applied sciences

Article

PPO-Based Joint Optimization for UAV-Assisted Edge Computing Networks

Zhihui Liu [1], Qiwei Zhang [1] and Yi Su [2,3,*]

[1] State Key Laboratory of Space-Ground Integrated Information Technology, Space Star Technology Co., Ltd., Beijing 100095, China; liuzhihui1225@163.com (Z.L.); qwzhang95@163.com (Q.Z.)
[2] College of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400044, China
[3] Key Laboratory of Flight Techniques and Flight Safety, CAAC, Guanghan 618307, China
* Correspondence: suyi@cqu.edu.cn

Abstract: In next-generation mobile communication scenarios, more and more user terminals (UEs) and edge computing servers (ECSs) are connected to the network. To ensure the experience of edge computing services, we designed an unmanned aerial vehicle (UAV)-assisted edge computing network application scenario. In the considered scenario, the UAV acts as a relay node to forward edge computing tasks when the performance of the wireless channel between UEs and ECSs degrades. In order to minimize the average delay of edge computing tasks, we design the optimization problem of joint UE–ECS matching and UAV three-dimensional hovering position deployment. Further, we transform this mixed integer nonlinear programming into a continuous-variable decision process and design the corresponding Proximal Policy Optimization (PPO)-based joint optimization algorithm. Sufficient data pertaining to latency demonstrate that the suggested algorithm can obtain a seamless reward value when the number of training steps hits three million. This verifies the algorithm's desirable convergence property. Furthermore, the algorithm's efficacy has been confirmed through simulation in various environments. The experimental findings ascertain that the PPO-based co-optimization algorithm consistently attains a lower average latency rate and a minimum of 8% reduction in comparison to the baseline scenarios.

Keywords: edge computing; unmanned aerial vehicle; proximal policy optimization; joint optimization

check for updates

Citation: Liu, Z.; Zhang, Q.; Su, Y. PPO-Based Joint Optimization for UAV-Assisted Edge Computing Networks. *Appl. Sci.* **2023**, *13*, 12828. https://doi.org/10.3390/app132312828

Academic Editors: Yujie Tang and Shu Fu

Received: 18 October 2023
Revised: 27 November 2023
Accepted: 28 November 2023
Published: 29 November 2023

1. Introduction

With the rapid development of mobile communication, the number of devices and the amount of application data at the network edge have significantly increased [1–3]. However, traditional cloud computing architectures require centralized processing of all data, which cannot meet the requirements of low latency, high traffic volume, and high reliability in current wireless communications. To overcome these issues, edge computing has been proposed as one of the key technologies in 5G application scenarios. Unlike centralized processing, edge computing enables user devices to offload computation tasks directly to edge computing servers (ECSs), reducing the data traffic on the network bandwidth and improving system responsiveness, thus better meeting the low-latency and high-reliability service requirements of 5G data networks [4–7]. By deploying ECSs, which act as micro-clouds, in ground infrastructure, edge computing brings computing resources closer to service users, reducing latency and energy consumption in long-distance data transmission processes. User devices offload computation-intensive and latency-sensitive tasks generated by local applications to ECSs that are closer in proximity, thereby improving computational efficiency and ensuring a better user service experience. It is crucial to acknowledge that dependable and unwavering edge computing services necessitate exceedingly strict demands on the offloading scheme of computing tasks. Inappropriate pairing of users with edge computing servers will result in elevated energy consumption and prolonged waiting latency, consequently reducing edge computing network throughput. Within the confines of limited edge servers and deployment locations, enhancing the flexibility of the edge computing network while effectively offloading computing tasks poses a significant challenge for the current state of edge computing.

It should be noted that, in edge computing, deployable ECSs play a crucial role. However, due to the high cost of hardware, it can be challenging to meet the business experience needs of all users with a limited number of ECSs deployed in fixed locations. This is especially true when wireless links are obstructed by buildings, mountains, and other obstacles, severely affecting the wireless link between user terminals (UEs) and ECSs. In recent years, unmanned aerial vehicles (UAVs) have been deployed in wireless communication networks due to their advantages in mobility and cost. UAVs can be used as mobile relay nodes to facilitate information exchange between distant users and as mobile base stations to enhance the coverage range of wireless networks. By leveraging the wide coverage, low cost, and flexibility of UAVs, they can be used as relay nodes in edge computing applications to effectively address the issue of degraded link quality, thereby ensuring a satisfactory user experience at the UE. However, the performance of UAV-assisted edge computing is directly related to the deployment location. To ensure a good user experience in edge computing, the deployment location of UAVs needs to be precisely designed. Furthermore, the incorporation of UAVs will impact the selection of deployment tactics, which is a convoluted process. As opposed to conventional terrestrial mobile edge computing networks, the integration of unmanned aerial vehicles enhances network flexibility, yet simultaneously introduces fresh obstacles to network deployment and decision-making. Joint optimization of computation offloading and drone deployment in drone-assisted edge computing networks, considering heterogeneous user demands and resource constraints, will be critical for enhancing the performance of edge computing networks.

Currently, many resource allocation problems in mobile communication networks are primarily solved using convex optimization, game theory, and other methods. Typically, these methods require the objective function or problem model to conform to specific forms. Moreover, the spatial scale of traditional solution methods exponentially increases with the variables of the problem, requiring more time to converge or even making the solution infeasible. Recently, reinforcement learning methods based on Markov decision processes have been proposed to solve dynamic decision problems under unknown models. In the absence of statistical information about the environment, an agent can learn effective strategies by continuously interacting with the environment, exhibiting a certain level of decision-making ability. Furthermore, deep learning neural networks possess perception capabilities, meaning they can effectively approximate functions. By combining the decision-making ability of reinforcement learning with the perception capabilities of neural networks in deep learning, i.e., deep reinforcement learning (DRL), an agent can make decisions based on local observations without prior information, interact with the environment, and adjust its strategy based on environmental feedback. Through long-term dynamic exploration and training, the agent can effectively learn a strategy to achieve the best long-term goal. DRL combines the perception capabilities of deep learning with the decision-making ability of reinforcement learning, enabling the resolution of complex high-dimensional state space decision problems. Therefore, deep reinforcement learning holds promise for addressing complex decision-making problems in UAV-assisted edge computing applications.

2. Related Work

2.1. State of Art

Edge Computing: Currently, research into the application of mobile edge computing has received significant attention [8–11]. The work in [9] investigates computation offloading and resource allocation issues in single ECS scenarios. The work in [12] proposes an online joint radio and computation resource management algorithm for the scenario of multiple mobile devices and a single ECS. By optimizing power and bandwidth allocation, it minimizes the long-term average weighted sum of mobile device and ECS power consumption. The work in [8] studies distributed computation offloading in a wireless-powered, multi-user, single-edge computing system. A game-theoretic computation offloading scheme is constructed to optimize the weighted sum of energy harvesting and offloading delay. Moreover, a two-stage joint optimization scheme was proposed for mobile edge computing using federated learning in [13]. Similar to federated learning, the work in [14] presents an edge computing collaboration and offloading optimization problem for underwater sensor networks. The study designed an artificial neural network model to solve this issue. When exploring the use of mobile edge computing in vehicular networking, several sources have investigated the associated challenges. For example, the work in [15,16] has focused on mobility issues specifically related to vehicular networking edge computing, proposing solutions aimed at addressing the demands of latency-sensitive and computation-intensive in-vehicle applications. Additionally, the work in [11,17–20] examines concerns surrounding load computation and resource allocation in multi-ECS scenarios. However, it is important to note that geographical limitations may impede the establishment of wireless connections between UEs and ECSs. The aforementioned works offer a comprehensive exploration of conventional terrestrial edge computing networks. Further research on mobile edge computing networks is required, with a primary focus on the wireless channel quality and various practical constraints.

UAV-Assisted Edge Computing: To address the issue of wireless connectivity arising from fixed deployment setups, UAVs have been incorporated into mobile edge computing due to their adaptable deployment attributes. There have been studies conducted on the viability of this approach. In the research into ECSs with UAV-assisted offloading for ground networks, the existing works primarily focus on addressing the energy consumption issues of UAVs. Seongah et al. [21] propose an ECS model based on UAVs, where a computing-capable UAV provides computation offloading services to ground systems with limited local data processing capabilities. They employ a successive convex approximation approach to minimize the energy consumption of ground applications. Zhou et al. [22] develop an edge computing system that enables wireless power transfer to UAVs and address the causality between computing and energy transfer under power minimization constraints. Hu et al. [23] utilize UAVs as computing servers to provide services to ground terminal devices and act as relays to offload tasks from terminal devices to access node computing. By employing an alternating optimization algorithm, they achieve the minimization of the total energy consumption of devices and UAVs. Zhang et al. [24] consider a three-tiered architecture for a UAV-assisted offloading edge computing system and obtain the optimal solution for the optimization objective using Lagrange duality. In addition, Refs. [25–27] investigate the deployment of UAVs while keeping in mind the energy consumption of edge computing networks. The current research places greater emphasis on the energy consumption of the network, whereas the investigation into latency, a crucial factor for meeting user requirements, requires further development. Furthermore, the integration of drones creates a connection between the offloading decision-making process of mobile edge computing and the drone deployment problem, which is frequently disregarded. The optimization of drone-assisted edge computing networks is vital for network performance and necessitates thorough research.

DRL in Edge Computing: Currently, the research into applying deep reinforcement learning to address the computation offloading and resource allocation issues in edge computing systems primarily focuses on the decision-making of computation offloading [27]. Ref. [28] proposes a deep Q-learning-based offloading scheme for IoT devices with energy harvesting to optimize the server selection and discrete action of the offloading rate. Ref. [29] introduces a method based on Deep Q Network (DQN) to estimate the number of tasks processed locally, at ECSs, and at cloud servers. This method improves the privacy level of user devices while reducing offloading costs such as latency and energy consumption. A solution based on the deep deterministic policy gradient is proposed to optimize the power allocation for continuous local execution and offloading. Ref. [30]

considers multiple users in edge computing systems, where multiple user devices can offload computations to ECSs through wireless channels. By discretizing the continuous offloading decision and allocating computing resources, this method minimizes the total delay cost and energy consumption of all terminals in the edge computing system. Note that existing work typically considers discrete variable planning or integer planning for edge computing optimization models. However, this differs significantly from the challenges presented in UAV-assisted edge computing networks, where the problem includes both discrete and integer variables. Unfortunately, current deep reinforcement learning methods are inadequate for scenarios where these variables intersect. Therefore, the design of a novel optimization framework that can cope with UAV-assisted edge computing networks necessitates in-depth discussion.

2.2. Motivation and Contribution

In the context of next-generation mobile communication scenarios, an increasing number of UEs and ECSs are being connected to the network. At the same time, users have higher demands for their service experience. However, current research on edge computing lacks consideration of the decision-making of multiple UEs and ECSs that cannot meet the increasing demands of wireless network traffic. Additionally, existing works on UAVs in edge computing lack comprehensiveness, and further research is needed on the joint deployment of UAVs and edge computing decision-making. Furthermore, the emergence of deep reinforcement learning provides a new research direction for UAV-assisted edge computing. Motivated by the above observations, to improve the flexibility and timeliness of UAV-aided edge computing, this work designs a UAV-assisted edge computing network and further proposes an optimization problem for joint UAV deployment and edge computing decision-making. Additionally, in order to address the mixed variable optimization that exists in current UAV-assisted mobile edge computing networks, a deep reinforcement learning algorithm based on Proximal Policy Optimization (PPO) is proposed to solve the optimization problem. Specifically, our contributions are as follows:

- This work designs a UAV-assisted edge computing network in which UAVs act as relay nodes to forward edge computing tasks when the wireless channel between UEs and ECSs is degraded. The devised network can proficiently tackle the issues of restricted wireless links and poor flexibility present in stationary ECS deployment settings, consequently encouraging the employment and evolution of mobile edge computing via UAVs.
- To meet the requirements of service experience, this work aims to minimize the average delay and designs an optimization problem for joint three-dimensional UAV deployment and scheduling decision-making of UEs and ECSs in edge computing. The model addresses the connection between decisions regarding the deployment and unloading of UAVs, effectively mitigating potential network performance issues arising from the interaction of multiple variables.
- This work proposes a joint optimization algorithm based on PPO within the framework of Deep Reinforcement Learning (DRL) to solve the complex optimization problem in the UAV-assisted edge computing network. The method proposed has the ability to handle situations in which discrete and continuous variables are present and is in line with UAV-aided edge computing networks. This is unlike traditional algorithms or existing DRL algorithms.
- Experimental results validate the effectiveness of the proposed PPO-based joint optimization algorithm. Compared to baseline algorithms, the PPO-based joint optimization algorithm achieves a lower average delay and better meets the requirements of user service experience.

The rest of the paper is organized as follows. In Section 3, the considered system model and joint optimization problem are established. In Sections 4 and 5, the PPO-based joint optimization algorithm is designed, and the performance is verified by numerical results, respectively. Finally, Section 6 concludes the paper.

3. System Model and Problem Formulation

In this section, we design a UAV-assisted multi-node edge computing network as shown in Figure 1. The considered system has K UEs, M ECSs, and a UAV, where each UE has a certain computational demand, each ECS has a certain processing capability, and the UAV has only a communication function without computational capability. In this network, the UE cannot establish a good direct wireless link with the edge calculator due to the obstruction of terrain and buildings. In order to satisfy the computational needs of the UE, a UAV is deployed as a relay node in the network. The computation task of the UE is first sent to the UAV and, after a reasonable selection, is forwarded to one of the ECSs. In particular, the matching of UEs and ECSs is crucial for the service experience of UEs. Through reasonable design, the transmission delay and waiting delay of the computation tasks can be ensured in an acceptable range. At the same time, UAVs are highly maneuverable, and by adjusting the hovering position of UAVs, the latency of the computation task can also be effectively reduced. Compared to the existing work [13–15], this scenario focuses on possible problems with the wireless link and compensates for the degradation of the channel quality by means of UAVs. The proposed scheme can be adapted to application scenarios such as large cities with heavy building occlusion and mountains with geographic occlusion.

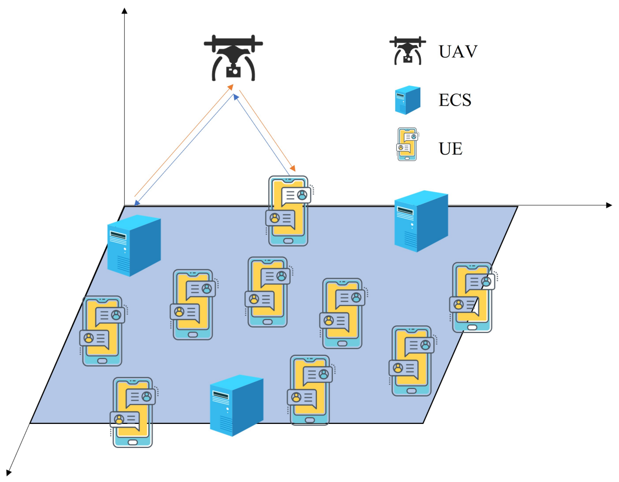

Figure 1. System model.

In this work, we adopt the probabilistic line-of-sight channel model. The wireless transmission channel between the UE and the UAV can be categorized by line-of-sight (LoS) and non-line-of-sight (NLoS). The probability of having a LoS wireless channel between UE and UAV is

$$P_k^{\mathrm{Los}} = \frac{1}{1 + a\exp\left(-b\left(\frac{180}{\pi}\tan^{-1}\left(\frac{z}{d_k}\right) - a\right)\right)}, \tag{1}$$

where a and b are constants determined by the environment, and z is the hovering height of the UAV. Denote the horizontal coordinates of the UAV by (x_u, y_u), and that of UE k by (x_k, y_k). The distance from UE k to the UAV in the horizontal plane is expressed as

$$d_k = \sqrt{(x_u - x_k)^2 + (y_u - y_k)^2}. \tag{2}$$

Then, the probability of having an NLoS wireless channel between the UE and the UAV is $P_k^{\mathrm{NLos}} = 1 - P_k^{\mathrm{Los}}$.

Let f_c denote the carrier frequency, and use c to denote the electromagnetic velocity in free space. Let ρ_{Los} and ρ_{NLos} denote the average additional path loss for LoS and NLoS channels, respectively. Then, the path loss of LoS channel and NLoS channel can be expressed respectively as

$$\zeta_{\text{Los}} = 20\log\left(\frac{4\pi f_c\sqrt{z^2+d_k^2}}{c}\right) + \rho_{\text{Los}}, \tag{3}$$

$$\zeta_{\text{NLos}} = 20\log\left(\frac{4\pi f_c\sqrt{z^2+d_k^2}}{c}\right) + \rho_{\text{NLos}}. \tag{4}$$

Then, the probabilistic average path loss from UE k to UAV is

$$\zeta(k) = \zeta_{\text{Los}} \times P_{\text{Los}} + \zeta_{\text{NLos}} \times P_{\text{NLos}} \tag{5}$$

$$= \frac{A}{1 + a\exp\left(-b\left(\frac{180}{\pi}\tan^{-1}\left(\frac{z}{d_k}\right) - a\right)\right)} + 20\log\left(\sqrt{z^2+d_k^2}\right) + \tilde{A}, \tag{6}$$

where $A = \rho_{\text{Los}} - \rho_{\text{NLos}}$, $\tilde{A} = 20\log\left(\frac{4\pi f_c}{c}\right) + \rho_{\text{Los}}$. Define the transmit power as p, the bandwidth as B, and the variance of add white Gaussian noise (AWGN) as σ^2; then, the maximum transmit rate at UE k can be expressed as

$$R_k = B\log\left(\frac{p \times 10^{-\frac{\zeta(k)}{10}}}{\sigma^2}\right). \tag{7}$$

From Figure 1, it can be seen that the computational task from generation to completion contains a total of sending delay, propagation delay, waiting delay, and processing delay, which can be expressed by the following equation:

$$Delay_{total} = Delay_{tran} + Delay_{porp} + Delay_{wait} + Delay_{comp}. \tag{8}$$

Considering that the range of this edge computing network is usually small and the signal propagation speed is fast, the propagation delay $Delay_{porp}$ in it is negligible. The sending delay, on the other hand, is determined by the size of the computing task and the sending rate. Suppose that end user k generates a computation task with task size L_k. In the case that the sending rate of the UE k is R_k^{UE}, and the sending rate is R_k^{UAV} when the UAV relays, the sending delay of this computational task can be expressed as

$$Delay_{tran}^k = \frac{L_k}{R_k^{UE}} + \frac{L_k}{R_k^{UAV}}, \tag{9}$$

where the sending delay in the link from the ECS to the UE is ignored, considering that the computation results returned from the ECS are usually small. Define the ECS matched by UE k as m, and define its corresponding computing power as C_m; then, the processing delay of the computing task of UE k can be obtained as

$$Delay_{comp}^k = \frac{L_k}{C_m}. \tag{10}$$

The waiting delay indicates the time that this computation task waits in the ECS to be processed. Consider a business cycle in which all UEs simultaneously send computation tasks to the UAV. The UAV is equipped with multiple antennas to forward all the computation tasks simultaneously. In the considered system, the first computational task to arrive at the ECS is the one with the smallest task size among the computational tasks matched by the ECS, and its waiting delay is 0. Subsequent computational tasks arriving at the ECS are categorized into two scenarios based on whether the previous computational task has been processed or not. When the previous task has been processed, the waiting delay of the task is also 0. Otherwise, the waiting delay of the task is accumulated from the time it arrives at the ECS until the previous task has been processed. Based on the above derivation, the waiting delay of a computation task of UE k at ECS m can be expressed as

$$Delay_{wait}^k = \begin{cases} 0, L_k = \min \mathcal{L}_m \ or \ T_{done}^j \le T_{arri}^k \\ T_{done}^j - T_{arri}^k, T_{done}^j > T_{arri}^k \end{cases}, \tag{11}$$

where $\mathcal{L}_m$ represents the set of computational tasks at ECS m, j represents the computational task that is second only to k in terms of task volume, and if j does not exist, then $T^j_{done} \leq T^k_{arri}$ holds forever. T^j_{done} and T^k_{arri} denote the moment when the processing of computational task j is completed and the arrival moment of computational task k, respectively.

Based on the above derivation, we can obtain the total delay of the computation task for user k as

$$Delay^k_{total} = \frac{L_k}{R^{UE}_k} + \frac{L_k}{R^{UAV}_k} + Delay^k_{comp} + Delay^k_{wait}. \tag{12}$$

Considering the entire edge computing network, we use the average delay of the network as a criterion for evaluating the experience of edge computing services, which can be expressed as

$$Delay^{sum}_{total} = \frac{1}{K}\sum_{k=1}^{K} Delay^k_{total}. \tag{13}$$

Our goal is to minimize the average delay of edge computing networks. In the considered UAV-assisted edge computing network, the matching of UEs and ECSs has a crucial impact on the waiting delay and processing delay. Also, the choice of UAV hovering position determines the maximum sending rate and affects the sending delay. Therefore, we consider establishing the optimization problem for joint UE–ECS matching and deployment of three hovering positions of UAVs. Specifically, the problem is modeled as follows:

$$\mathbf{P1}: \quad \min_{\mathbf{q},\mathbf{u}} \frac{1}{K}\sum_{k=1}^{K} Delay^k_{total} \tag{14a}$$

$$\text{s.t.} \quad \mathbf{q} \in \mathbf{q}_c, \tag{14b}$$

$$u_k \in \{1, 2, \cdots, M\}, k = 1, 2, \cdots, K \tag{14c}$$

$$|u_k| = 1, k = 1, 2, \cdots, K \tag{14d}$$

where the optimization variables are the three UAV deployment locations, $\mathbf{q}$, and the matching of the UEs, $\mathbf{u}$. The first constraint is that the UAV needs to hover within a reasonable range, $\mathbf{q}_c$; the second constraint is that the user needs to select the ECSs present in the network for matching, in which u_k represents the matching of the UEs, k; and the last constraint is that each edge user can select only one ECS for matching.

4. PPO-Based Joint Optimization Algorithm

We provided the optimization problem for joint UE–ECS matching and deployment of UAVs with three hovering positions in the previous section. However, it can be observed by looking at the problem model that the problem is a complex mixed integer nonlinear programming problem, which is usually difficult to solve by traditional optimization methods. Fortunately, the advancement of artificial intelligence has introduced novel research techniques for mobile edge computing. PPO has gained widespread acceptance as a reinforcement learning algorithm due to its stability and reliability. In contrast to traditional policy gradient algorithms and other RL algorithms based on the actor-critic framework, PPO incorporates an older actor network to constrain the variance of the new policy. This approach effectively mitigates instability issues that could potentially arise in other algorithms. Secondly, the PPO algorithm can consistently perform well in various environments and can effectively suit the UAV-assisted edge computing environment that is being studied. Hence, in this work, we consider designing solution algorithms for joint optimization based on the PPO framework. Specific algorithmic details are given next.

Note that deep reinforcement learning frameworks, including PPO, can usually only handle discrete or continuous decision problems. However, the joint optimization problem in this work is mixed integer programming with both discrete and continuous variables, which is difficult to handle for PPO. Therefore, we first introduce matching factors to convert the original mixed integer programming into a continuous variable problem. We design M matching factors for each UE, which represent the matching priority of each of the M ECSs at that UE. The value range of the matching factors is $[0, 1]$. Following this design principle, the matching factors of UE k are $o_1^k, o_2^k, \cdots, o_M^k$; then, the ECS matched by UE k is the one with the largest matching factor, denoted by

$$\arg\max\left(o_1^k, o_2^k, \cdots, o_M^k\right). \tag{15}$$

So far, the original integer variable $\mathbf{u}$ is transformed into a continuous variable, i.e., the matching factor, and the original problem is transformed into a continuous-variable decision process that can be solved by the PPO implementation. We first model the problem as a Markov decision process, where the state space, action space, state transfer function, and reward function are described as follows.

- *State space*: the environment is described to the agent body by the state space, which needs to provide enough information to make the agent body take appropriate actions. For the considered UAV-assisted edge computing network, the hovering position of the UAV determines the sending delay, while the matching of the UE and the ECS determines the processing delay and the waiting delay. Therefore, the state space designed in this work contains the following points: the three-dimensional hovering position of the UAV and the matching factor at all UEs. The state space is defined as $\mathcal{S}$, and the state at time step t is defined as s_t.
- *Action space*: based on the decision variables and state space in the problem model, we design the action space for the problem. The action space also contains two parts: one is the displacement of the UAV in three-dimensional space, and the other is the amount of change of the matching factor. The action space is defined as $\mathcal{A}$, and the state at time step t is defined as a_t.
- *State transfer probability function*: according to the state and action of the agent body designed above, the next state of the agent body can be obtained from its current state plus the action, i.e., the current three-dimensional position of the UAV base station plus the displacement to the next three-dimensional position. Therefore, the state transfer probability function is defined as

$$\mathcal{P}(s_{t+1}|s_t, a_t) = \begin{cases} 1, s_{t+1} = s_t + a_t, s_t \in \mathcal{S}, a_t \in \mathcal{A} \\ 0, \text{others} \end{cases}. \tag{16}$$

- *Reward function*: the designed PPO-based joint optimization algorithm aims to minimize the average delay of the UAV-assisted edge computing network. Therefore, the average delay is designed as the body of the reward function. Meanwhile, in order to make the agent body transfer minimize the average delay, the average delay exists in the form of the opposite number in the reward function. Specifically, the reward function is expressed as

$$r_t = p\left(q - \left(\frac{1}{K}\sum_{k=1}^{K} Delay_{total}^k | t\right)\right), \tag{17}$$

where p and q are constant terms to adjust the reward value to within a suitable range.

Based on the modeling of the Markov decision process, the agent body can perform actions to obtain the reward value and complete the state transfer. The agent body in the PPO algorithm collects the state transfer trajectories for every T time steps, then performs a round of updating. Each round of updating uses the group of trajectory data to update D times, and the object of the updating is the parameter θ of the Actor network and the parameter ϕ of the Critic network.

The role of the Actor network is to fit the agent's policy $\pi_\theta(a|s)$, representing the policy in the form of a Gaussian distribution, which can be fully described by its mean μ and standard deviation σ. Then, the Actor network completes the state to (μ, σ) mapping. When the intelligence needs to take an action, it first recovers that Gaussian distribution through (μ, σ) and then randomly samples to pick an action. The PPO, in order to keep the old policy unchanged in each round of update, initializes a network with the same structure and parameters as the Actor network, called Actor-old. The Actor-old network does not participate in training and only copies parameters from the Actor network before each round of updates to keep the old strategy π_{θ_k} for the current round; then, the Actor network can continue searching for the new strategy $\pi_{\theta_{k+1}}$. The Actor network is trained to maximize the agent objective function, whose parameters are given by the following equation update

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{T} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A_t^{\pi_{\theta_k}}, \text{clip}\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon \right) A_t^{\pi_{\theta_k}} \right), \quad (18)$$

where k denotes the first round of update and $A_t^{\pi_{\theta_k}}$ is the dominance function calculated based on the state value estimated by the Critic network and the temporal difference method, where R_t and $V(s_t)$ are the reward value of the time step and the state value of s_t, respectively.

The Critic network completes the estimation of the state value $V(s)$, and its training objective is to minimize the loss function based on the mean square error. Then, its parameter ϕ is updated by the following equation:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{T} \sum_{t=0}^{T} \left(V_\phi(s_t) - \widehat{R}_t \right)^2, \quad (19)$$

where $\widehat{R}_t$ is the accumulated reward computed from the state transfer trajectory data. Based on the above neural network design, the basic framework of PPO is built.

According to the above description, both the main framework and Markov decision process of the PPO algorithm have been designed. Next, we reduce the average delay of the network by jointly optimizing the UE–ECS matching and the UAV three-dimensional hovering position. The designed optimization algorithm is called the PPO-based joint optimization algorithm, and the detailed algorithm flow is shown in Algorithm 1.

Algorithm 1 PPO-based joint optimization algorithm

```
1:  Initialize the parameters $\theta$ and $\phi$ of the Actor network and Critic estimation network;
    initialize the Actor-old network.
2:  for each training round do
3:    Initialize the three-dimensional position of the UAV and the matching factor.
4:    for each time step $t$ do
5:      UAV base station observes its own three-dimensional position and reads matching
        factor as state $s_t$.
6:      Actor inputs $s_t$ and outputs a Gaussian distribution for the strategy; the agent
        body recovers the strategy and randomly selects an action $a_t$.
7:      Calculate the average delay of the system and obtain the reward value.
8:      Agent body implements action $a_t$ and updates to the next state $s_{t+1}$.
9:      Collect the state transfer trajectory $(s_t, a_t, r_t)$ into the cache.
10:     Based on the output of the Critic network and the reward value of each time step
        in the cache, backtrack to calculate the state value of the corresponding time step.
11:     Copy the parameters of the Actor network to the Actor-old network to maintain
        the old policy.
12:     for $1 : D$ do
13:       Update the parameters of Actor network and Critic network.
14:     end for
15:   end for
16: end for
17: Return: UAV hover position, match factor, and average delay.
```

5. Simulation Results

In this work, we simulate and validate the designed UAV-assisted edge computing network. We consider the presence of $K = 9$ UEs and $M = 3$ ECSs in a square cell of length 400 m. The UAV hovers over the cell and is at an altitude between 40 m and 200 m. The carrier frequency is set to $f_c = 1000$ MHz, the bandwidth is 1 MHz, and the noise power spectral density is -174 dBm/Hz. The air-to-ground channel is set with reference to the commonly used environment, i.e., $a = 9.61$, $b = 0.16$, $\rho_{\mathrm{Los}} = 1$, and $\rho_{\mathrm{NLos}} = 2$. The UEs and ECSs are generated using a hybrid process of uniform randomization plus Gaussian randomization. This is because, if the users were completely uniformly distributed in the region, it would make the planar location deployment of the UAVs not optimized. The PPO-related parameter settings are shown in Table 1.

Table 1. Parameter settings.

Simulation Parameter	Value
Steps per epoch	200
Gamma	0.99
Batch size	64
Clip range	0.2
Learning rate	0.0003
Number of environments	8
Steps per update	2048

We plot the change in reward values during training of the PPO-based joint optimization algorithm in Figure 2. The vertical coordinate is the cumulative reward value per 200 time steps, and the horizontal coordinate is the time step. It can be seen that all three curves increase gradually with the increase in time steps and stabilize at three million time steps. The above changes can effectively illustrate that the proposed PPO-based joint optimization algorithm has good convergence properties. Meanwhile, the trend of the reward increasing with time steps also verifies the effectiveness of the proposed algorithm in UAV-assisted edge computing networks. The 3 curves in the figure correspond to varying transmission power levels, where training processes 1, 2, and 3 correspond to transmit power levels of 1.5 W, 1 W, and 0.5 W. As the figure illustrates, the reward value initially increases with the increase in power, implying that increasing power can achieve better system performance to some extent.

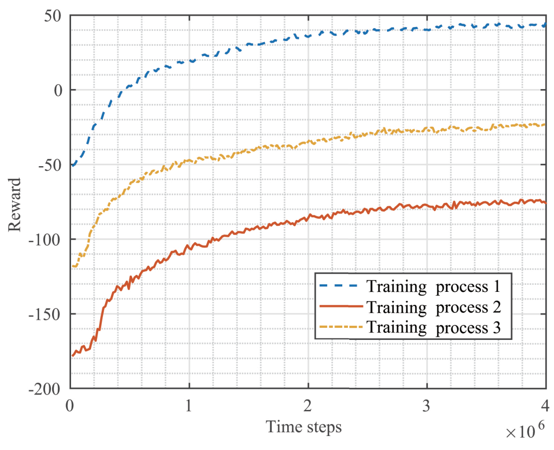

Figure 2. Training curve.

Figure 3 plots the average delay versus transmit power for the UAV-assisted edge computing network, where PBJO stands for the proposed PPO-based joint optimization algorithm. Baseline 1 is the greedy scheme, in which the UEs always choose the ECS closest to the UAV, and the UAV hovers in the center of the cell. Baseline 2 is the stochastic scheme, in which the UEs are randomly matched with the ECSs, and the UAV randomly chooses the hovering location. The figure shows that, as transmit power increases, the average delay typically decreases. This is because higher transmit power facilitates increased communication rates, which ultimately reduces transmission delays. Notably, the effect of transmit power on the average delay in Baseline 1 is minimal. In this scheme, all UEs select the ECS closest to the UAV for matching. This results in an average delay that is dominated by waiting time. However, it has been observed that the proposed scheme consistently achieves a lower average delay, which implies a superior edge computing service experience. For instance, at a transmission power of 1 Watt, the PBJO scheme achieves an average delay of approximately 0.88 s, which is 0.12 and 0.3 s lower than Baseline 1 and Baseline 2, respectively. Although the waiting delay in Baseline 1 is on the high side, its average delay is still lower than that of Baseline 2, which shows the necessity of optimizing the three hovering positions of the UAV.

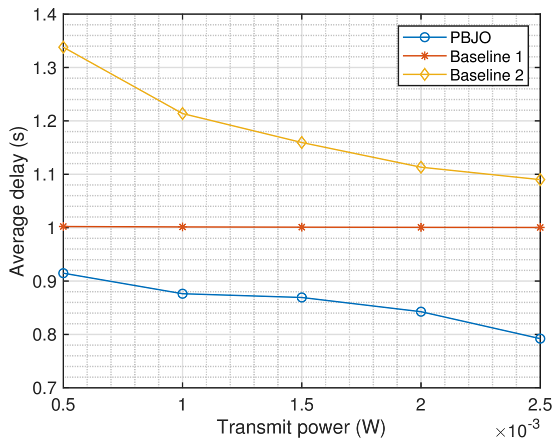

Figure 3. Average delay vs. transmit power.

Figure 4 plots the variation in the average delay of the UAV-assisted edge computing network versus the computational ability of the ECS. The average latency consistently decreases with increased computational power, aligning with our expectations. This is due to the ECS's ability to process tasks more quickly with greater computational power, effectively reducing the waiting latency of the edge computing network. Similarly, the proposed PPO-based joint optimization algorithm achieves a lower average delay compared to Baseline 1 and Baseline 2, which verifies the superiority of the proposed algorithm. At 1 Mbps, PBJO decreases latency by 0.3 s compared to the baseline scenarios, and the performance gap widens with increasing computing power. In addition, as the computational ability increases, we can observe that the trend of the average delay gradually flattens out. This is because the computational ability affects the waiting delay. Therefore, the significance of the delay change due to increasing computing ability gradually decreases relative to the total delay.

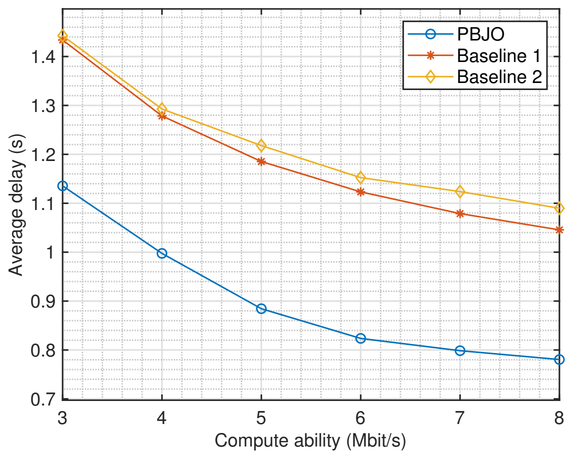

Figure 4. Average delay vs. computing ability.

To further examine how deployment of ECSs affects the UAV-aided edge computing network under consideration, we plotted the changes in the average latency in Figure 5 with the number of ECSs. As anticipated, the average latency consistently decreases with an increase in the number of ECSs. An increase in ECSs will significantly enhance network computation, resulting in a reduction in the average latency. However, it is worth noting that there is a limit to the benefits of increasing ECSs. Specifically, deploying 6 ECSs only results in a reduction of around 0.2 s. When there are five or more ECSs, increasing the number of ECSs does not significantly reduce the average latency. This is because queuing is eliminated in the network when there are sufficient ECSs, and the latency of the network is mainly determined by the sending latency. The above event highlights the significance of logically strategizing the number of ECSs deployed, which is also a notable aspect of edge computing research. Furthermore, it is observed that the PBJO algorithm achieves a lower average delay than the two extreme algorithms, reaffirming its superiority. We have observed a fascinating phenomenon in which the disparity between PBJO and the other baseline algorithms initially rises and then falls in correlation with the number of ECSs. Specifically, with only one ECS, PBJO lags behind the other algorithms by a mere 0.1 s. However, PBJO boasts the most significant improvement when the number of ECSs is 2, achieving a decrease of 0.5 s. As the number of ECSs hits 6, the gap between the algorithms goes back to 0.1 s. When there are only a few ECSs, matching UEs and ECSs has minimal impact on system performance, and joint optimization brings only marginal improvement. However, as the number of ECSs increases, the computational offloading strategy plays a bigger role, and the benefits of employing PBJO become fully apparent. Further, when a large number of ECSs are present, the effect of computation offloading becomes minimal, and the gap between algorithms decreases accordingly.

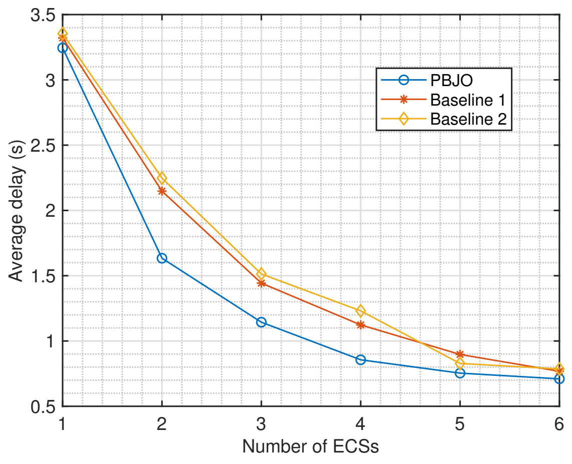

Figure 5. Average delay vs. number of ECSs.

Additionally, we plotted the trend of average delay based on the number of UEs in Figure 6. Clearly, the average delay constantly rises with an increase in UEs. It is undeniable that a greater quantity of UEs equals a heavier load on the network, causing a higher average delay. Furthermore, we observe an interesting phenomenon where the average delay produced by the PBJO algorithm experiences a significant increase whenever the number of UEs surpasses a multiple of three. For instance, the average delay increases from 1.15 s to 1.55 s when the number of UEs rises from 9 to 10, which is 5 times greater than when the number of UEs increases from 10 to 11. This is because there are three ECSs in the simulation environment. As a result, an additional UE means that one ECS will have to manage four UEs, thus leading to a rise in the average network delay. Furthermore, the figure illustrates that PBJO exhibits a decrease of at least 0.15 s in comparison to the two baseline algorithms, thereby confirming the algorithm's efficacy.

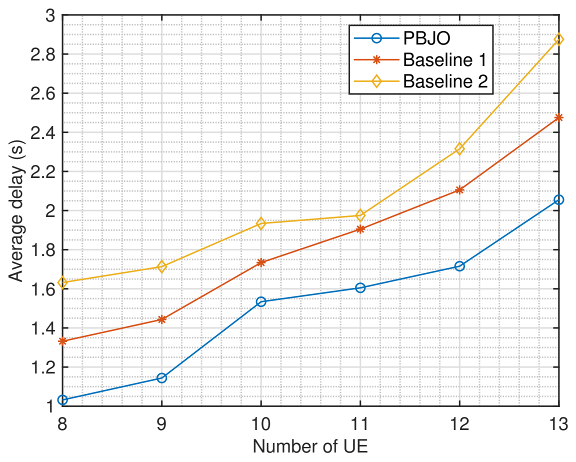

Figure 6. Average delay vs. number of UEs.

6. Conclusions

In this work, we designed a UAV-assisted edge computing network application scenario for minimizing the average delay. By analyzing this scenario, we established the optimization problem for joint UE–ECS matching and deployment of three UAV hovering positions. However, the joint optimization problem involves mixed integer nonlinear programming, which poses a great challenge to the solution. To address the joint optimization, we established a PPO-based joint optimization algorithm based on the PPO framework and implemented it for problem solving. Simulation analysis verified the convergence and outstanding performance of the proposed scheme. This study concentrated on edge computing networks deployed by a single UAV. Expanded deployment of multiple UAVs will enhance network flexibility, making it a valuable research element. Furthermore, future research will investigate UAV-assisted edge computing networks that consider multidimensional characteristics, such as energy consumption and delay, in a comprehensive manner.

Author Contributions: Z.L.: Theoretical analysis, numerical simulation, writing—original draft preparation. Q.Z.: System model, theory, review, and proofreading. Y.S.: System model, editing, and proofreading. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Young Elite Scientists Sponsorship Program by CAST (2022QNRC001), the CAST funding program (2022LX-312-Y-GZ-15), and Open Fund of Key Laboratory of Flight Techniques and Flight Safety, CAAC (No. FZ2022KF16).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: Author Qiwei Zhang was employed by the company Space Star Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Fu, S.; Wang, Y.; Feng, X.; Di, B.; Li, C. Reconfigurable Intelligent Surface Assisted Non-Orthogonal Multiple Access Network Based on Machine Learning Approaches. *IEEE Netw.* **2023**, 1–8. [CrossRef]
2. Zhang, R.; Feng, Y.; Yang, Y.; Li, X. Task Offloading with Data-Dependent Constraints in Satellite Edge Computing Networks: A Multi-Objective Approach. *Aerospace* **2023**, *10*, 804. [CrossRef]
3. Sun, M.; Bao, T.; Xie, D.; Lv, H.; Si, G. Edge Collaborative Online Task Offloading Method Based on Reinforcement Learning. *Electronics* **2023**, *12*, 3741. [CrossRef]

4. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
5. Garcia, M.H.C.; Molina-Galan, A.; Boban, M.; Gozalvez, J.; Coll-Perales, B.; Şahin, T.; Kousaridas, A. A Tutorial on 5G NR V2X Communications. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 1972–2026. [CrossRef]
6. Fu, S.; Feng, X.; Sultana, A.; Zhao, L. Joint Power Allocation and 3D Deployment for UAV-BSs: A Game Theory Based Deep Reinforcement Learning Approach. *IEEE Trans. Wirel. Commun.* **2023**, 1. [CrossRef]
7. Fu, S.; Guo, X.; Fang, F.; Ding, Z.; Zhang, N.; Wang, N. Towards Energy-Efficient Data Collection by Unmanned Aerial Vehicle Base Station With NOMA for Emergency Communications in IoT. *IEEE Trans. Veh. Technol.* **2023**, *72*, 1211–1223. [CrossRef]
8. Zhang, Y.; Dong, X.; Zhao, Y. Decentralized Computation Offloading over Wireless-Powered Mobile-Edge Computing Networks. In Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS), Dalian, China, 20–22 March 2020; pp. 137–140. [CrossRef]
9. Zhu, M.; Hou, Y.; Tao, X.; Sui, T.; Gao, L. Joint Optimal Allocation of Wireless Resource and MEC Computation Capability in Vehicular Network. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Seoul, Republic of Korea, 6–9 April 2020; pp. 1–6. [CrossRef]
10. Zhou, P.; Yang, B.; Chen, C. Joint Computation Offloading and Resource Allocation for NOMA-Enabled Industrial Internet of Things. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020; pp. 5241–5246. [CrossRef]
11. Zhou, F.; Hu, R.Q. Computation Efficiency Maximization in Wireless-Powered Mobile Edge Computing Networks. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 3170–3184. [CrossRef]
12. Mao, Y.; Zhang, J.; Song, S.H.; Letaief, K.B. Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 5994–6009. [CrossRef]
13. Nugroho, A.K.; Shioda, S.; Kim, T. Optimal Resource Provisioning and Task Offloading for Network-Aware and Federated Edge Computing. *Sensors* **2023**, *23*, 9200. [CrossRef]
14. Liu, X.; Du, X.; Zhang, S.; Han, D. Cooperative Computing Offloading Scheme via Artificial Neural Networks for Underwater Sensor Networks. *Appl. Sci.* **2023**, *13*, 1886. [CrossRef]
15. Liu, Z.; Jia, Z.; Pang, X. DRL-Based Hybrid Task Offloading and Resource Allocation in Vehicular Networks. *Electronics* **2023**, *12*, 4392. [CrossRef]
16. Shi, W.; Chen, L.; Zhu, X. Task Offloading Decision-Making Algorithm for Vehicular Edge Computing: A Deep-Reinforcement-Learning-Based Approach. *Sensors* **2023**, *23*, 7595. [CrossRef]
17. Dinh, T.Q.; Tang, J.; La, Q.D.; Quek, T.Q.S. Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling. *IEEE Trans. Commun.* **2017**, *65*, 3571–3584. [CrossRef]
18. Lim, D.; Joe, I. A DRL-Based Task Offloading Scheme for Server Decision-Making in Multi-Access Edge Computing. *Electronics* **2023**, *12*, 3882. [CrossRef]
19. Song, Z.; Liu, Y.; Sun, X. Joint Task Offloading and Resource Allocation for NOMA-Enabled Multi-Access Mobile Edge Computing. *IEEE Trans. Commun.* **2021**, *69*, 1548–1564. [CrossRef]
20. Wu, Y.; Ni, K.; Zhang, C.; Qian, L.P.; Tsang, D.H.K. NOMA-Assisted Multi-Access Mobile Edge Computing: A Joint Optimization of Computation Offloading and Time Allocation. *IEEE Trans. Veh. Technol.* **2018**, *67*, 12244–12258. [CrossRef]
21. Jeong, S.; Simeone, O.; Kang, J. Mobile Edge Computing via a UAV-Mounted Cloudlet: Optimization of Bit Allocation and Path Planning. *IEEE Trans. Veh. Technol.* **2018**, *67*, 2049–2063. [CrossRef]
22. Zhou, F.; Wu, Y.; Sun, H.; Chu, Z. UAV-Enabled Mobile Edge Computing: Offloading Optimization and Trajectory Design. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [CrossRef]
23. Hu, X.; Wong, K.K.; Yang, K.; Zheng, Z. Task and Bandwidth Allocation for UAV-Assisted Mobile Edge Computing with Trajectory Design. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; pp. 1–6. [CrossRef]
24. Zhang, T.; Xu, Y.; Loo, J.; Yang, D.; Xiao, L. Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT. *IEEE Trans. Ind. Inform.* **2020**, *16*, 5505–5516. [CrossRef]
25. Ma, T.; Yang, Y.; Xu, H.; Song, T. Optimizing Task Completion Time in Disaster-Affected Regions with the WMDDPG-GSA Algorithm for UAV-Assisted MEC Systems. *Processes* **2023**, *11*, 3000. [CrossRef]
26. Liang, W.; Ma, S.; Yang, S.; Zhang, B.; Gao, A. Hierarchical Matching Algorithm for Relay Selection in MEC-Aided Ultra-Dense UAV Networks. *Drones* **2023**, *7*, 579. [CrossRef]
27. Han, Z.; Zhou, T.; Xu, T.; Hu, H. Joint User Association and Deployment Optimization for Energy-Efficient Heterogeneous UAV-Enabled MEC Networks. *Entropy* **2023**, *25*, 1304. [CrossRef]
28. Min, M.; Xiao, L.; Chen, Y.; Cheng, P.; Wu, D.; Zhuang, W. Learning-Based Computation Offloading for IoT Devices With Energy Harvesting. *IEEE Trans. Veh. Technol.* **2019**, *68*, 1930–1941. [CrossRef]

29. Zhang, G.; Ni, S.; Zhao, P. Learning-Based Joint Optimization of Energy Delay and Privacy in Multiple-User Edge-Cloud Collaboration MEC Systems. *IEEE Internet Things J.* **2022**, *9*, 1491–1502. [CrossRef]
30. Li, J.; Gao, H.; Lv, T.; Lu, Y. Deep reinforcement learning based computation offloading and resource allocation for MEC. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018; pp. 1–6. [CrossRef]