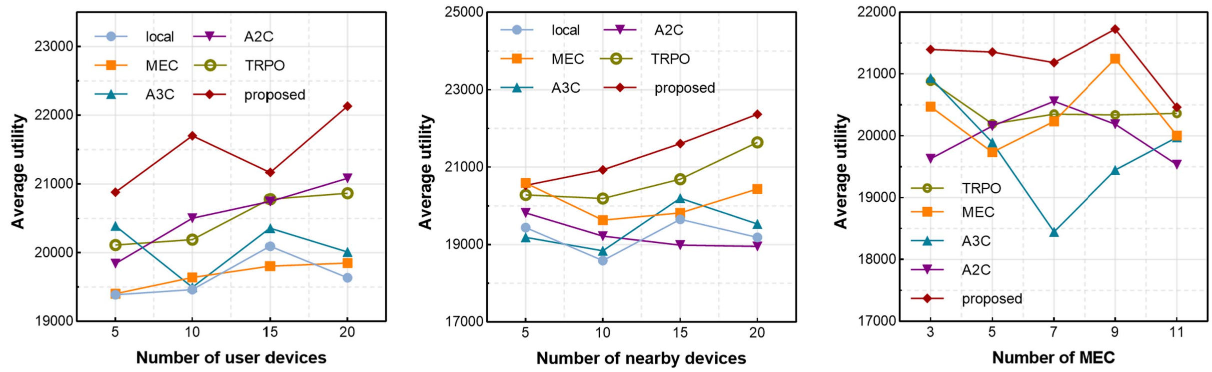
Task Size vs*Average Utility Function:* As shown in Fig. 9(a), the above experimental results show that as the size of the input task increases, the average utility function values corresponding to different methods all decline. The reason for this downward trend is that as the size of the task increases, the energy consumption required by the device to execute and transmit the task increases, and the remaining energy consumption of the device decreases accordingly. At the same time, it is founded that our proposed method shows the largest average utility function value under different input task sizes. Moreover, the average utility function value decreases the fastest under the local method, while the proposed method decreases slowly. While task size changes from 30 to 110, the average utility function decreases 11.69% under our proposed approach, 20.18% under A2C, 17.18% under A3C, 13.83% under MEC and 61.93% under local method, which indicates that our method has better performance and stability under different input task sizes. In particular, the average utility function of our method is 6.79% higher than that of TRPO method.

MEC Computing Capability vs*Average Utility Function:* Next, in Fig. 9(b), the average utility function corresponding to different MEC computing capability under different methods is shown. Note that due to the local method under the average utility function value is not affected by MEC computing capability change, so we exclude it. We can also find that with the increase of MEC computing capability, more of the task is allocated to the MEC server for execution, and the energy consumed in the process of transmitting the task increases, and the average utility function decreases. However, in this experiment, the average utility function value of the proposed method under different MEC computing capability is higher than that under other methods. Especially, compared with MEC method, the average utility function under our proposed method is average 8.27% higher than that under MEC method.

Weight of Energy and Social Relationship vs *Average Utility Function:* Moreover, the variation of average utility function values of different methods under different weight values is shown in Fig. 9(c). We can find that under all methods, as the weight value increases between 0 and 1, which means that the weight of energy consumption in the optimization problem decreases, and the value of the average utility function decreases. The experimental results show that it is necessary to consider the energy consumption and residual power of the devices in the MEC network while optimizing the task offloading problem in MEC network.

Second, we conducted the experiments on the effect of different numbers of different devices on the average utility function. The results of different kinds of device are shown in Fig. 10.

User Device Number vs*Average Utility Function:* Fig. 10(a) illustrates the effect of the number of user devices on the average utility function under different methods. It can be concluded that with the gradual increase of the number of user devices, the average utility function basically presents an upward trend. This is because with the increase of user device number, the task will execute on the user device more, which will reduce energy consumption of transmitting tasks. Moreover, our method shows



(a) User device number vs Average utility function

(b) The nearby device number vs Average utility function

(c) The MEC number vs Average utility function

Fig. 10. The effect of the number of different devices on average utility function.

a greater advantage in average utility function values, while method MEC has a smaller average utility function value.

The Nearby Device Number vs *Average Utility Function:* Then, we conduct the experiment of the impact of the number of nearby auxiliary devices on the average utility function value under different methods. The result is shown in Fig. 10(b). We can find that the average utility function value increases with the nearby auxiliary devices' number increasing from 5 to 20, which means the D2D makes a difference on reducing the energy consumption and improving the efficiency of task offloading in MEC networks. And the average utility function under our proposed method increases 8.97% while others decreases or has a lower rate of increase. Besides, it is also shown that the average utility function value increases more rapidly and is higher than other approaches.

The MEC Number vs*Average Utility Function:* Lastly, in Fig. 10(c), we compare the average utility function of different methods with the MEC number ranging from 3 to 11. We do not adopt the local method as the comparison method, because changing the number of MEC devices has no effect on the results under the local method. The experimental result shows that our method still performs higher average utility function values when changing the number of MEC devices compared with other methods, which proves the superiority of our method. In particular, the average utility function under our proposed method is 7.24% higher than that under A3C.

VI. Conclusion

In this paper, we study the wireless charging and partial computation offloading in time-varying MEC by considering social relationship and devices' remaining energy. To address the problem, we propose a DRL-based approach under the framework of actor-critic with three hybrid actor modules and one critic network. In particular, we use the PPO-clip to update the actor networks and critic network. We carry out simulation experiments to evaluate the performance in terms of the convergence and the average utility function by comparing our proposed method with other methods in different scenarios. The experimental results show that our proposed approach is superior to others in terms of the average utility function value with 1.57% higher than that of A2C and 2.60% than that of A3C. In the future, we will focus on real-life application scenarios of computation offloading and wireless charging in MEC, which will bring us with convenient services and satisfied QoS.

References

[1] F. Vhora and J. Gandhi, "A comprehensive survey on mobile edge computing: Challenges, tools, applications," in *Proc. 4th Int. Conf. Comput. Methodol. Commun.*, 2020, pp. 49–55.

[2] L. A. Haibeh, M. C. E. Yagoub, and A. Jarray, "A survey on mobile edge computing infrastructure: Design, resource management, and optimization approaches," *IEEE Access*, vol. 10, pp. 27591–27610, 2022.

[3] C. Shang, Y. Sun, and H. Luo, "A hybrid deep reinforcement learning approach for dynamic task offloading in noma-MEC system," in *Proc. 19th Annu. IEEE Int. Conf. Sens., Commun., Netw.*, 2022, pp. 434–442.

[4] R. Yadav et al., "Smart healthcare: Rl-based task offloading scheme for edge-enable sensor networks," *IEEE Sensors J.*, vol. 21, no. 22, pp. 24910–24918, Nov. 2021.

[5] O. L. A. López, H. Alves, R. D. Souza, S. Montejo-Sánchez, E. M. G. Fernández, and M. Latva-Aho, "Massive wireless energy transfer: Enabling sustainable IoT toward 6G ERA," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8816–8835, Jun. 2021.

[6] Z. Zhou et al., "Energy-efficient resource allocation for energy harvesting-based cognitive machine-to-machine communications," *IEEE Trans. Cogn. Commun. Netw.*, vol. 5, no. 3, pp. 595–607, Sep. 2019.

[7] J. Liao, B. Li, and W. Wu, "Wireless powered secure MEC system with intelligent reflecting surface assistance," in *Proc. 10th Int. Conf. Inf. Syst. Comput. Technol.*, 2022, pp. 68–72.

[8] J. Xu, C. Dong, and W. Wen, "Stability-aware control based on power adaptation and energy harvesting in the MEC-WPT system," in *Proc. 4th Int. Conf. Adv. Comput. Technol., Inf. Sci. Commun.*, 2022, pp. 1–6.

[9] R. Malik and M. Vu, "Energy-efficient joint wireless charging and computation offloading in MEC systems," *IEEE J. Sel. Topics Signal Process.*, vol. 15, no. 5, pp. 1110–1126, Aug. 2021.

[10] S. Zhang, H. Gu, K. Chi, L. Huang, K. Yu, and S. Mumtaz, "DRL-based partial offloading for maximizing sum computation rate of wireless powered mobile edge computing network," *IEEE Trans. Wireless Commun.*, vol. 21, no. 12, pp. 10934–10948, Dec. 2022.

[11] S. Yu, B. Dab, Z. Movahedi, R. Langar, and L. Wang, "A socially-aware hybrid computation offloading framework for multi-access edge computing," *IEEE Trans. Mobile Comput.*, vol. 19, no. 6, pp. 1247–1259, Jun. 2020.

[12] M. Li, T. Chen, J. Zeng, X. Zhou, K. Li, and H. Qi, "D2D-assisted computation offloading for mobile edge computing systems with energy harvesting," in *Proc. 20th Int. Conf. Parallel Distrib. Comput., Appl. Technol.*, 2019, pp. 90–95.

[13] D. Wu, F. Wang, X. Cao, and J. Xu, "Joint communication and computation optimization for wireless powered mobile edge computing with D2D offloading," *J. Commun. Inf. Netw.*, vol. 4, no. 4, pp. 72–86, 2019.

[14] C. Ling, W. Zhang, H. He, R. Yadav, J. Wang, and D. Wang, "QoS and fairness oriented dynamic computation offloading in the internet of vehicles based on estimate time of arrival," *IEEE Trans. Veh. Technol.*, vol. 73, no. 7, pp. 10554–10571, Jul. 2024.

[15] C. Shang, Y. Sun, H. Luo, and M. Guizani, "Computation offloading and resource allocation in NOMA–MEC: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 10, no. 17, pp. 15464–15476, Sep. 2023.

[16] 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.

[17] J. Wang, J. Hu, G. Min, A. Y. Zomaya, and N. Georgalas, "Fast adaptive task offloading in edge computing based on meta reinforcement learning," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 1, pp. 242–253, Jan. 2021.

[18] X. Jiao et al., "Deep reinforcement learning for time-energy tradeoff online offloading in MEC-enabled industrial internet of things," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 6, pp. 3465–3479, Nov./Dec. 2023.

[19] S. Birhanu Engidayehu, T. Mahboob, and M. Young Chung, "Deep reinforcement learning-based task offloading and resource allocation in mec-enabled wireless networks," in *Proc. 27th Asia Pacific Conf. Commun.*, 2022, pp. 226–230.

[20] Q. He, Y. Lv, L. Zhen, and K. Yu, "Reinforcement learning based mec architecturewith energy-efficient optimization for arans," in *Proc. IEEE Int. Conf. Commun.*, 2022, pp. 1–6.

[21] S. Nath and J. Wu, "Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems," *Intell. Converged Netw.*, vol. 1, no. 2, pp. 181–198, 2020.

[22] J. Zhang, J. Du, Y. Shen, and J. Wang, "Dynamic computation offloading with energy harvesting devices: A hybrid-decision-based deep reinforcement learning approach," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9303–9317, Oct. 2020.

[23] Y. Li, S. Wang, G. Xu, and L. Ma, "Deep Q-learning enabled energy-efficient resource allocation and task deployment for MEC," in *Proc. 2022 IEEE Smartworld, Ubiquitous Intell. Comput., Scalable Comput. Commun., Digit. Twin, Privacy Comput., Metaverse, Auton. Trusted Veh. (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta)*, 2022, pp. 1259–1265.

[24] R. Malik and M. Vu, "On-request wireless charging and partial computation offloading in multi-access edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 20, no. 10, pp. 6665–6679, Oct. 2021.

[25] F. Wang, H. Xing, and J. Xu, "Real-time resource allocation for wireless powered multiuser mobile edge computing with energy and task causality," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 7140–7155, Nov. 2020.

[26] D. Wu, F. Wang, X. Cao, and J. Xu, "Wireless powered user cooperative computation in mobile edge computing systems," in *Proc. 2018 IEEE Globecom Workshops*, 2018, pp. 1–7.

[27] M. Wu, Q. Song, L. Guo, and I. Lee, "Energy-efficient secure computation offloading in wireless powered mobile edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6907–6912, May 2023.

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

[29] K. Zheng, G. Jiang, X. Liu, K. Chi, X. Yao, and J. Liu, "DRL-based offloading for computation delay minimization in wireless-powered multi-access edge computing," *IEEE Trans. Commun.*, vol. 71, no. 3, pp. 1755–1770, Mar. 2023.

[30] N. C. Luong et al., "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3133–3174, Fourth Quarter 2019.

[31] H. Cui, D. Wang, Q. Li, X. Liu, and H. Lu, "A2C deep reinforcement learning-based MEC network for offloading and resource allocation," in *Proc. 7th Int. Conf. Comput. Commun.*, 2021, pp. 1905–1909.

[32] X. Deng, J. Yin, P. Guan, N. N. Xiong, L. Zhang, and S. Mumtaz, "Intelligent delay-aware partial computing task offloading for multiuser industrial internet of things through edge computing," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 2954–2966, Feb. 2023.

[33] S. Zhang, S. Bao, K. Chi, K. Yu, and S. Mumtaz, "DRL-based computation rate maximization for wireless powered multi-AP edge computing," *IEEE Trans. Commun.*, vol. 72, no. 2, pp. 1105–1118, Feb. 2024.

[34] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, and S. Yu, "Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14198–14211, Dec. 2020.

[35] J. Du et al., "Resource pricing and allocation in MEC enabled blockchain systems: An A3C deep reinforcement learning approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 1, pp. 33–44, Jan./Feb. 2022.

[36] J. Li, X. You, and J. Zheng, "Performance modeling and analysis of an MEC system with task priority and expiring time constraint," *IEEE Commun. Lett.*, vol. 27, no. 7, pp. 1754–1758, Jul. 2023.

[37] A. Gao, S. Zhang, Y. Hu, W. Liang, and S. X. Ng, "Game-combined multi-agent DRL for tasks offloading in wireless powered mec networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 7, pp. 9131–9144, Jul. 2023.

[38] F. Wang, J. Xu, and S. Cui, "Optimal energy allocation and task offloading policy for wireless powered mobile edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2443–2459, Apr. 2020.

[39] X. Pei, W. Duan, M. Wen, Y.-C. Wu, H. Yu, and V. Monteiro, "Socially aware joint resource allocation and computation offloading in noma-aided energy-harvesting massive IoT," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5240–5249, Apr. 2021.

[40] B. Fan, Y. Jiang, F.-C. Zheng, M. Bennis, and X. You, "Social-aware cooperative caching in fog radio access networks," in *Proc. IEEE Int. Conf. Commun.*, 2022, pp. 1672–1677.

[41] J. Wang, J. Hu, G. Min, W. Zhan, A. Y. Zomaya, and N. Georgalas, "Dependent task offloading for edge computing based on deep reinforcement learning," *IEEE Trans. Comput.*, vol. 71, no. 10, pp. 2449–2461, Oct. 2022.

[42] L. An, Z. Wang, J. Yue, and X. Ma, "Joint task offloading and resource allocation via proximal policy optimization for mobile edge computing network," in *Proc. Int. Conf. Netw. Netw. Appl.*, 2021, pp. 466–471.

[43] X. Mi and H. He, "Multi-agent deep reinforcement learning for D2D-assisted MEC system with energy harvesting," in *Proc. 25th Int. Conf. Adv. Commun. Technol.*, 2023, pp. 145–153.

[44] K. Du, X. Xie, Z. Shi, and M. Li, "Joint time and power control of energy harvesting CRN based on PPO," in *Proc. 2022 Wireless Telecommun. Symp.*, 2022, pp. 1–6.

[45] F. Zishen, X. Xianzhong, S. Zhaoyuan, and C. Xiping, "Proximal policy optimization based continuous intelligent power control in cognitive radio network," in *Proc. IEEE 6th Int. Conf. Comput. Commun.*, 2020, pp. 820–824.

Xinyuan Zhu received the BSc degree in Internet of Things engineering from Zhengzhou University, China, in 2022. She is currently working toward the MSc degree with the School of Computer Science, Shaanxi Normal University, China. She is currently interested in the research of Mobile Edge Computing, Cloud Computing, and Internet of Things.

Fei Hao received the PhD degree in computer science and engineering from Soonchunhyang University, South Korea, in 2016. From 2020 to 2022, he was a Marie Curie Fellow with the University of Exeter, Exeter, U.K. He is currently a professor with the School of Computer Science, Shaanxi Normal University, Xi'an, China. He has published more than 150 papers in the leading international journals and conference proceedings, such as *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Network Science and Engineering*, *IEEE Communications Magazine*, *IEEE Internet Computing*, *ACM Transactions on Multimedia Computing, Communications, and Applications*, ACM SIGIR, and IEEE GLOBECOM. His research interests include social computing, soft computing, big data analytics, pervasive computing, and data mining. He is also a member of ACM, CCF, and KIPS. In addition, he was a recipient of the Best Paper Award from IEEE GreenCom 2013. He was a recipient of the Outstanding Service Award with IEEE DSS 2018, the IEEE SmartData 2017, the IEEE Outstanding Leadership Award with IEEE CPSCom 2013, and the 2015 Chinese Government Award for Outstanding Self-Financed Students Abroad.

Lianbo Ma received the BSc degree in communication engineering and the MSc degree in communication and information system from Northeastern University, Shenyang, China, in 2004 and 2007, respectively, and the PhD degree from University of Chinese Academy of Sciences, China, in 2015. He is currently a professor of Northeastern University, China. His current research interests include computational intelligence and machine learning. He has published more than 100 journal articles, books, and refereed conference papers.

Changqing Luo (Member, IEEE) received the BE and ME degrees from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2004 and 2007, respectively, and the PhD degrees from Case Western Reserve University, Cleveland, OH, USA and Beijing University of Posts and Telecommunications, Beijing, China, in 2018 and 2011, respectively. He is an assistant professor with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA, USA. His research interests include cybersecurity and complex networks.

Geyong Min received the BSc degree in computer science from the Huazhong University of Science and Technology, China, in 1995 and the PhD degree in computing science from the University of Glasgow, United Kingdom, in 2003. He is a professor of High Performance Computing and Networking in the Department of Computer Science within the College of Engineering, Mathematics and Physical Sciences with the University of Exeter, U.K. His research interests include Computer Networks, Wireless Communications, Parallel and Distributed Computing, Ubiquitous Computing, Multimedia Systems, Modelling and Performance Engineering.

Laurence T. Yang received the BE degree in computer science and technology and the BSc degree in applied physics both from Tsinghua University, China, and the PhD degree in computer science from the University of Victoria, Canada. He is a professor with the School of Computer Science and Artificial Intelligence, Zhengzhou University, China, and with the Department of Computer Science, St. Francis Xavier University, Canada. His research interests include cyber-physical-social systems, parallel and distributed computing, embedded and ubiquitous/pervasive computing, and big data. His research has been supported by the National Sciences and Engineering Research Council, Canada, and the Canada Foundation for Innovation.