Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5579637, 10 pages
https://doi.org/10.1155/2021/5579637

Research Article

Video Stream Session Migration Method Using Deep Reinforcement Learning in Cloud Computing Environment

Lingling Li and Huixia Liu

Department of Cyber Security, Henan Police College, Zhengzhou, Henan 450000, China

Correspondence should be addressed to Lingling Li; lll007@hnp.edu.cn

Received 28 January 2021; Revised 11 March 2021; Accepted 26 March 2021; Published 12 April 2021

Academic Editor: Shan Zhong

Copyright © 2021 Lingling Li and Huixia Liu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the resource scheduling of streaming Media Edge Cloud (MEC), in order to balance the cost and load of migration, this paper proposes a video stream session migration method based on deep reinforcement learning in cloud computing environment. First, combined with the current popular OpenFlow technology, a novel MEC architecture is designed, which separates streaming media service processing in application layer from forwarding path optimization in network layer. Second, taking the state information of the system as the attribute feature, the session migration is calculated, and gradient reinforcement learning is combined with in-depth learning and deterministic strategy for video stream session migration to solve the user request access problem. The experimental results show that the method has a better request access effect, can effectively improve the request acceptance rate, and can reduce the migration cost, while shortening the running time.

1. Introduction

In recent years, with the maturity of cloud computing technology, streaming media services are gradually transforming to cloud form, that is, streaming Media Cloud. Streaming media cloud pushes the content requested by users to the edge of the network by placing media edge cloud in different geographical locations, so that to reduce the user response delay and reduce the traffic load of the main network [1]. At the same time, the subcloud can adapt to the changes of system load and the size of user requests, so that to effectively solve the problem of traditional streaming media services [2].

In streaming Media Edge Cloud (MEC), system resources are virtualized into resource pools to ensure service transparency. Cloud resource allocation is automatically adjusted by the cloud platform according to the scale of actual demand, so how to allocate system resources in real-time to meet user needs. Under the condition of limited resource allocation, the fluctuation and randomness of user request mode will make the system load unbalanced and affect the access effect of user request [3, 4].

In order to solve the above problems, domestic and foreign scholars have proposed a migration-based task scheduling method for streaming media. Ref. [5] proposed a session migration strategy based on dynamic threshold allocation (SMS-DTA). According to the popularity distribution, the session allocation thresholds of all kinds of videos on each server are determined, and the user request access is guided by the allocation thresholds. Ref. [6] proposed resource dispatch based on data priority (RDDP) algorithm. However, considering the impact of the urgency and scarcity of data blocks on priority, quantitative calculation is not given. Only the balance factor is used to measure the quantitative relationship between them, and the influence of time factor on emergency quantification is omitted. Ref. [7] proposed a direct access storage device (DASD) hopping algorithm to migrate sessions of nodes with different loads in order to maintain the load balance of hard disk. However, due to the lack of self-adaptability, it is difficult to adjust the strategy according to the system operation scenario. Moreover, the mathematical model is relatively complex and the calculation is large, which cannot solve the problem of large-scale resource allocation.

Ref. [8] explored how to make the streaming media edge cloud admit more requests via online session migration and proposed an adaptive strategy of online session migration. Besides the load information, the video popularity is adopted for obtaining the allocation thresholds of different videos on each server, and a new request would be admitted under the guidance of the obtained threshold distribution. Specially, when the video popularity varies, the allocation thresholds would be recalculated. Ref. [9] proposed a joint optimization algorithm of session migration and video deployment, the proposed strategy is more adaptive to dynamic fluctuation of video popularity, and thus gains a flexible balance between service cost and quality. The trace-driven experiment verified the effectiveness of the proposed method.

According to the resource allocation of streaming media edge cloud, in order to balance the cost and load of migration, considering the cost of migration, load balancing, and other constraints, this paper proposes a video stream session migration method based on deep reinforcement learning. Based on the current popular OpenFlow technology, a novel MEC architecture is designed, which separates streaming media service processing in application layer from forwarding path optimization in network layer to ensure service transparency. The main innovations are as follows:

(1) This paper improves the resource utilization by effectively utilizing the state information of the MEC system, combining in-depth learning and deterministic strategy for video stream session migration

(2) This paper proposes a session migration computing model to process user requests more scientifically, maximize the access rate of user requests, and control the migration cost appropriately, at the same time, make the system achieve load balancing as far as possible

2. Streaming Media Edge Cloud Architecture

Streaming Media Edge Cloud is located on the edge of the network, which is responsible for local video services. As shown in Figure 1, combined with the current popular OpenFlow technology, this paper designs a novel MEC architecture. The whole MEC is composed of streaming media server, business management server, and OpenFlow controller and switch, in which the streaming media server is responsible for providing media streaming to users; the business management server is mainly responsible for the access scheduling of user requests, generating migration strategies and sending them to OpenFlow controller; the OpenFlow controller and switch, on the one hand, it constitutes a media stream distribution network, on the other hand, it is responsible for the actual implementation of session migration; OpenFlow controller is responsible for generating flow tables according to migration strategy and sending them to switches; OpenFlow switch completes the modification and forwarding of data packets according to flow tables.

By introducing MEC architecture, streaming media service processing in application layer is separated from forwarding path optimization in network layer, and transparency of video service is realized.

3. Session Scheduling Strategy Based on Deep Reinforcement Learning

Assuming that the video content provided by the MEC system has I kind of video content, the i kind of video is represented by v_i. Each kind of video is encoded at a constant bit rate and serves at the same bit rate [10, 11]. Assuming that the total number of MEC streaming media servers is J, the j server is represented by M_j, and D is defined as the video deployment matrix with the size of $I \times J$, and the element $d_{ij} \in \{0, 1\}$ represents whether or not a copy of v_i is deployed on M_j. Assuming that all servers are homogeneous, and a single server can provide up to C streaming sessions at the same time, as well as up to S videos [12].

K is defined as a session distribution matrix with the size of $I \times J$. A single element $k_{ij} \in [0, 1]$ represents the ratio of all sessions of video v_i on server M_j to the total service capacity (JC) of the system. G is defined as a server adjacency matrix with the size of $J \times J$. Element $g_{nj} \in \{0, 1\}$ denotes whether there are sessions on M_n server that can be migrated to the server, where $n = 1, 2, \cdots, J$.

Define l_j as the load of streaming media server M_j, that is, the total number of access sessions, then:

$$l_j = \sum_{i=1}^{I} \left(k_{ij} \times JC\right). \tag{1}$$

Define $\bar{L}$ as the average load of all streaming media servers in the system, then:

$$\bar{L} = \sum_{j=1}^{J} l_j / N. \tag{2}$$

In this paper, the state information of the MEC system is taken as an attribute feature, and the decision-maker and value function are fitted by deep convolution neural network combined with reinforcement learning elements such as state space, action set, and return function. In order to improve the efficiency of the algorithm, the deterministic strategy gradient is used to train the neural network.

3.1. Session Scheduling Model. For streaming media edge cloud system, the goal of reinforcement learning is to access the video request to the most suitable server independently according to the current MEC system status and video request according to the experience strategy. Then, according to the load state of the server, the optimal migration method for the current incoming user requests is obtained by using the migration video strategy to perform the request access or one-step session migration action [13, 14].

In this paper, the deep reinforcement learning method is applied to session scheduling in streaming media edge cloud, and its session migration method is shown in Figure 2.

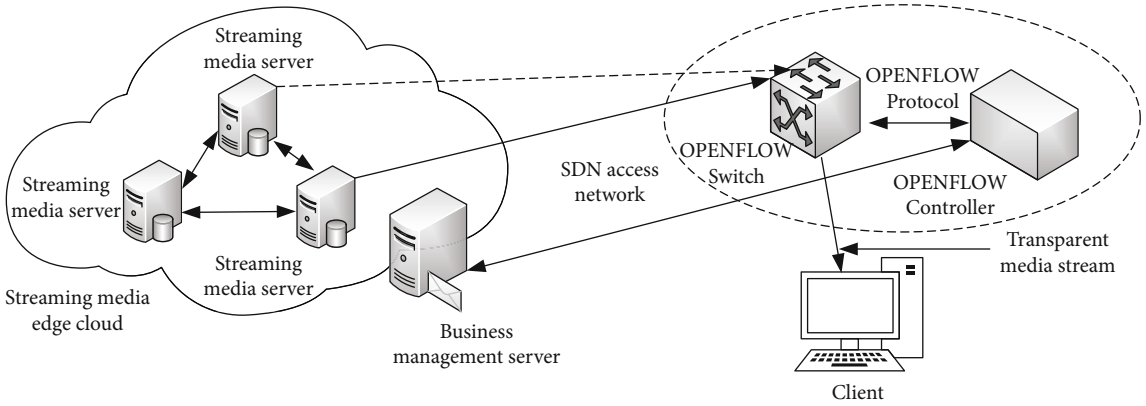

Figure 1: The novel architecture of MEC.

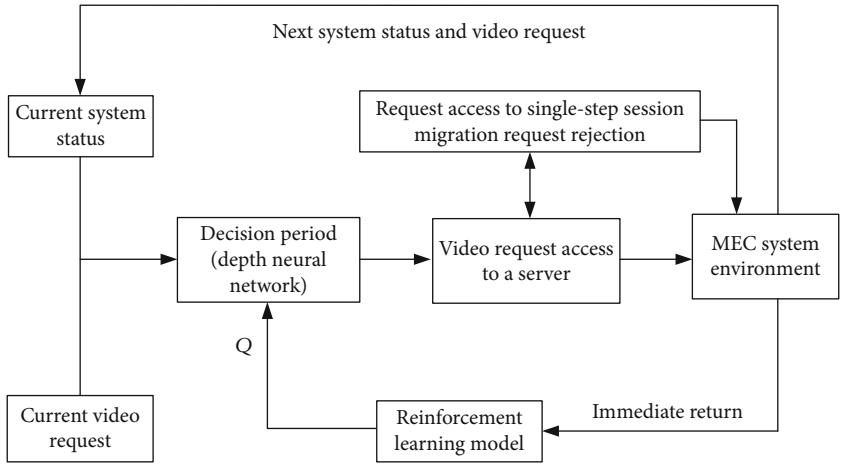

Figure 2: Conversation transfer model based on deep reinforcement learning.

In Figure 2, for the current step video request v_i, the decision-making action of the decision-maker is to connect the video request to a server, assuming that the server accessed is M_j, then the strategy of moving out video is: if M_j is not full, no video needs to be moved out, set the number of moving out video to be 0, corresponding to the request access; if M_j is full, it needs to move out the video and move out the video. The set of numbers is $\{v_m \mid k_{mj} \geq 1, m \neq i, m \leq I\}$, corresponding to one-step session migration.

3.2. Enhanced Learning Model of Conversation Transfer. According to the characteristics of the problem, the state of t time step in the MEC system is defined as follows:

$$s_t = [G_t : R_t : D_t : K_t]. \tag{3}$$

Among them, G_t is the server adjacency matrix and the size is $J \times J$, which indicates whether session migration can be carried out among servers, R_t is the video request matrix of t time step, the size is $I \times J$, the elements of one row in the video request matrix are all 1, the other elements are all 0, and the corresponding video number of the row with the elements all 1 is the appropriate one. The former video request D_t is the video deployment matrix and the size is $I \times J$, which reflects the deployment of video copies in the MEC system. K_t is the session distribution matrix and the size is $I \times J$, which will reflect the distribution of video sessions in the MEC system. Every time a new video request is processed, the system will undergo a state transition [15–17].

Since the task is to decide which server to access or reject the request based on the current MEC system status and video request, the action is defined as the server number M_j to which the video request is accessed, where $i = 1, 2, \cdots, I$. For the current step video request v_i, the optional action set is shown in Formula (4). When v_i accesses MEC directly or through session migration, the set of optional actions is the set of servers deployed with video v_i; when rejecting video request v_i, the corresponding action is 0.

$$a_t = \begin{cases} M_j | d_{ij} = 1, j \leq J, \text{insert}, \\ 0, \text{ refuse}. \end{cases} \tag{4}$$

If video request v_i is accessed to server M_j according to the decision-making action, this paper chooses the deployment of video v_i on server M_j, the load of server M_j, and the variance of load balance of MEC system after executing the action as the immediate return function. The video

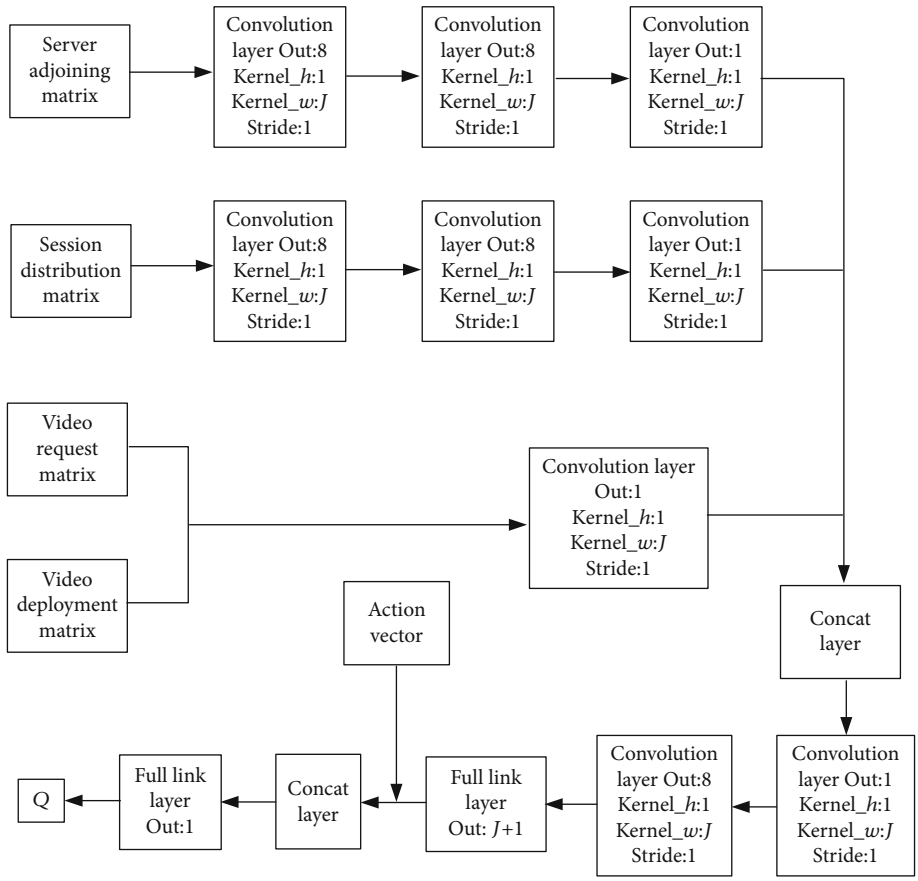

Figure 3: Q network model based on bible network.

deployment on the server, the load value of the server, and the load balancing variance of the system are different, so the load balancing variance of the system is normalized. The load balancing variance function is defined as:

$$\sigma = \arctan\left(\frac{\sum_{j=1}^{J}\left(l_j - \bar{L}\right)^2}{J}\right) \times \frac{2}{\pi}. \tag{5}$$

Since the variance of load balancing is the inverse of variance, the formula above shows that the larger the variance σ of load balancing, the more balanced the load of the system.

If the video requested in time step t is v_i, the quotation value returned by action a_t of that step is

$$r(s_t, a_t) = \begin{cases} \omega_1 \times d_{ij} + \omega_2 \times \left(1 - \dfrac{l_j}{C}\right) + \omega_3 \times \sigma, \text{insert}, \\ -1 + \omega_1 \times d_{ij} + \omega_3 \times \sigma, \text{transfer}, \\ -1, \text{ refuse}, \end{cases} \tag{6}$$

where $i = 1, 2, \cdots, I$, $j = 1, 2, \cdots, J$, when video v_i is deployed on the server M_j corresponding to decision action a_t, there will be corresponding reward value ω_1. If video v_i is not deployed on server M_j, the action is not a reasonable access action. It is not in the optional action set, reward value 0, and reward value $1 - l_j/C$ represents the remaining service capability of the server. When the server is full, that is, the residual service capacity is 0, the reward value is 0. When migration occurs, because session migration has a certain cost, the reward value is reduced by 1 as the corresponding penalty. When the action is to reject the video request, the return value is set to -1. ω_1, ω_2, and ω_3 represent the weights of the returns from the three optimization objectives, respectively. The weights can be set according to the importance of the optimization objectives, but the sum of the three weights must satisfy $\sum_{i=1}^{3}\omega_i = 1$.

Defined in MEC system state s_t, after taking action a_t, if strategy μ is continuously implemented, the expected value of immediate return is action-value function. Defined Bellman equation as follows:

$$Q^{\mu}(s_t, a_t) = E[r(s_t, a_t) + \gamma Q^{\mu}(s_{t+1}, \mu(s_{t+1}))], \tag{7}$$

where $r(s_t, a_t)$ is the immediate return value after taking action a_t under the state s_t of the MEC system. In the whole session scheduling process, the above equation is the final solution of the equation, and the optimal scheduling strategy is obtained by solving the equation.

3.3. Migration Computing Model. In MEC architecture, the migration cost can be expressed as $\sum_{i=1}^{I}\sum_{j=1}^{J}\max(k_{ij}-a_{ij},0)$ by the number of migrated sessions. In addition, this paper specifies the maximum threshold M, i.e., $\sum_{i=1}^{I}\sum_{j=1}^{J}\max(k_{ij}-a_{ij},0)\le M$, for a single migration cost, where the value of M is determined by OpenFlow's flow processing capability.

Under the unbalanced load distribution, the full-loaded servers can continue to access new requests only if some sessions are moved out. Therefore, whether the load is balanced or not will indirectly affect the cost of migration. In practice, due to the fluctuation of request distribution, all kinds of video requests do not arrive strictly according to popularity. The scheme of optimizing the acceptance rate mentioned above can easily lead to an unbalanced load and increase the cost. Therefore, a goal of load balancing maintenance is introduced.

(1) For new requests arriving immediately r, due to the directive allocation threshold, r can only be connected to nodes that have not yet reached M, in order to minimize the load imbalance. Therefore, the following new optimization objectives have been added

$$\min\left\{\sum_{i=1}^{I}\sum_{j=1}^{J}\max\left(a_{ij}^{r}-a_{ij},0\right)\right\}, \tag{8}$$

where A^r is a constant matrix, the calculation method is: for v_r, assuming that M_j is the node deployed v_r with the smallest load, the corresponding element $a_{rj}^{r}=k_{rj}+1$; of course, the corresponding element $a_{rj}^{r}=k_{rj}+1, i\ne r$ on the other nodes. In addition, for the rest of the video v_i, $i\ne r$, the corresponding element is $a_{rj}^{r}=k_{rj}, \forall j\le 1$.

(2) For all subsequent arrival requests, in order to connect them to the minimum load node, it is necessary to ensure that each allocation threshold is larger than the number of sessions [18]. In addition, considering the continuity and randomness of request arrival, the difference between the allocation threshold and the number of sessions should also be related to the arrival of requests and other factors [19, 20]. Therefore, the following new optimization objectives have been added

$$\min\left\{\sum_{i=1}^{I}\sum_{j=1}^{J}\theta_i\times\max\left(a_{ij}^{*}-a_{ij},0\right)\times\gamma_j\right\}, \tag{9}$$

where A^* is a constant matrix, assuming that R_i represents the number of requests that v_i has not yet arrived, it can be approximately expressed as $R_i=\max(JCp_i-\sum_{j=1}^{J}k_{ij},0)$. For the node set $N_i=\{j\mid d_{ij}=1, j\le J\}$ of deployment v_i, the corresponding element $a_{ij}^{*}=k_{ij}+R_i/\sum_{j=1}^{J}d_{ij}$, and for the remaining nodes, the corresponding element $a_{ij}^{*}=0$. $\theta_{1\times I}$ and $\gamma_{J\times 1}$ are weight vectors, considering that popular video is more likely to affect the load distribution, the weight θ_i is desirable p_i; considering that lightweight nodes should allocate larger thresholds, the weight γ_i is desirable $(1-l_j/C)$.

From the effect point of view, the smaller the load of nodes, the larger the allocation threshold to undertake more requests for access. However, since this optimization strategy is adopted after the start of MEC, the load of each node is basically balanced, so the abovementioned average allocation processing can still achieve the desired effect.

In addition to the limitation of the cost of a single migration, the following constraints should be considered: the service capacity limitation of each server; the value range limitation of a_{ij}; and the principle of "no reduction in the number of actual sessions."

In summary, with session assignment matrix A as a decision variable, the migration computation model can be expressed as follows:

$$\begin{aligned}&\text{Obj 1}:\min\left\{\sum_{i=1}^{I}\sum_{j=1}^{J}\max\left(a_{ij}^{r}-a_{ij},0\right)\right\},\\&\text{Obj 2}:\min\left\{\sum_{i=1}^{I}\sum_{j=1}^{J}\theta_i\times\max\left(a_{ij}^{*}-a_{ij},0\right)\times\gamma_j\right\},\end{aligned} \tag{10}$$

where A and A^* are const matrix.

Subject to:

$$\begin{aligned}&\sum_{i=1}^{I}\sum_{j=1}^{J}\max\left(k_{ij}-a_{ij},0\right)\le M,\\&\sum_{j=1}^{J}a_{ij}\ge\sum_{j=1}^{J}k_{ij},\forall i\le I,\\&\sum_{j=1}^{J}a_{ij}\ge\alpha p_i,\forall i\le I,\\&\sum_{i=1}^{I}a_{ij}\ge 1/J,\forall j\le J,\\&0\le a_{ij}\le d_{ij},\forall i\le I,\forall j\le J.\end{aligned} \tag{11}$$

3.4. Scheduling Algorithm Based on Reinforcement Learning

3.4.1. Choice of Behavior Strategies. This paper chooses deterministic behavior strategies and defines a function μ, which is expressed as:

$$a_t=\mu(s_t). \tag{12}$$

The behavior of each step can be obtained by calculating function μ. Function μ is simulated by using convolutional neural network. The network is a strategy network with a parameter of θ^{μ}. A function $J(\mu)$ is used to measure the

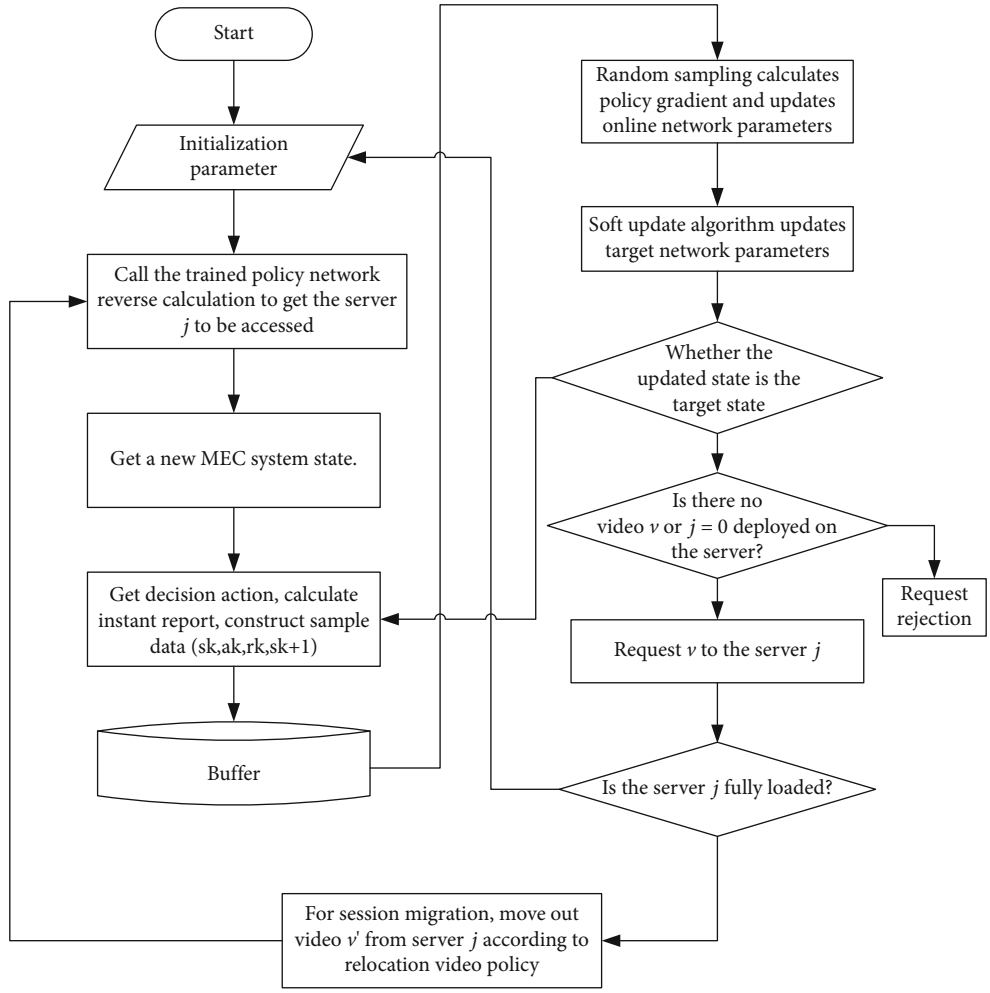

Figure 4: Strengthened learning algorithms for video session migration flow chart.

performance of strategy μ, which is defined as:

$$J(\mu) = E[Q^{\mu}(s_t, \mu(s_t))], \tag{13}$$

where s_t is the state of the system, $Q^{\mu}(s_t, \mu(s_t))$ is in each state, if the action a_t is selected according to policy μ, the Q value can be generated, that is, $J(\mu)$ is the expected value of $Q^{\mu}(s_t, \mu(s_t))$ when policy μ. Therefore, the optimal behavior strategy is the strategy μ which maximizes $J(\mu)$, that is,

$$\mu = \arg\max (J(\mu)). \tag{14}$$

Network input is MEC system state, that is, video request matrix, video deployment matrix, session distribution matrix, and server adjacency matrix with size $J \times J$. The eigenvectors of the video request matrix and the video deployment matrix represent the deployment information of the current video on the server. The size of the eigenvectors is $I \times J$. The three eigenvectors are connected through concat layer. Finally, the probability distribution of server number is obtained by using Softmax classifier. The dimension is d, and the decision-making action is the server number corresponding to the maximum probability.

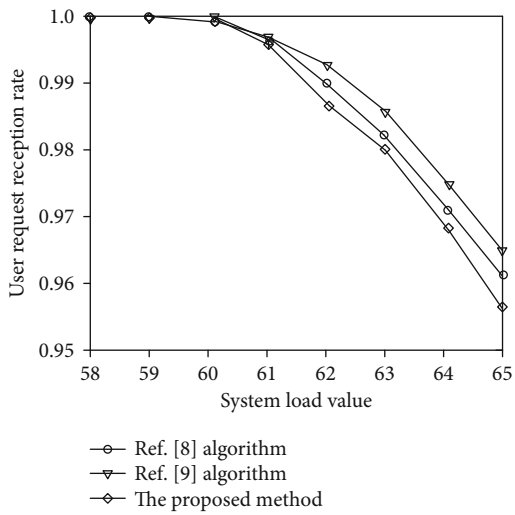

Figure 5: The relation between the receiving rate of user request and system load under Zipf distribution.

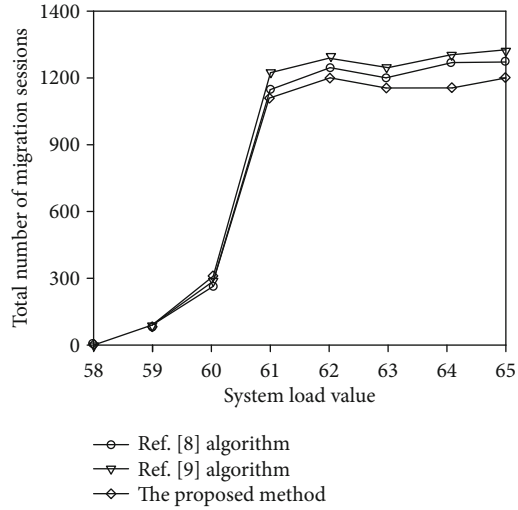

Figure 6: The relationship between total migration sessions and system load under Zipf distribution.

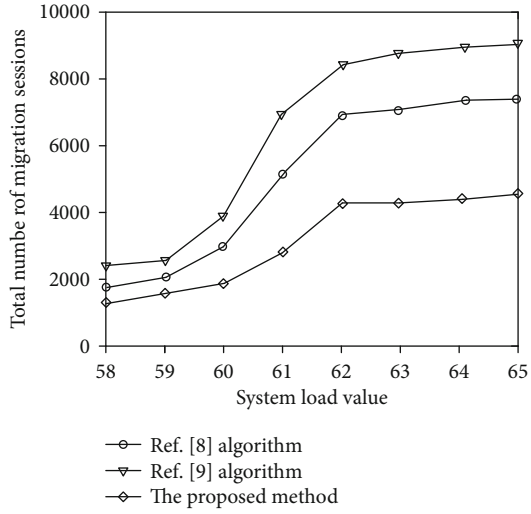

Figure 7: The relation between running time and system load under Zipf distribution.

In order to make it more exploratory, on the basis of the deterministic strategy, behavior search is added, that is, 30% of the actions are randomly selected in the optional action space, and the remaining actions are the output of the strategy network.

3.4.2. Iterative Value Calculation. In this paper, the convolution neural network is used to simulate Q function. The network is called Q network. Its parameter is θ^Q. The model of Q network is shown in Figure 3.

The input of Q network is the MEC system state and action vector, and the action vector is the result of transforming the probability distribution vector of the output of policy network into one-hot vector, the size of which is $I \times J$. In network training, input sample data is highly correlated with time, and direct training is not easy to converge. In order to break the correlation between data, the method of "experience playback" is used to save the generated sample data into the buffer, and the sample data used in training is randomly extracted from the buffer.

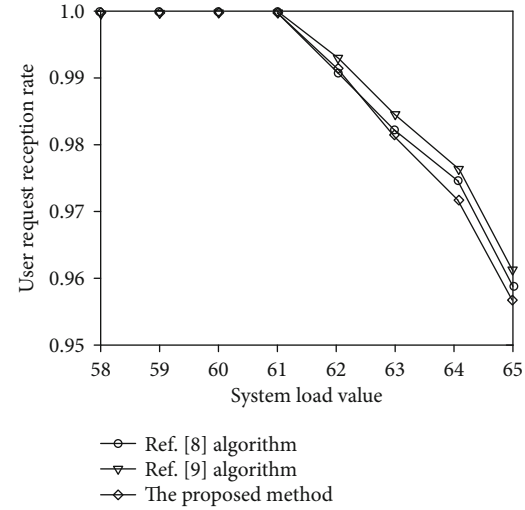

Figure 8: The relation between the receiving rate of user request and system load under random uniform distribution.

In the process of network training, this paper uses the target network method to establish the copy θ^μ and θ^Q of the policy network and Q network to calculate the target value, and then update the original network slowly in the proportion of τ. Through this network learning method, the learning process will be more stable and convergence will be more guaranteed. The flow chart of reinforcement learning algorithm based on deterministic strategy gradient is shown in Figure 4.

4. Experiment and Analysis

4.1. Parameter Setting. In the environment of the MEC system simulation, the environment parameters are as follows: the total number of streaming media servers $J = 30$, the capacity of each streaming media server$S = 40$, the maximum number of service sessions $C = 100$, and the number of video types $I = 350$. At the same time, assuming that the arrival rate of users' requests obeys Poisson distribution of ζ requests per minute, the value of ζ ranges from 58 to 65. The average playback time is set to 30 minutes, and the system can support 2000 (JC) user video requests concurrently in one playback time. Therefore, when $\zeta = 65(1900/30)$, the system reaches full load. The video content requested by users obeys Zipf distribution and random uniform distribution, respectively.

In order to analyze the effectiveness and practicability of deep reinforcement learning algorithm, this paper programmed on tensorflow platform and applied it in the MEC system session scheduling strategy. The parameters of the algorithm are as follows: the learning rate of policy network is 0.0001, the learning rate of Q network is 0.001, the discount coefficient is 0.95, the capacity of buffer is 1,00000, the preheating coefficient of buffer is 1,000, the number of iterations is 100,000, the upper limit of time step is 60 steps,

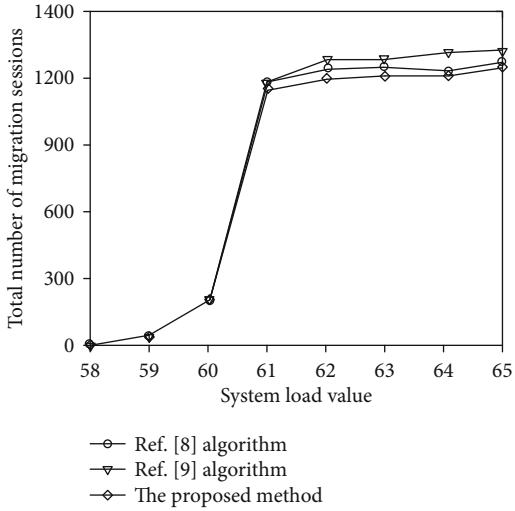

Figure 9: The relationship between total migration session number and system load under random uniform distribution.

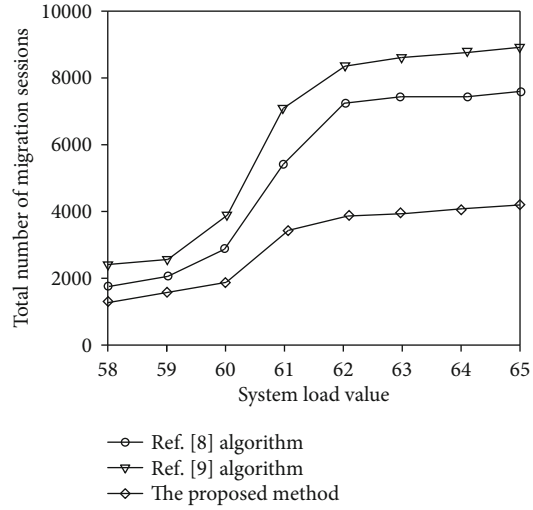

Figure 10: The relation between running time and system load under random uniform distribution.

the number of samples for each iteration is 600, and the weight coefficient in the return function is $\omega_1 = 0.8$, $\omega_2 = 0.1$, $\omega_3 = 0.1$.

4.2. Result Analysis. In this paper, according to the parameters set in Section 4.1, the deep neural network training is carried out. The trained network model is used in the MEC system simulation experiment, and the simulation time is set to 300 minutes. In order to better reflect the effect of algorithm optimization, under the same experimental conditions, the proposed algorithm is compared with Ref. [8] algorithm and Ref. [9] algorithm. In this paper, user request receipt rate, total number of migrated sessions, and running time are used as performance evaluation indicators.

Set the video content requested by the user to follow Zipf distribution. Figures 5–7 show the relationship between the user request reception rate, the total number of migration sessions and the running time of the simulation algorithm, and the system load under this condition, respectively.

As can be seen from Figures 5 and 6, in the case of low load ($\zeta \leq 61$), the user request reception rate and the total number of migration sessions of this method are basically the same as Ref. [8] algorithm and Ref. [9] algorithm. In the case of high load ($\zeta > 61$), the receiving rate of user requests and the total number of migrating sessions of this method are lower than Ref. [8] algorithm and Ref. [9] algorithm.

Compared with Ref. [8] algorithm and Ref. [9] algorithm, the average receipt rates of user requests in this method are reduced by 0.85% and 1.72%, respectively, and the total number of migrated sessions is reduced by 3.55% and 5.29%, respectively. The result shows the advantage of reinforcement learning. Because session transfer is cost-effective, in order to obtain greater returns, the decision-maker constantly adjusts the decision-making actions and ultimately reduces the cost of transfer, while guaranteeing a higher request reception rate.

As can be seen from Figure 7, for both low load and high load, the running time of the proposed algorithm is better than Ref. [8] algorithm and Ref. [9] algorithm, and the running time is shortened by 39.98% and 54.54% on average, respectively. Because in the process of user requesting access, the session allocation threshold needs to be constantly updated by method Ref. [8] algorithm, which leads to a lot of computation, and method Ref. [9] algorithm needs to be constantly overlapped. In order to find the optimal solution, the deep reinforcement learning method used in this paper only needs to make scheduling decisions through the trained strategy network, which has less computational complexity and improves efficiency.

In order to evaluate the adaptability of the proposed algorithm, a random uniform distribution of video content requested by users is set up. Figures 8–10 show the relationship between the request reception rate, the total number of migration sessions, and the running time of the simulation algorithm and the system load under this condition, respectively. Compared with Ref. [8] algorithm and Ref. [9] algorithm, the average receipt rate of user requests in this algorithm is reduced by 0.41% and 1.19%, the total number of migrated sessions is reduced by 3.64% and 6.57%, respectively, and the running time is reduced by 45.28% and 56.03%, respectively. The experimental results show that the proposed algorithm has a certain degree of self-adaptability. When the distribution of user requests changes, the scheduling strategy can still be adjusted in the training process, resulting in a lower migration cost and a higher user request reception rate.

In summary, the proposed deep reinforcement learning-based session scheduling strategy for streaming media edge cloud not only achieves better request access effect but also has lower migration cost. More importantly, it has a great speed advantage, that is, shorter running time. At the same time, it has strong adaptability in an uncertain MEC system environment.

5. Conclusion

In order to achieve efficient and smooth resource scheduling for streaming media service system in cloud mode, this paper proposes a video stream session migration method based on deep reinforcement learning. The method transforms session migration problem into reinforcement learning problem; defines state space, action set, and return function; calculates session volume according to load; and uses convolutional neural network to fit behavior selection strategy function and action-value function. The experimental results show that compared with the methods of Ref. [8] algorithm and Ref. [9] algorithm, this strategy can reduce the migration cost and shorten the running time.

This paper only considers the video session request access server as the output of Q network. Later research focuses on the video session request access server and the video session moved out of the server as the output of Q network, in order to improve the migration method of streaming media edge cloud session and extend the application object to dynamic video session.

Data Availability

The data included in this paper are available without any restriction.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

We wish to express their appreciation to the reviewers for their helpful suggestions which greatly improved the presentation of this paper. This work was supported by the Henan Province Science and Technology Project (142102210366), Henan Public Security Think Tank Project (No. 2020-25), and Henan Police College Education and Teaching Reform Research and Practice Project (No. 2020-9).

References

[1] K. Kaur, S. Garg, G. S. Aujla, N. Kumar, J. J. P. C. Rodrigues, and M. Guizani, "Edge computing in the industrial internet of things environment: software-defined-networks-based edge-cloud interplay," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 44–51, 2018.

[2] R. Kiruthiga and D. Akila, "Heterogeneous fair resource allocation and scheduling for big data streams in cloud environments," in *2021 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM)*, pp. 128–132, Dubai, United Arab Emirates, 2021.

[3] P. Pradhan, P. K. Behera, and B. N. B. Ray, "Improved max-min algorithm for resource allocation in cloud computing," in *2020 Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pp. 22–24, Waknaghat, India, 2020.

[4] E. Korpeoglu, C. Sahin, D. Agrawal, A. E. Abbadi, T. Hosomi, and Y. Seo, "Dragonfly: cloud assisted peer-to-peer architecture for multipoint media streaming applications," in *2013 IEEE Sixth International Conference on Cloud Computing*, pp. 269–276, Santa Clara, CA, USA, 2013.

[5] J. Tongquan, Z. Wang, and X. Hongsheng, "Session migration strategy for streaming media edge cloud based on dynamic threshold allocation," *Computer Engineering*, vol. 43, no. 1, pp. 55–60, 2017.

[6] I. Baumgart, B. Heep, and S. Krause, "Over Sim: a scalable and flexible overlay framework for simulation and real network applications," in *2009 IEEE Ninth International Conference on Peer-to-Peer Computing*, pp. 87-88, Seattle, WA, USA, 2009.

[7] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[8] T. Jiang, Z. Wang, Z. Chen, and H. Xi, "An adaptive strategy of online session migration for Streaming Media Edge Cloud," in *2016 35th Chinese Control Conference (CCC)*, pp. 5278–5283, Chengdu, China, 2016.

[9] Z. Chen, Z. Wang, and H. Xi, "A dynamic two-phase schedule strategy for streaming media edge cloud," in *2017 29th Chinese control and decision conference (CCDC)*, pp. 2281–2286, Chongqing, 2017.

[10] R. Viola, A. Martin, M. Zorrilla, and J. Montalbán, "MEC proxy for efficient cache and reliable multi-CDN video distribution," in *2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–7, Valencia, Spain, 2018.

[11] M. Liu, Y. Teng, F. R. Yu, V. C. M. Leung, and M. Song, "A mobile edge computing (MEC)-enabled transcoding framework for Blockchain-based video streaming," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 81–87, 2020.

[12] Y. Liu, B. Du, S. Wang, H. Yang, and X. Wang, "Design and implementation of performance testing utility for RTSP streaming media server," in *2010 First International Conference on Pervasive Computing, Signal Processing and Applications*, pp. 193–196, Harbin, China, 2010.

[13] H. Wang, J. Li, C. Zhao, and Z. Ying, "Design of an embedded streaming media server in video monitoring," in *2013 Ninth International Conference on Natural Computation (ICNC)*, pp. 1324–1328, Shenyang, China, 2013.

[14] Q. Fan and X. Wang, "Design of streaming media server based on softswitch platform," in *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, pp. 288–290, Zhangjiajie, China, 2012.

[15] A. Argyriou, "Link scheduling for multiple multicast sessions in distributed wireless networks," *IEEE Wireless Communications Letters*, vol. 2, no. 3, pp. 343–346, 2013.

[16] Yongfei Zhang, Shiyin Qin, and Zhihai He, "Fine-granularity transmission distortion modeling for video packet scheduling over mesh networks," *IEEE Transactions on Multimedia*, vol. 12, no. 1, pp. 1–12, 2010.

[17] S. Zhang and C. Chan, "Provisioning of survivable multicast sessions in wavelength-routed optical networks with scheduled traffic," *Journal of Lightwave Technology*, vol. 29, no. 5, pp. 685–690, 2011.

[18] Y. Han, Z. Wang, S. Chen, G. Li, X. Zhang, and X. Yuan, "Interactive assigning of conference sessions with visualization and topic modeling," in *2020 IEEE Pacific Visualization Symposium (Pacific vis)*, pp. 236–240, Tianjin, China, 2020.

[19] W. Kuo and C. Wang, "Robust and optimal opportunistic scheduling for downlink 2-flow inter-session network coding with varying channel quality," in *IEEE INFOCOM 2014- IEEE Conference on Computer Communications*, pp. 655–663, Toronto, ON, Canada, 2014.

[20] B. Kim and J. Lee, "Opportunistic resource scheduling for OFDMA networks with network coding at relay stations," *IEEE Transactions on Wireless Communications*, vol. 11, no. 1, pp. 210–221, 2012.