symmetry

MDPI

Article

A DRL-Driven Intelligent Optimization Strategy for Resource Allocation in Cloud-Edge-End Cooperation Environments

Chao Fang [1,2,*], Tianyi Zhang [3], Jingjing Huang [4,*], Hang Xu [1], Zhaoming Hu [1], Yihui Yang [1], Zhuwei Wang [1], Zequan Zhou [3] and Xiling Luo [3,*]

[1] Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2] Purple Mountain Laboratories, Nanjing 211111, China
[3] School of Electronic and Information Engineering, Beihang University, Beijing 100191, China
[4] Beijing Saixi Technology Development Company with Limited Liability, Beijing 100007, China
* Correspondence: fangchao@bjut.edu.cn (C.F.); hjj187264726@126.com (J.H.); luoxiling@buaa.edu.cn (X.L.)

Abstract: Complex dynamic services and heterogeneous network environments make the asymmetrical control a curial issue to handle on the Internet. With the advent of the Internet of Things (IoT) and the fifth generation (5G), the emerging network applications lead to the explosive growth of mobile traffic while bringing forward more challenging service requirements to future radio access networks. Therefore, how to effectively allocate limited heterogeneous network resources to improve content delivery for massive application services to ensure network quality of service (QoS) becomes particularly urgent in heterogeneous network environments. To cope with the explosive mobile traffic caused by emerging Internet services, this paper designs an intelligent optimization strategy based on deep reinforcement learning (DRL) for resource allocation in heterogeneous cloud-edge-end collaboration environments. Meanwhile, the asymmetrical control problem caused by complex dynamic services and heterogeneous network environments is discussed and overcome by distributed cooperation among cloud-edge-end nodes in the system. Specifically, the multi-layer heterogeneous resource allocation problem is formulated as a maximal traffic offloading model, where content caching and request aggregation mechanisms are utilized. A novel DRL policy is proposed to improve content distribution by making cache replacement and task scheduling for arriving content requests in accordance with the information about users' history requests, in-network cache capacity, available link bandwidth and topology structure. The performance of our proposed solution and its similar counterparts are analyzed in different network conditions.

Keywords: resource allocation; cloud-edge-end cooperation networks; deep reinforcement learning; in-network caching; request aggregation

Citation: Fang, C.; Zhang, T.; Huang, J.; Xu, H.; Hu, Z.; Yang, Y.; Wang, Z.; Zhou, Z.; Luo, X. A DRL-Driven Intelligent Optimization Strategy for Resource Allocation in Cloud-Edge-End Cooperation Environments. *Symmetry* **2022**, *14*, 2120. https://doi.org/10.3390/sym14102120

Academic Editor: Mihai Postolache

Received: 16 September 2022
Accepted: 3 October 2022
Published: 12 October 2022

1. Introduction

The emerging network applications (e.g., HD streaming media transmission, multiplayer online cloud games) have led to the explosive growth of mobile traffic while bringing forward more challenging service requirements to future radio access networks, such as low-latency content delivery, efficient task offloading, massive user access, asymmetrical control and so on [1,2]. It is difficult for mobile cloud computing (MCC) to tackle these challenges due to the centralized service paradigm [3]. Therefore, how to effectively allocate limited heterogeneous network resources to improve content delivery for massive application services to ensure network quality of service (QoS) becomes particularly urgent in next-generation wireless networks [4,5].

With the advent of the Internet of Things (IoT) and the fifth generation (5G), mobile edge computing (MEC) has attracted great attention as a rising paradigm by integrating computational resources and caching capacities of the mobile edge networks [6]. MEC provides an ultra-low latency and high bandwidth network service environment to meet

the significant service requirements by pushing network functions towards access networks [7,8]. Li et al. [9] designed a collaborative multi-tier framework in mobile edge networks to offload duplicated traffic from mobile users. Moreover, Li et al. [10] proposed a hierarchical edge-end cooperation strategy to reduce the central network traffic pressure and transmission delay. Wang et al. [11] presented a scalable edge computing architecture in heterogeneous vehicular networks to promote content distribution. To meet the rich flexibility of different mobile users' demands, Chen et al. [12] proposed a comprehensive framework consisting of a resource-efficient computation offloading mechanism for the users and a joint communication and computation resource allocation mechanism for the network operator. In [13], the resource allocation and computation offloading of a cloud-MEC collaborative were studied and a game-theoretic collaborative computation offloading scheme was proposed to minimize users' energy consumption while satisfying the users' computation execution time constraint. The application of artificial intelligence (AI) to MEC can further improve network performance due to the enhanced data processing capabilities, especially in time-varying and complicated environments [14–16]. Li et al. [17] presented a heterogeneous MEC network based on reinforcement learning (RL) to optimize resource allocation in the wireless system. Khoramnejad et al. [18] discussed a DRL-based joint traffic offloading and resource management scheme to promote content delivery in MEC-assisted networks. Xu et al. [19] integrated collaborative caching and DRL to build an intelligent edge caching framework to reduce redundant content and transmission delay. Wu et al. [20] presented a computation offload scheme for dynamic resource allocation based on DRL to optimize computing performance and energy consumption in MEC systems.

Given that the limited computing and resource capacity of edge servers make MEC difficult to efficiently improve content delivery and support the multiple network requirements, in this article, we present a DRL-driven intelligent optimization scheme for resource allocation, where content caching and request aggregation mechanisms are considered in cloud-edge-end cooperation networks. Meanwhile, the asymmetrical control problem caused by complex dynamic services and heterogeneous network environments is discussed and overcome by distributed cooperation among cloud-edge-end nodes in the system.

The major contributions of this article are summarized as follows.

- We formulate the optimal resource allocation problem as a maximal traffic offloading model in heterogeneous cloud-edge-end cooperation environments, where content caching and request aggregation mechanisms are utilized to ameliorate the situation of network content redundant transmission.
- We propose a novel DRL policy to improve content distribution by making cache replacement and task scheduling rely on the information about users' history requests, in-network cache capacity, available link bandwidth and topology structure.
- We evaluate the performances of the proposed solution compared with conventional and baseline solutions in different network environments. The simulation results prove the effectiveness of the proposed mechanism and strategy.

The remainder of this article is organized as follows. We describe the system model and formulate the optimization problem in Section 2. Section 3 discusses a new DRL-driven cache replacement and task scheduling scheme. Finally, performance evaluations are conducted and analyzed in Section 4, and conclusions are given in Section 5.

2. System Model

In this section, we introduce the system models of cloud-edge-end cooperation environments and content popularity, and design a maximal traffic offloading model to describe resource allocation. The notations for key model parameters for this paper are summarized in Table 1.

Table 1. Notations for key model parameters.

Symbols	Notations
V_{im}, $\mathcal{V}_{im}$	Number and set of MCs accessed to the ith BS
V_i, $\mathcal{V}_i$	Number and set of adjacent BSs of BS i
B, $\mathcal{B}$	Number and set of BSs in the system
F, $\mathcal{F}$	Number and set of different network contents
C	Maximal cache size of the MC or BS
Q_i	Maximal queue capacity of node i
l_{ij}	Network link from node i to node j
L_{ij}	Maximal bandwidth about link l_{ij}
s^k	File size of content k
X_i^k	Boolean variable indicating whether content k is cached at node i
P_i^k	Boolean variable indicating whether content k is in the queue of node i
$Y_{im,in}$	Boolean variable indicating whether there is an indirect link between MC m and MC n accessed to BS i
$Z_{i,j}$	Boolean variable indicating whether there is a direct link between BS i and BS j
λ^k	Request arrival rate about content k

2.1. Network Model

A multi-layer cloud-edge-end cooperation network includes mobile clients (MCs), base stations (BSs) and content providers (CPs) as shown in Figure 1. In this architecture, all the contents that MCs are interested in are stored in CPs, and MCs and BSs have limited caching capacity. We try to explore the potential of this network and present the basic properties of various nodes by abstracting system topology as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$. Notably, the sets of network nodes and links are denoted by $\mathcal{V}$ and $\mathcal{L}$, respectively.

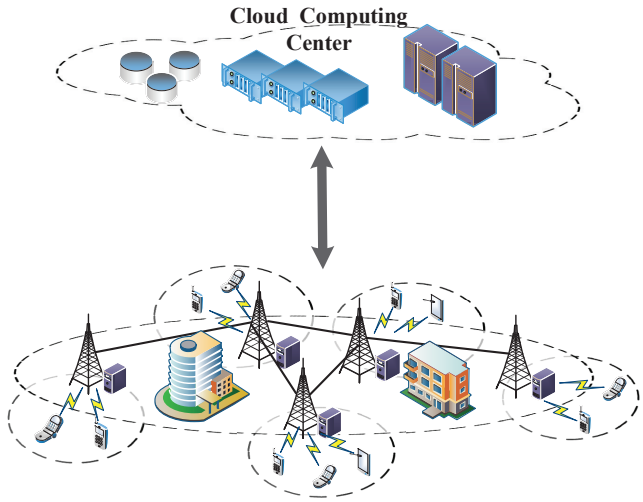

Figure 1. Network topology model of of multi-layer cloud-edge-end environments.

In the hierarchical network model, the ith BS is followed by V_{im} MCs. The maximal cache size and request queue length of the ith BS are denoted by C_i and Q_i, respectively. Similarly, the maximal storage capacity and request queue length of the mth MC at the ith BS are represented by C_{im} and Q_{im}. We assume that each link is two-directional connected, where the maximal fronthaul and backhaul bandwidths from node i to node j are denoted by L_{ij} and L_{ji}, respectively, and the network link between node i and node j is expressed by l_{ij}. $Y_{im,in}$ is a boolean variable indicating whether there is an indirect network link between MC m and n accessed to BS i. $Y_{im,in} = 1$ means that the indirect network link between MC m and n accessed to BS i exists, and 0 otherwise. $Z_{i,j}$ is a boolean variable indicating whether there is a direct network link between BS i and BS j. $Z_{i,j} = 1$ means that the direct link l_{ij} exists, and 0 otherwise.

2.2. Content Popularity Model

In our hypothesis, there are F different kinds of network contents in the cloud-edge-end environments. The required cache space of the kth content in the file set $\mathcal{F}$ is denoted by s^k. X_i^k is a boolean variable indicating whether content k is cached at node i. $X_i^k = 1$ means that node i caches content k, and 0 otherwise. P_i^k is a boolean variable indicating whether content k is in the queue of node i. $P_i^k = 1$ means that the queuing system of node i has content k, and 0 otherwise. We use the Zipf distribution model to describe content popularity and assume the popular content in descending order from 1 to F [21]. The request probability of the kth content, represented by p^k, can be shown as

$$p^k = \frac{r_k^{-\alpha}}{\sum_{i=1}^{F}(r_i^{-\alpha})}, \ \forall k \in \mathcal{F}, \tag{1}$$

where the rank of content k is denoted by r_k and skewness factor is expressed by α. A larger value of α means that more popular contents are requested by MCs in the system.

2.3. Problem Formulation

In this part, we analyze the optimal resource allocation problem by maximizing offloaded traffic in cloud-edge-end cooperation network environments. However, as shown in [12,22,23], the optimal resource allocation problem with large-scale concurrent requests based on cloud-edge-end cooperation network environments is NP-hard even without considering the mobile behaviors of network users. From the perspective of engineering, we delve into the request processing of a single request to maximize its traffic offloading with real-time network resource constraints, thus transforming the optimal overall network resources allocation into maximizing offloaded traffic of each request [24]. Meanwhile, the formulated problem is a Mixed-Integer Non-linear Programming (MINLP) problem, which is NP-hard. To tackle this problem, we decompose it into simpler subproblems for reaching optimal in-network cache status and determining a suitable scheduling algorithm, respectively [25]. Then, the offloaded traffic is modeled under different request processing ways in the following discussions.

For a request task sent from the mth MC of BS i to obtain kth content, the caused local offloaded traffic is written as

$$f_{im}^k = (X_{im}^k + P_{im}^k - X_{im}^k P_{im}^k)\lambda_{im}^k s^k, \tag{2}$$

where λ_{im}^k is the arrival rate about content request k of the mth MC at BS i.

If the request task for kth content is not satisfied by the mth MC of BS i, the offloaded traffic at the ith BS is expressed as

$$f_i^k = (X_i^k + P_i^k - X_i^k P_i^k)\lambda_{im}^k s^k. \tag{3}$$

If the request task for kth content is not satisfied at the ith BS, it will be routed to the nth indirect MC of the ith BS. The offloaded traffic at the nth MC of BS i is written as

$$f_{in}^k = (X_{in}^k + P_{in}^k - X_{in}^k P_{in}^k)Y_{im,in}\lambda_{im}^k s^k, \tag{4}$$

where $\mathcal{V}_{im}$ is the set of MCs at the ith BS.

If the request task for kth content is not satisfied at BS i and its accessed MCs, it will be routed to the neighboring BSs. The offloaded traffic of the neighboring BSs about BS i to process the content request k is written as

$$f_j^k = (X_j^k + P_j^k - X_j^k P_j^k)Z_{i,j}\lambda_{im}^k s^k, \tag{5}$$

where $\mathcal{V}_i$ is the set of the neighboring BSs about BS i.

Generally speaking, for a MC, if the required content is cached at the local terminal or the same request task is aggregated in the MC's queue, the request will be processed by the

local MC. Otherwise, the request task will be routed to the local direct BS, local indirect MCs, neighboring BSs and cloud servers on the basis of static cooperative routing (SCR) as shown in Algorithm 1 [9].

Therefore, the offloaded traffic caused by the request task for kth content sent from the mth MC of BS i, denoted by $f_{i,m}^k$, can be written as in Equation (6) (on the top of the next page). Based on Equation (6), the maximal offloaded traffic in heterogeneous cloud-edge-end cooperation environments can be formulated as Equation (7), where optimal caching status and routing decisions are made to promote content delivery.

$$f_{i,m}^k = f_{im}^k + \left(1 - \frac{f_{im}^k}{\lambda_{im}^k s^k}\right)\left\{f_i^k + \left(1 - \frac{f_i^k}{\lambda_{im}^k s^k}\right)\left[\sum_{n \in \mathcal{V}_{im} \setminus m} f_{in}^k + \prod_{n \in \mathcal{V}_{im} \setminus m}\left(1 - \frac{f_{in}^k}{\lambda_{im}^k s^k}\right)\sum_{j \in \mathcal{V}_i} f_j^k\right]\right\} \tag{6}$$

$$\begin{aligned}
&\max_{X^k, P^k} \sum_{i \in \mathcal{B}} \sum_{m \in \mathcal{V}_{im}} \sum_{k \in \mathcal{F}} f_{i,m}^k \\
s.t.\ &C_1 : \sum_{k \in \mathcal{F}} X_{im}^k s^k \le C_{im}, \forall m \in \mathcal{V}_{im}, i \in \mathcal{B} \\
&C_2 : \sum_{k \in \mathcal{F}} X_i^k s^k \le C_i, \forall i \in \mathcal{B} \\
&C_3 : \sum_{k \in \mathcal{F}} P_{im}^k \le Q_{im}, \forall m \in \mathcal{V}_{im}, i \in \mathcal{B} \\
&C_4 : \sum_{k \in \mathcal{F}} P_i^k \le Q_i, \forall i \in \mathcal{B} \\
&C_5 : \sum_{k \in \mathcal{F}} X_i^k \lambda_{im}^k s^k \le L_{im}, \forall m \in \mathcal{V}_{im}, i \in \mathcal{B} \\
&C_6 : \sum_{k \in \mathcal{F}} X_{in}^k Y_{im,in} \lambda_{im}^k s^k \le min(L_{ni}, L_{im}), \\
&\quad \forall m, n \in \mathcal{V}_{im}, i \in \mathcal{B} \\
&C_7 : \sum_{k \in \mathcal{F}} X_j^k Z_{i,j} \lambda_{im}^k s^k \le min(L_{ji}, L_{im}), \forall j \in \mathcal{V}_i, i \in \mathcal{B} \\
&C_8 : \sum_{j \in \mathcal{V}_i} X_j^k Z_{i,j} \le 1, \forall i \in \mathcal{B}, k \in \mathcal{F}
\end{aligned} \tag{7}$$

where $\mathcal{B}$ is the set of BSs in the system.

In the constraints, $C_1 - C_2$ represent that the amount of contents cached in MCs and BSs cannot exceed their maximal cache capacities C_{im} and C_i. $C_3 - C_4$ indicate that the number of content requests served by MCs and BSs cannot exceed their maximal queue lengths Q_{im} and Q_i. Moreover, C_5 means that the traffic on the link between MC m and its accessed BS i cannot exceed its maximal available bandwidth. C_6 means that the traffic on link l_{ni} and link l_{im} cannot exceed their maximal available bandwidths. C_7 indicates that the traffic on link l_{ji} and link l_{im} cannot exceed their maximal available bandwidths. C_8 means that the complementary cache mechanism is used to increase cache hit rate in the neighboring BSs.

Algorithm 1: Static Cooperative Routing Process for a Content Request

Input: Network topology $\mathcal{G}$, content request k, the network environment parameters

Output: Routing path

if *the imth MC receives the content request k* **then**
  Satisfy the request if $X_{im}^k = 1 || P_{im}^k = 1$.
else
  if *the link bandwidths and queue capacities are enough for content request k* **then**
    Route to the ith BS and satisfy it if $X_i^k = 1 || P_i^k = 1$.
    If not yet satisfied, check the following options:
    1. Route to the inth indirect MC and satisfy it if $X_{in}^k = 1 || P_{in}^k = 1$.
    2. Route to the jth neighboring BS and satisfy it if $X_j^k = 1 || P_j^k = 1$.
    3. Download the content from the cloud.
  else
    Request packet loss
  end
end
Generate the routing path.

The maximal offloaded traffic model (Equation (7)) in heterogeneous cloud-edge-end cooperation environments is an MINLP problem, which is decomposed into the subproblem of reaching optimal in-network cache status and determining a suitable scheduling algorithm. Generally, the enumeration cache placement and traversal routing algorithm can be applied to solve this problem, which leads to an optimal solution. However, this solution brings high computational complexity and computational time [13]. A DRL-driven intelligent scheme with low computation complexity is presented to analyze the maximal offloaded traffic model (Equation (7)) in the next section. Moreover, it can be used to obtain a optimal solution compared with the scheme with the common cache replacement strategies and traversal routing algorithm SCR in Section 4.

3. DRL-Based Caching Replacement and Task Scheduling

3.1. The DRL Framework

In the previous section, it was assumed that the optimal caching status and routing decisions are already made in cloud-edge-end environments to obtain the optimal resource allocation. The maximal offloaded traffic can be reached optimally by exploring all network nodes and making cache placement by way of enumeration. Nevertheless, the complexity of the exhaustive search method is high, and it is impossible to output real-time decisions in practical scenarios [26–28]. In recent years, the superiority of RL has been widely verified in obtaining the optimal action decisions based on the current situation, it becomes computationally prohibitive for decision-making in large-scale environments [29,30]. DRL, as a advanced combination of RL with deep learning (DL), can solve the curse of dimension by using the deep neural network (DNN) and automatically learn the feature representations from the raw collected data with high dimension network [31]. As a branch of DRL, Deep Q Network (DQN) is a typical model-free method that is apt for finding the optimal solution in complex unknown environments. In detail, DQN can efficiently handle the complex coupling relationships among constraint conditions, scheduling objectives and environmental parameters of cloud-edge-end network to obtain the optimal decision with an acceptable time. Motivated by this, in this section, we propose a new DQN policy to make optimal caching replacement and task scheduling on the basis of bygone users' request behavior and available resources. Moreover, the proposed DQN solution can efficiently solve the asymmetrical control problem when satisfying the complex dynamic services in the heterogeneous network environments.

As shown in Figure 2, there are two DNNs in DQN, including the evaluation network and the target network. The role of evaluation network is to output action value $Q(s_t, a_t; \omega)$ by inputting state s_t at a time slot t. To avoid the decision algorithm falling into a local optimal value, we can obtain the action by ϵ-greedy policy,

$$a_t = \begin{cases} \arg\max_{a_t} Q(s_t, a_t; \omega) & \text{with probability of } \epsilon \\ \textit{randomly select an action} & \text{otherwise} \end{cases} \tag{8}$$

where ω is the weights of the evaluation network. In our DQN policy, the back-propagation (BP) and gradient descent algorithms are adopted to update the parameters of the evaluation network. The target network, which is reset as evaluation network every K steps to promote the stability and convergence of algorithm, provides the target Q value $\max_{a_{j+1}} Q'(s_{j+1}, a_{j+1}; \omega^-)$. The loss function can be calculated as

$$L(\omega) = E\left[\left(r_j + \gamma \max_{a_{j+1}} Q'(s_{j+1}, a_{j+1}; \omega^-) - Q(s_j, a_j; \omega)\right)^2\right] \tag{9}$$

where ω^- is the weights of the target network and r_j is the reward value, which is provided by an experience replay.

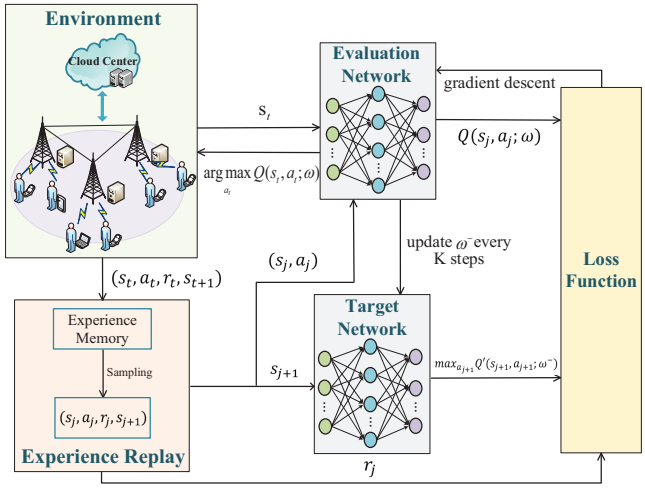

Figure 2. The framework of Deep Q Network in our system.

3.2. DQN-Based Caching Replacement and Task Scheduling

In our model, the network state space s_t at time t consists of network topology $\mathcal{G}$, the node number n_t that current request arrives at, caching status set $\boldsymbol{X_{n_t}} = (X^1_{n_t}, X^2_{n_t}, \ldots, X^F_{n_t})$, request queuing status set $\boldsymbol{P_{n_t}} = (P^1_{n_t}, P^2_{n_t}, \ldots, P^F_{n_t})$ and maximal transmission bandwidth set $L_{n_t,i}$ and L_{i,n_t}, which can be written as $s_t = \{\mathcal{G}, n_t, \boldsymbol{X_{n_t}}, \boldsymbol{P_{n_t}}, L_{n_t,i}, L_{i,n_t}, \forall n_t, i \in \mathcal{V}\}$. The purpose of the action a_t at time t is to update the caching state $\boldsymbol{X_{n_t}}$ and choose the node number of next hop n_{t+1} in the training stage, which can be denoted by $a_t = \{\boldsymbol{X_{n_t}}, n_{t+1}, \forall n_t, n_{t+1} \in \mathcal{V}\}$.

To obtain the optimal caching status and request routing, an efficient reward function is designed in our DRL model, which integrates environmental feedback signals with the optimization objective. In the process of routing, the inverse of a link bandwidth will be sent to the agent as feedback signals to promote load balance of network traffic. The agent will receive a reward signal depending on network traffic and content popularity when the request is satisfied. If the request is met at the edge of network (e.g., local MC, local BS, local indirect MCs and neighboring BSs), the agent will receive more reward to reduce content transmission latency. If the request packet is lost in the routing process, the agent

cannot obtain a reward. Therefore, the corresponding reward of the solving process is defined as

$$r_t = \begin{cases} \dfrac{1}{L_{n_t,n_{t+1}}} & \text{if request } k \text{ is not satisfied at } n_{t+1} \\ f_{i,m}^k \cdot p^k \cdot \delta & \text{if request } k \text{ is satisfied at } n_{t+1} \end{cases} \tag{10}$$

where δ is a decay factor indicating the influence of the type of network node on the current reward. Hence, the desired purpose of our DQN solution is to obtain the optimal caching status and task scheduling scheme based on current network environments through maximizing the expected accumulated reward r_t. The training process of DQN-based caching replacement and task scheduling is given in Algorithm 2.

Algorithm 2: Training process of DQN-Based Caching Replacement and Task Scheduling

Initialize experience replay memory D.
Initialize the evaluation network Q with weights ω.
Initialize the target network Q' with weights ω^-.
for *episode*=1 *to* n_e **do**
  Reset the cloud-edge-end cooperation environments.
  for *request*=1 *to* n_r **do**
    Receive a request task and observe state s_t.
    With probability ϵ select a random action a_t; otherwise, select $a_t = \arg\max_{a_t} Q(s_t, a_t; \omega)$.
    Execute a_t in emulator and observe immediate reward r_t and next state s_{t+1}.
    Package (s_t, a_t, r_t, s_{t+1}) and store it into D.
    Select samples (s_j, a_j, r_j, s_{j+1}) from D randomly and feed into the DNNs.
    Then, the weights of evaluation network are optimized by using BP and gradient descent algorithms with respect to the network parameter ω to minimize the loss $E\left[\left(r_j + \gamma \max_{a_{j+1}} Q'(s_{j+1}, a_{j+1}; \omega^-) - Q(s_j, a_j; \omega)\right)^2\right]$.
    Reset the network weights ω^- of Q' after every k steps via $\omega^- \leftarrow \omega$.
  end
  Make caching replacement on the basis of bygone users' request behavior.
end

3.3. Complexity Analysis

In this part, the complexity of DQN-based caching replacement and task scheduling is analyzed. According to the process of Algorithm 2, we assume that the operation time of initializing experience replay and DNNs is denoted by t_0 and the operation time of resetting network environments is represented by t_1. Moreover, the program operation time of task scheduling and caching replacement are expressed by t_2 and t_3, respectively. Thus, the execution time of DQN algorithm can be written as

$$\begin{aligned} T_{DQN} &= t_0 + (t_1 + t_2 \times n_r + t_3) \times n_e \\ &= t_0 + (t_1 + t_3) \times n_e + t_2 \times n_r \times n_e \\ &= (t_1 + t_3) \times n_e + t_2 \times n_r \times n_e \\ &= t_2 \times n_r \times n_e \\ &= n_r \times n_e \end{aligned} \tag{11}$$

where n_r is the number of user requests in each episode and n_e is the number of episodes. As shown in Equation (11), the values of n_r and n_e have a great influence on the execution time. We can ignore the influence of the constant term and the coefficient when n_r and n_e are very large; thus, the time complexity of this algorithm is calculated as $O(n_r n_e)$.

4. Simulation and Results

In this section, we evaluate the performance of the proposed DRL-based intelligent optimization strategy in cloud-edge-end networks and discuss the obtained results with different strategies.

4.1. Simulation Setting

To prove the effectiveness of the proposed strategy, we compare the performance of three conventional existing solutions and two baseline approaches, referred to as "LRFU+SCR", "LFU+SCR", "LRU+SCR" , "Popularity+SCR" and "NoCache+SCR" in different network environments. In "NoCache+SCR", the cache is not deployed at MCs or BSs, and all the contents are fetched from cloud servers. In "Popularity+SCR", the multi-layer cooperative caching is initially set based on the known content popularity distribution, which can improve the caching and routing efficiency [32]. "LRFU+SCR", "LFU+SCR" and "LRU+SCR" utilize the most widely used three online caching replacements to promote traffic offloading. In "LRU+SCR" and "LFU+SCR", MCs and BSs make cache replacement with the Least Recently Used (LRU) and the Least Frequently Used (LFU) replacement strategies. In "LRFU+SCR", MCs and BSs adjust their caching states with Least Recently Frequently Used (LRFU) replacement strategy, which integrates the advantages of LRU and LFU to improve cache hit rate [33]. The task scheduling algorithm of above comparison schemes are made by SCR. In our proposed scheme, the caching replacement and scheduling decisions are driven by DQN policy, which can realize optimal resource allocation in accordance with bygone users' request behavior and current environments. Meanwhile, the request aggregations are considered to solve the problem of redundant transmission in the simulation.

We use MATLAB and Python to implement a simulator that constructs the hierarchical cloud-edge-end network, in which the cloud is connected to six BSs and each BS is connected to two MCs, as modeled in Section 2. Based on [21,24,34], we assume that the skewness factor of Zipf distribution varies from 0.4 to 1.6 and cache capability of network node is defined as a percentage from 0.1% to 1%, which represents the proportion of node cache size and different network content number. All codes of conventional and baseline solutions are written in MATLAB(9.9.0). To implement the DQN algorithm, two DNNs were configured with the same settings, where each of them consisted of four fully connected layers, two of which were hidden layers with 32 neurons, respectively. The activation function we adopted for all hidden layers was the sigmoid for which the range was $(0, 1)$. The RMSprop optimizer method was used to learn the DNN weight with the given learning rate. Besides, the coefficient of ϵ-*greedy* is set as 0.7, learning rate is chosen as 0.1 and discount factor is defined as 0.9. We design that the reply memory size is 10,000 and batch size is 32. The update interval of target network is set as 300 to promote the algorithm convergence speed. All simulation results of our proposed scheme were obtained with Tensorflow 2.0 on the Python 3.8 platform on a personal computer (AMD Ryzen 9 5900HX with Radeon Graphics @3.3 GHz, 32 GB RAM).

4.2. Result Discussion

As shown in Figure 3, when caching capability increases, network nodes are able to store more network contents, which enhances the ability of content delivery. In this process, the DQN-based strategy shows superior performance and the performance gap between the proposed solution and others is enlarged simultaneously. The main reason is that DQN policy can make optimal routing decisions with current network environments and predict future requests based on users' request behavior to update the cache. Similarly, the gap

between "Popularity+SCR" and online solutions increases due to the fact that network nodes in online solutions prefer to store some short-term rather than long-term popular network contents according to the users' request behavior. "LRFU+SCR" is close to "Popularity+SCR" and better than "LFU+SCR" and "LRU+SCR". The reason is that "LRFU+SCR" can adjust the caching contents by considering both the time and frequency characteristics of network requests, which reduces the negative impact caused by short-term request behavior and improves cache hit rate. With the increase in cache size, the performance of "NoCache+SCR" remains constant because BSs and MCs can only offload mobile traffic through the request aggregation mechanism.

As shown in Figure 4, when content popularity varies, more requests are sent by MCs to obtain popular network files in the system, which greatly increases the number of node cache hit and makes the performance of the models with in-network caching closer. For "NoCache+SCR", its performance has a significant improvement owing to request aggregation mechanism. With the increase in content popularity, the DQN-based strategy can converge fast and offload more mobile traffic than other solutions. In this process, frequent caching replacement can be gradually avoided, which bridges the performance disparity between the intelligent caching based on DQN and other online caching replacements. As content popularity grows, the advantage of intelligent caching and scheduling gets diminishing, which makes "Popularity+SCR" perform close to the proposed solution.

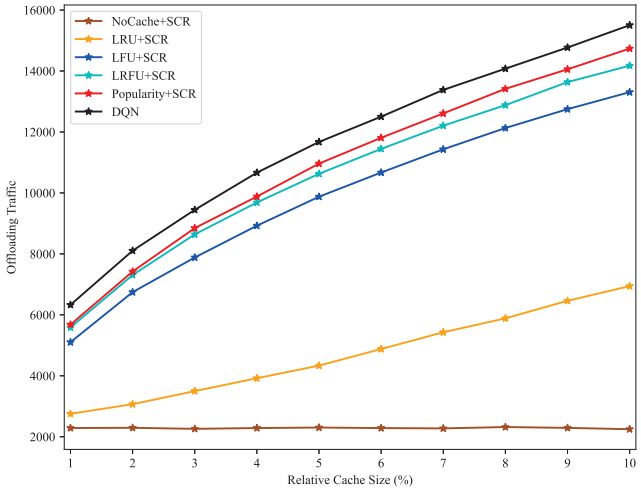

Figure 3. Offloaded network traffic versus cache size.

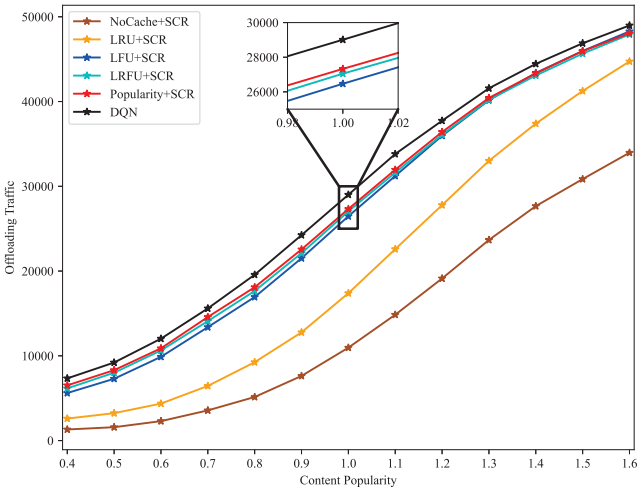

Figure 4. Offloaded network traffic versus content popularity.

As shown in Figure 5, the increasing number of different network contents indicates that more unpopular contents are requested by network users, which deteriorates cache hit rate of the solutions with limited storage capacity and efficiency of request aggregation in network nodes. When content diversity increases, the performance of in-network caching solutions is significantly degraded while that of "NoCache+SCR" is slightly affected. The reason is that most of the requests in "NoCache+SCR" can only be satisfied in the cloud instead of by request aggregation. In this process, the DQN-based strategy performs better than "Popularity+SCR" and other solutions. The reason is that DQN policy can make better cache replacement and task scheduling to improve content delivery on the basis of current cloud-edge-end environments.

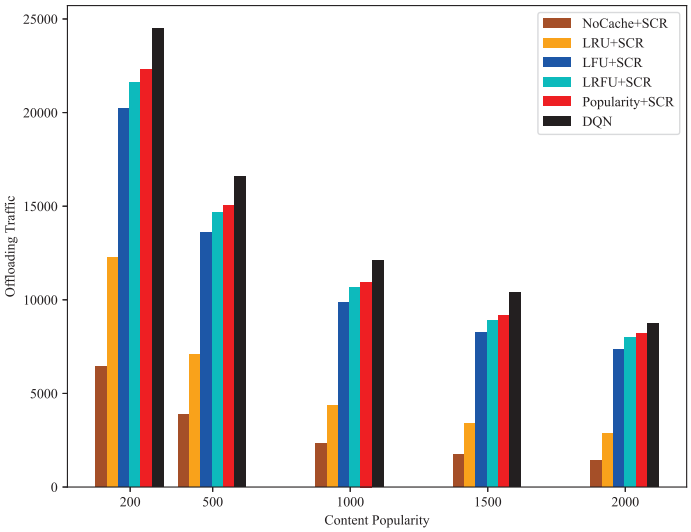

Figure 5. Offloaded network traffic versus the number of different contents.

As shown in Figure 6, a larger request arrival rate indicates that more content requests are sent in a slot, which improves popularity prediction in solutions with content caching and request aggregation mechanisms. In this process, the problem of packet loss becomes more serious due to the limited link bandwidth. When the request arrival rate increases, offloading traffic of all the solutions is promoted with a slowing growth rate. When the request arrival rate continues to grow and exceeds a certain value, the performance of in-network caching solutions barely increases anymore. The reason is that more network requests are lost and retransmitted in the routing process, which balances the influence among packet loss and retransmission, cache hit rate and request aggregation. However, the DQN-based strategy achieves the best performance among all solutions, because the probability of packet loss can be relieved by optimally allocating network resources and caching contents. The performance of "NoCache+SCR" is improved due to the growing request aggregation.

The ability of network nodes to aggregate requests is mostly influenced by queue capacity. When the request arrival rate of MCs is constant, as illustrated in Figure 7, the offloading traffic of all solutions is promoted slowly with an increase in the maximum queue length. Further, the performance of the DQN-based strategy is better than other solutions since the effect of request aggregation and the packet loss of requests can be greatly improved through intelligent task scheduling. When the queue length continues to grow and exceeds a certain value, the offloading traffic of all the solutions is barely increasing anymore, even beginning to decrease. This phenomenon shows that as queue capacity increases, network performance tends to become saturated and the ability of network nodes to aggregate requests barely improves, demonstrating that the average queue length is steady at a given request arrival rate.

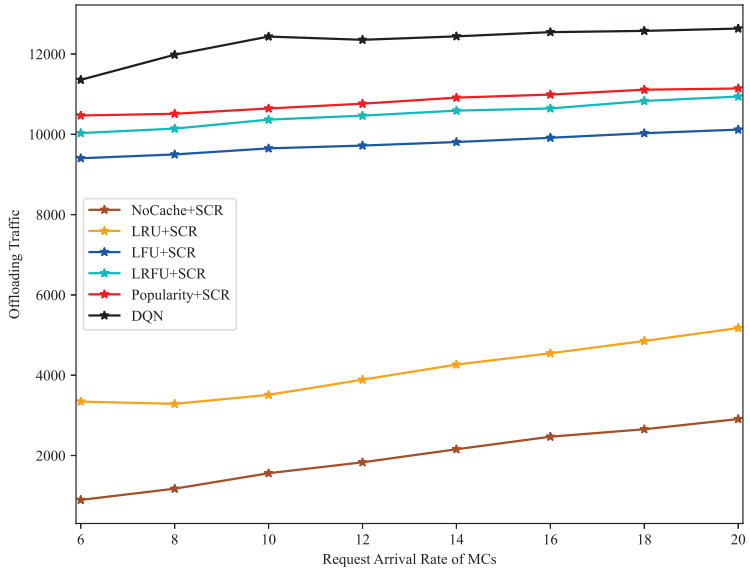

Figure 6. Offloaded network traffic versus request arrival rate.

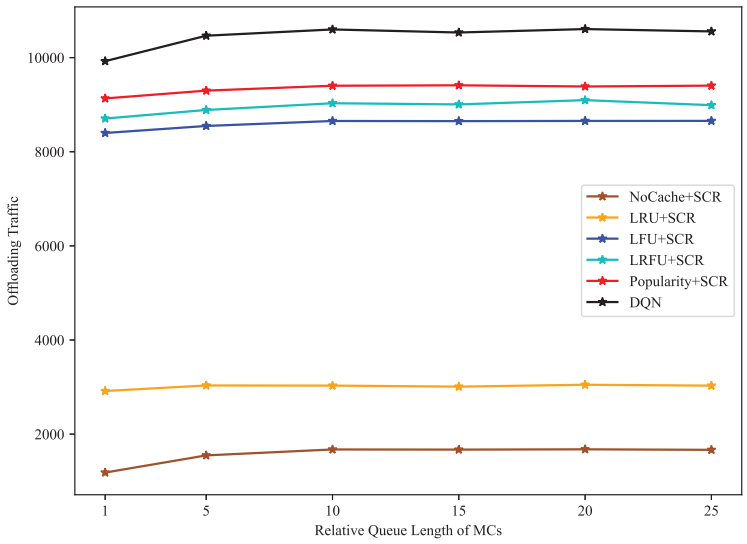

Figure 7. Offloaded network traffic versus queue length.

In the simulation, we set the learning rate as 0.1 in pursuit of a better convergence effect and the number of training episodes is 300. As shown in Figure 8, we compare the average reward of DQN policy in each episode under different caching capability of network nodes. During the training episodes, the average reward can converge stably when episode is about 40 and the success rate for meeting users' requests by cloud-edge-end cooperation is 75%, which demonstrates that DQN policy can quickly obtain the optimal decision. Moreover, the average reward for each episode under different caching capability was described by the traffic offloading. Hereby, a larger cache size indicates that more popular contents can be stored in the cloud-edge-end environments, which improves resource allocation and achieves more network reward.

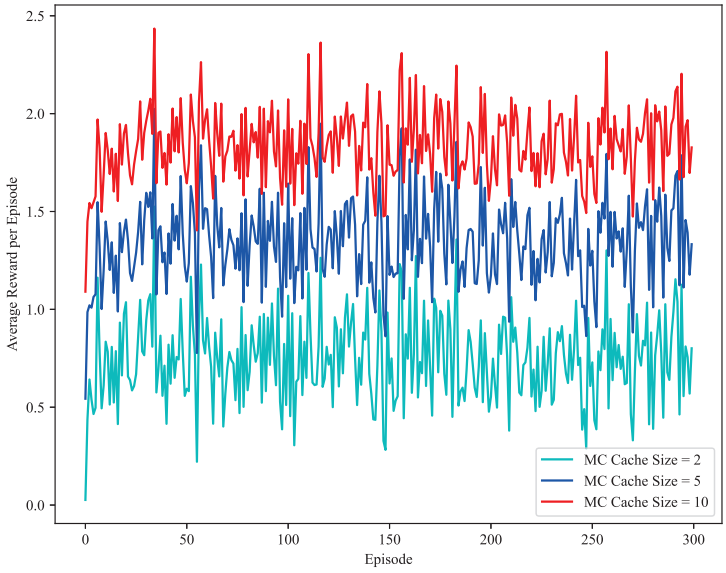

Figure 8. Reward versus cache size.

5. Conclusions

In this article, we presented a DRL-driven intelligent optimization strategy for resource allocation in heterogeneous cloud-edge-end collaboration environments, where content caching and request aggregation mechanisms are considered. We use a maximal traffic offloading model to describe the issue of multi-layer heterogeneous resource allocation. Then, a novel DRL policy was designed to improve content distribution and solve the asymmetrical control problem by making cache replacement and task scheduling for arriving content requests in accordance with the information about users' history requests, in-network cache capacity, available link bandwidth and topology structure. We evaluated the performances of the proposed solution against both conventional and baseline solutions in different network environments, The simulation results prove the effectiveness of the proposed mechanism and strategy.

In future work, the mobile behaviors of network users and spectrum resource reuse will be taken into consideration to make our proposed model more complete in practical scenarios. The related multi-objective optimization (e.g., energy efficiency and response latency) problems will be investigated and discussed to meet differentiated service requirements of network users in cloud-edge-end cooperation networks. Moreover, full-dimensional collaboration and sophisticated asymmetrical control will be investigated to verify the system performance in more complicated environments.

Author Contributions: Conceptualization, C.F. and T.Z.; methodology, C.F., T.Z. and Y.Y.; software, T.Z., H.X. and Y.Y.; validation, C.F. and T.Z.; formal analysis, C.F., T.Z., H.X. and Z.H.; investigation, C.F. and J.H.; resources, C.F. and J.H.; data curation, C.F., T.Z. and H.X.; writing—original draft preparation, C.F. and T.Z.; writing—review and editing, C.F., J.H., Z.W., Z.Z. and X.L.; supervision, C.F., J.H. and X.L.; project administration, C.F., Z.W., Z.Z. and X.L.; funding acquisition, C.F., Z.W., Z.Z. and X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Beijing Nova Program of Science and Technology grant number *Z*191100001119094, Beijing Natural Science Foundation grant number *L*202016 and Zhejiang 'JIANBING' R&D Project grant number 2020C05005.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Luo, Q.; Hu, S.; Li, C.; Li, G.; Shi, W. Resource scheduling in edge computing: A survey. *IEEE Commun. Surv. Tutor.* **2021**, *23*, 2131–2165. [CrossRef]

2. Fang, C.; Yao, H.; Wang, Z.; Wu, W.; Jin, X.; Yu, F.R. A survey of mobile information-centric networking: Research issues and challenges. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2353–2371. [CrossRef]
3. Kumar, M.; Sharma, S.C.; Goel, A.; Singh, S.P. A comprehensive survey for scheduling techniques in cloud computing. *J. Netw. Comput. Appl.* **2019**, *143*, 1–33. [CrossRef]
4. Lopez, P.G.; Montresor, A.; Epema, D.; Datta, A.; Higashino, T.; Iamnitchi, A.; Barcellos, M.; Felber, P.; Riviere, E. Edge-Centric Computing: Vision and Challenges. *ACM SIGCOMM Comput. Commun. Rev.* **2015**, *45*, 37–42. [CrossRef]
5. Fang, C.; Guo, S.; Wang, Z.; Huang, H.; Yao, H.; Liu, Y. Data-driven intelligent future network: Architecture, use cases, and challenges. *IEEE Commun. Mag.* **2019**, *57*, 34–40. [CrossRef]
6. Abbas, N.; Zhang, Y.; Taherkordi, A.; Skeie, T. Mobile edge computing: A survey. *IEEE Internet Things J.* **2017**, *5*, 450–465. [CrossRef]
7. Mouradian, C.; Naboulsi, D.; Yangui, S.; Glitho, R.H.; Morrow, M.J.; Polakos, P.A. A comprehensive survey on fog computing: State-of-the-art and research challenges. *IEEE Commun. Surv. Tutor.* **2017**, *20*, 416–464. [CrossRef]
8. Vaquero, L.M.; Rodero-Merino, L. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 27–32. [CrossRef]
9. Li, X.; Wang, X.; Li, K.; Han, Z.; Leung, V.C. Collaborative multi-tier caching in heterogeneous networks: Modeling, analysis, and design. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 6926–6939. [CrossRef]
10. Li, X.; Wang, X.; Xiao, S.; Leung, V.C. Delay performance analysis of cooperative cell caching in future mobile networks. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 5652–5657.
11. Wang, C.; Li, Y.; Jin, D.; Chen, S. On the serviceability of mobile vehicular cloudlets in a large-scale urban environment. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2960–2970. [CrossRef]
12. Chen, X.; Li, W.; Lu, S.; Zhou, Z.; Fu, X. Efficient resource allocation for on-demand mobile-edge cloud computing. *IEEE Trans. Veh. Technol.* **2018**, *67*, 8769–8780. [CrossRef]
13. Guo, H.; Liu, J. Collaborative computation offloading for multiaccess edge computing over fiber–wireless networks. *IEEE Trans. Veh. Technol.* **2018**, *67*, 4514–4526. [CrossRef]
14. Chen, M.; Qian, Y.; Hao, Y.; Li, Y.; Song, J. Data-driven computing and caching in 5g networks: Architecture and delay analysis. *IEEE Wirel. Commun.* **2018**, *25*, 70–75. [CrossRef]
15. Wang, Y.; Li, P.; Jiao, L.; Su, Z.; Cheng, N.; Shen, X.S.; Zhang, P. A data-driven architecture for personalized qoe management in 5g wireless networks. *IEEE Wirel. Commun.* **2016**, *24*, 102–110. [CrossRef]
16. Jiang, K.; Sun, C.; Zhou, H.; Li, X.; Dong, M.; Leung, V.C. Intelligence-empowered mobile edge computing: Framework, issues, implementation, and outlook. *IEEE Netw.* **2021**, *35*, 74–82. [CrossRef]
17. 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.
18. Khoramnejad, F.; Erol-Kantarci, M. On joint offloading and resource allocation: A double deep q-network approach. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 1126–1141. [CrossRef]
19. Xu, S.; Liu, X.; Guo, S.; Qiu, X.; Meng, L. Mecc: A mobile edge collaborative caching framework empowered by deep reinforcement learning. *IEEE Netw.* **2021**, *35*, 176–183. [CrossRef]
20. Wu, G.; Zhao, Y.; Shen, Y.; Zhang, H.; Shen, S.; Yu, S. Drl-based resource allocation optimization for computation offloading in mobile edge computing. In Proceedings of the IEEE INFOCOM 2022—IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), New York, NY, USA, 2–5 May 2022; pp. 1–6.
21. Breslau, L.; Cao, P.; Fan, L.; Phillips, G.; Shenker, S. Web caching and zipf-like distributions: Evidence and implications. In Proceedings of the IEEE INFOCOM'99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, the Future Is Now (Cat. No. 99CH36320), New York, NY, USA, 21–25 March 1999; Volume 1, pp. 126–134.
22. Dai, J.; Hu, Z.; Li, B.; Liu, J.; Li, B. Collaborative hierarchical caching with dynamic request routing for massive content distribution. In Proceedings of the 2012 Proceedings IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 2444–2452.
23. Zhao, J.; Li, Q.; Gong, Y.; Zhang, K. Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 7944–7956. [CrossRef]
24. Fang, C.; Xu, H.; Yang, Y.; Hu, Z.; Tu, S.; Ota, K.; Yang, Z.; Dong, M.; Han, Z.; Yu, F.R.; et al. Deep-reinforcement-learning-based resource allocation for content distribution in fog radio access networks. *IEEE Internet Things J.* **2022**, *9*, 16874–16883. [CrossRef]
25. Sun, C.; Li, H.; Li, X.; Wen, J.; Xiong, Q.; Wang, X.; Leung, V.C. Task offloading for end-edge-cloud orchestrated computing in mobile networks. In Proceedings of the 2020 IEEE Wireless Communications and Networking Conference (WCNC), Seoul, Korea, 25–28 May 2020; pp. 1–6.
26. Hong, Z.; Chen, W.; Huang, H.; Guo, S.; Zheng, Z. Multi-hop cooperative computation offloading for industrial iot–edge–cloud computing environments. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *30*, 2759–2774. [CrossRef]
27. Wang, H.; Liu, T.; Kim, B.; Lin, C.-W.; Shiraishi, S.; Xie, J.; Han, Z. Architectural design alternatives based on cloud/edge/fog computing for connected vehicles. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 2349–2377. [CrossRef]
28. Dai, B.; Niu, J.; Ren, T.; Atiquzzaman, M. Towards mobility-aware computation offloading and resource allocation in end-edge-cloud orchestrated computing. *IEEE Internet Things J.* **2022**, *9*, 19450–19462. [CrossRef]

29. Nath, S.; Wu, J. Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems. *Intell. Converg. Netw.* **2020**, *1*, 181–198. [CrossRef]
30. Luong, N.C.; Hoang, D.T.; Gong, S.; Niyato, D.; Wang, P.; Liang, Y.-C.; Kim, D.I. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 3133–3174. [CrossRef]
31. Xu, C.; Liu, S.; Zhang, C.; Huang, Y.; Lu, Z.; Yang, L. Multi-agent reinforcement learning based distributed transmission in collaborative cloud-edge systems. *IEEE Trans. Veh. Technol.* **2021**, *70*, 1658–1672. [CrossRef]
32. Wang, X.; Chen, M.; Taleb, T.; Ksentini, A.; Leung, V. Cache in the air: exploiting content caching and delivery techniques for 5g systems. *IEEE Commun. Mag.* **2014**, *52*, 131–139. [CrossRef]
33. Lee, D.; Choi, J.; Kim, J.-H.; Noh, S.; Min, S.L.; Cho, Y.; Kim, C.S. Lrfu: A spectrum of policies that subsumes the least recently used and least frequently used policies. *IEEE Trans. Comput.* **2001**, *50*, 1352–1361.
34. Li, J.; Liu, B.; Wu, H. Energy-efficient in-network caching for content-centric networking. *IEEE Commun. Lett.* **2013**, *17*, 797–800. [CrossRef]