QoE-Driven Content-Centric Caching With Deep Reinforcement Learning in Edge-Enabled IoT

Xiaoming He
College of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing, CHINA

Kun Wang
Department of Electrical and Computer Engineering, University of California, Los Angeles, CA, USA

Wenyao Xu
Department of Computer Science and Engineering, University at Buffalo, State University of New York, Buffalo, USA

Digital Object Identifier 10.1109/MCI.2019.2937608
Date of current version: 14 October 2019

Abstract—Edge-enabled Internet of Things (IoT) services for users are subject to intelligent management of content-centric caching. Although managing edge caching can reduce storage cost and transmission latency, maintaining a high Quality of Experience (QoE) of caching is still a crucial challenge. In this environment, we study QoE-based content-centric caching. To evaluate the qualities of edge-enabled IoT, we introduce a QoE model which can grasp the influencing factors: (1) storage cost, based on available bandwidth, and (2) transmission latency, depending on the Signal-to-Interference-plus-Noise Ratio (SINR) and caching capacity. As the requirements and signals

Corresponding Author: Kun Wang and Wenyao Xu (E-mail: wangk@ucla.edu, wenyaoxu@buffalo.edu).

1556-603X/19©2019IEEE

are stochastic, we use a Reinforcement Learning (RL) architecture to jointly determine the Q-value. Estimating the Q-value, constrained by a maximum QoE, can be conducted in a Deep Neural Network (DNN) approximator, as the states and action spaces are on a large scale. Unfortunately, training DNN models can lead to RL instability. To address this issue, fixed target network, experience replay, and adaptive learning rate methods are proposed to balance the Q-value accuracy and accelerate stability in Deep RL (DRL). Experimental results indicate that our approach can gain a higher value of QoE, compared to existing methods.

The growing number of edge users inclined towards popular content dataflow, such as social dialogue, video-streaming, online games, and web surfing, oblige service providers to purchase new technologies, offering a satisfactory Quality of Experience (QoE) [2].

I. Introduction

Current electronic devices (i.e., smart phones, notebooks, mobile vehicles, etc.) are connected to the Internet of Things (IoT), resulting in a never-before-seen dataflow [1]. The growing number of edge users inclined towards popular content dataflow, such as social dialogue, video-streaming, online games, and web surfing, oblige service providers to purchase new technologies, offering a satisfactory Quality of Experience (QoE) [2]. Unfortunately, during peak periods of dataflow, backhaul links faced with congestion and a large-scale Signal-to-Interference-plus-Noise Ratio (SINR) lead to a lower QoE for edge users.

Transferring peak hour dataflow to off-peak hours can ease the drawback. Caching technology achieves this transfer by grabbing popular content and storing it in Base Stations (BSs) during the off-peak, and re-using them during peak periods. To manage content-centric caching intelligently, BSs with a caching unit and a computing unit need to learn the available observations. Deep Reinforcement Learning (DRL) technology [3] can facilitate the management of BSs, in which the intelligent caching and computing units can be learned and estimated.

Most previous studies have focused on content-centric caching in edge-enabled IoT. Enabling BSs to learn and estimate unknown popular content is necessary for a caching service. Liu et al. [4] introduced an approach, based on Reinforcement Learning (RL), to mold dataflow controlled by download-rate of video clients, cutting down bit-rate oscillations. Li et al. [5] proposed a model, based on a Markov Decision Process (MDP), which considered the energy efficiency for the decision of caching issue, where a distributed mechanism hinging on the popularity of content updated the caching dataset in BSs. Morozs et al. [6] considered a local and global MDP model, as well as an RL-based algorithm, to seek the optimal policy, followed by a linear function approximation on the basis of Q-learning, used to reduce memory requirements. Wei et al. [7] discussed an Actor-Critic DRL (AC-DRL) framework using the solutions of decision-caching, minimizing the average transmission latency. A Deep Neural Network (DNN) was used for the value-function estimates in the critic part. However, training that DNN in RL causes the learning algorithm to diverge or be unstable.

Few research efforts have been paid to the caching problem with the QoE optimal issue, as well as the metrics for caching performance. Note that, QoE can be understood as the evaluation of a Quality of Service (QoS) mechanism. The International Telecommunication Union (ITU) has defined QoE as the whole accepted application or service in some environments (i.e., Long-Term Evolution (LTE) networks, future 5G networks, edge networks, etc.). Considering the transmission parameters, the caching performance with QoE was studied for the failure probability and transmission latency. Cui et al. [8] derived the QoE-based caching placement issue in mobile edge for dynamic video-streaming. Caching representations were selected in the edge environment for each edge server, in terms of reducing the storage cost of the BSs while maintaining a high value of QoE. Meanwhile, Liu et al. [9] investigated the issue of optimal content caching management in an HTTP bit-rate streaming environment, maximizing the personal-content QoE under a limited storage budget for edge users. Wu et al. [10] jointly presented a co-operative mechanism with an improvement in content delivery efficiency, based on caching-coded methods, enhancing the value of QoE for users. Aiming to improve the energy efficiency, we establish a model of energy consumption. As a result, the caching issue needs to be optimized for the QoE performance with two metrics, that is, transmission latency and storage cost.

Motivated by the research efforts mentioned above, we pay particularly close attention to the issue of content-centric caching with QoE in an edge-enabled IoT. Aiming at caching intelligence in this environment, a novel model is proposed to balance the QoE and the caching management. Based on the analysis of RL, we give a decision-making model with the purpose of Q-value. Finally, improved DNN is used to estimate the Q-value with the subject of maximizing QoE because of large states and action spaces. Furthermore, the proposed algorithm can achieve a trade-off between Q-value accuracy and DRL-accelerated stability. We summarize our contributions as follows:

- ❏ In the framework of an edge-enabled IoT, we introduce a model aimed at improving the QoE value. By comprehensively considering the influence factors of transmission latency and storage cost, we make a sustained effort to seek a trade-off between the content-centric caching quality and the user experience.
- ❏ To enhance the satisfaction of QoE for intelligent caching in this environment, a novel DRL algorithm for our studied

In the paper, we pay close attention to the issue of caching with QoE in an edge-enabled IoT. A novel QoE model and the improved DRL algorithm are studied jointly for intelligent content-centric caching.

issue of QoE-maximization is devised. Then, we intend to seek out a balance between Q-value accuracy and DRL-accelerated stability.

- ❑ Extensive experiments are conducted to make evident the superiority of our proposal over existing approaches.

The organization of the rest of this paper is as follows. The relevant existing literature is summarized in Section II. Section III proposes some models for content-centric caching. A caching strategy is presented in Section IV. Some experimental evaluations are performed and the analysis of the results is given in Section V. Finally, Section VI concludes this paper.

II. Related Work

In this section, the correlations of the literature can be segmented into two classes: (1) Caching with QoE, and (2) DRL methods.

A. Caching with QoE

Many recent studies have paid attention to leveraging the caching in IoT. From the point of view of adaptive-rate, Fazio et al. [11] investigated the issue of a sudden rate change and bit-rate oscillation occurring through the interaction between user clients and caching units, and an approach was proposed to eliminate oscillations by using shaping. Mork et al. [12] studied the trade-off in the cloud between caching and computing transcoding, and proposed a partial transcoding scheme of cost-efficiency with the purpose of caching content management on the basis of user points. To improve the user QoE, Tasaka et al. [13] proposed a logarithmic QoE model driven by observational results. The work used a convex optimization problem to formulate a cache management problem in the context of adaptive streaming. Furthermore, the authors provided a framework to analyze this engineering problem. Mu et al. [14] introduced an in-network policy of video caching for the information-centric network, enhancing user QoE depending on the distribution of content popularity, in terms of average user throughput. Aiming towards the decision-making of the caching replacement, Pang et al. [15] derived an adaptation algorithm of video caching driven by QoE, relying on the popularity of content chunks and the network bandwidth of the downlink. Unfortunately, most existing studies did not analyze these works in a holistic manner and only concentrated on operational consumption or delivery latency. As the proposed works neglected the effect that the combination of cost and latency have in edge-enabled IoT, we intend to study content-centric caching.

B. DRL Methods

In contrast to the traditional methods of machine learning, a DNN has an advantage in terms of accuracy for value-estimation. As a result, we employ DRL to exploit DNN and RL for decision-making.

A value-based DRL was used for value function estimation of the state-action. This work derived a Deep Q-learning Network (DQN) as a common learning architecture, which was used to classify Atari games with a performance at a human level [16]. In order to reduce the error, the algorithm of double Q-learning [17] was proposed, due to Q-value over-estimation in the present state of potential actions caused by regular DQN. The over-estimation reduction was substantial and the training process was faster when using this algorithm, and the algorithm was also used for large function approximation. The function of advantages and state-value were shown separately by a dueling DQN architecture [18]. Two separate approximators were integrated with an individual Q-function of this algorithm at the final layer, as a consequence, Atari games outperformed the results in this architecture. Compared with the existing works, which had made efforts to solve content-centric caching in terms of accuracy, tossing DNN at RL results in a poor algorithm and may make DRL unstable.

In the paper, we pay close attention to the issue of caching with QoE in an edge-enabled IoT. A novel QoE model and the improved DRL algorithm are studied jointly for intelligent content-centric caching.

III. System Model

In this section, we summarize the caching model for the QoE model with two metrics, i.e., transmission rate and storage cost.

A. Network Model

Suppose that a local edge-enabled IoT with a single BS is connected to the backbone with low bandwidth, high latency, and the backhaul link, as shown in Figure 1. Note that, a BS is equipped with M, $M \in N^+$, local caching units to store content chunks. Each content chunk $m_t \in M$, $t = 1, 2, 3, \ldots$, is assumed to have different bit-rates L_{mt}. Each m_t, $t = 1, 2, 3, \ldots$, is also assumed to contain τ_{m_t} seconds. We can see that the size of each m_t is $L^{bit}_{m_t} = L_{m_t} \cdot \tau_{m_t}$. Each caching unit works in a first-in-first-out method. That is, a new m_t replaces the old m_t when an old m_t makes a decision to leave.

We further suppose that the BS selects M content chunks from the number of total files $F \geq M$ in the cloud server, and stores them for possible utilization in time slot t. The binary variable $a^{ca}_{m_t}$, denotes whether to store the requesting m_t or not. If $a^{ca}_{m_t} = 1$, then m_t is cached in BS, incurring zero cost and reducing the transmission latency. Otherwise, the BS can grab m_t from F, leading to sizable cost and a large latency, along with the satisfaction of QoE for users. Note that, $L^{c}_{m_t}$ can be

seen as the bit-rate of m_t. If m_t is not cached, we simply set $L^c_{m_t}$ as zero. The BS needs to intelligently select any m_t for low cost and latency in a caching service.

B. Caching Model

When an edge user requests m_t in t, then m_t is transferred from BS if it is cached. Otherwise, BS can grab the m_t from the cloud server, and then the m_t is delivered to the edge user. The transmission latency from BS to edge user is relatively small, and it has a little effect on the caching decision. We assume that the wired round-trip transmission latency is the same constant d [19] for transferring the requests from the cloud server to the BS. In addition, if the cache-hit-probability is increased, the requirements of the backhaul link can be reduced and the average transmission latency, in the same way. That is, the probability of caching a special m_t is in direct proportion to content popularity.

Suppose that the popularity of each requesting m_t is defined as $p_{m_t} \in [0, 1]$. We are more in favor of caching content chunks with higher popularity, as the popularity is known. Consequently, the function of transmission latency is denoted as follows:

$$La = \frac{1}{M}\sum_{m_t=1}^{M}\frac{L^{bit}_{m_t}}{r_{m_t}} + \frac{1}{M}\sum_{m_t=1}^{M} d(1-a^{ca}_{m_t})(1-p_{m_t}), \tag{1}$$

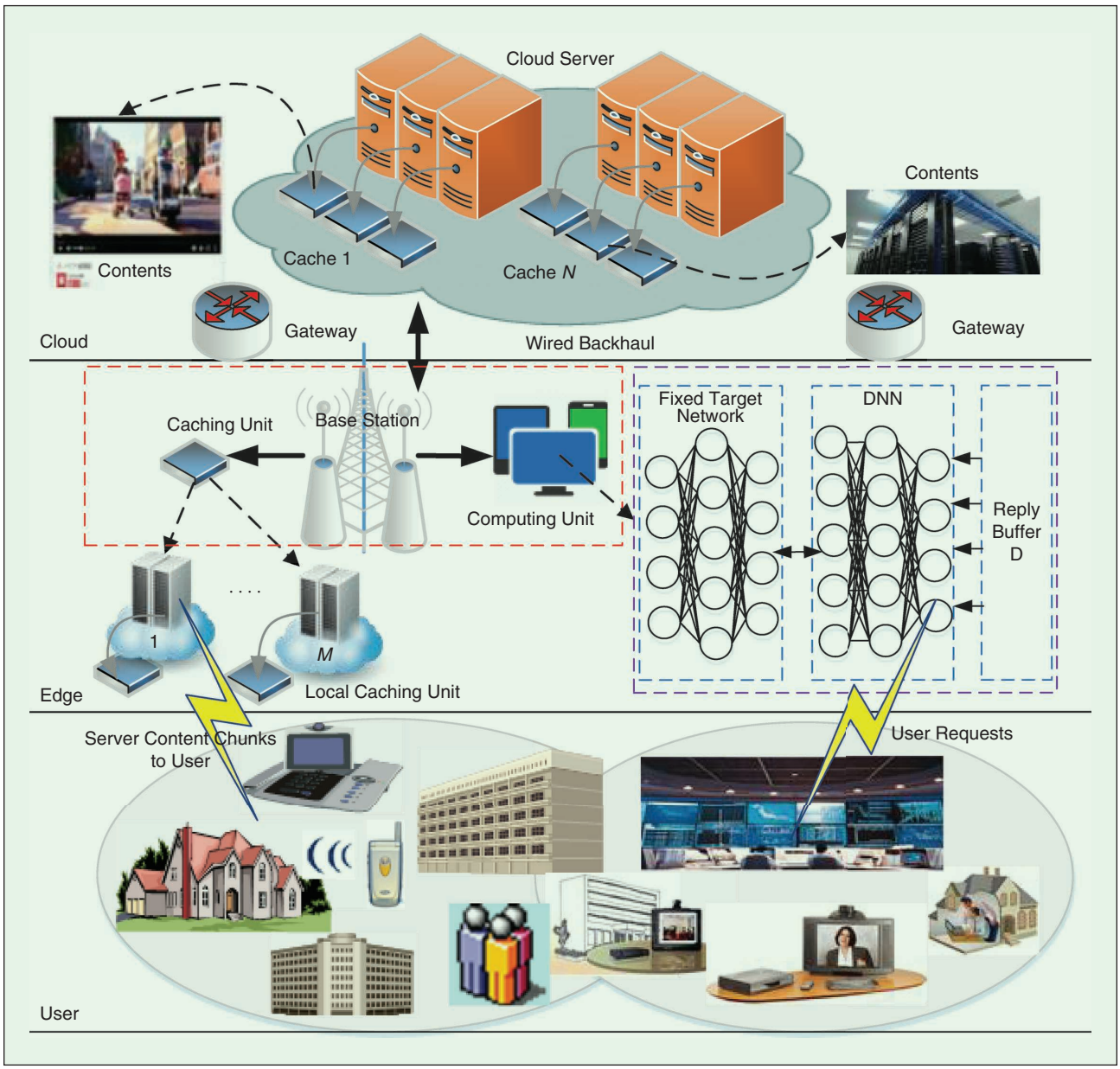

FIGURE 1 Illustration of edge-enabled IoT architecture.

Specifically, taking QoE into consideration from two perspectives is an emergency: transmission latency and storage cost are negatively proportional to QoE.

where the left part is the average wireless transmission latency for all M requirements, and r_{m_t} refers to the SINR, which exists for different content chunks. The existing interference is inversely proportional to transmission latency in a certain process. The right part is the average wired round-trip transmission latency, which is influenced by content popularity.

On the other hand, caching m_t can cause storage cost in M units of the BS, which is positively correlated to the size $L_{m_t}^{bit}$ of m_t. Note that, if the m_t is cached, only the increased parts of M in BS generate storage cost. Then, the average storage cost is calculated as follows:

$$Co = \frac{1}{M} \sum_{m_t=1}^{M} \eta \cdot (L_{m_t} - L_{m_t}^{c})^{+} \cdot \tau_{m_t}, \tag{2}$$

where η is the parameter of storage pricing, indexed in [8], and $(x)^{+} = \max\{0, X\}$ [13].

C. QoE Model for Content-Centric Caching in Edge-Enabled IoT

The Index of QoE Level

To improve the experience of the user in an edge-enabled IoT, we measure QoE as the quality perceived by the service coming from content-centric caching. On the basis of this study, we show two metrics to assess QoE commonly. The first one is based on the transmission latency, which is related to the popular content chunks caused by the requirements of users. A lower transmission latency means a higher quality of experience for the user. The second part utilizes the storage cost, especially for diverse requirements. The higher the storage cost is, the lower the quality of experience for the user becomes. The effect on QoE in an edge-enabled IoT is exerted jointly by these two terms. Detailed evaluation methodologies are described as follows:

QoE Model

The transmission latency in an edge-enabled IoT can be related to QoE. When the transmission latency La is high, requests for a lower transmission latency will be made by users. In addition, the storage cost can be added as another term of QoE. If the storage cost Co is not low, the BS can fetch more popular m_t to caching units at a lower cost. Therefore, we can obtain QoE as follows:

$$QoE = -La - \xi Co, \tag{3}$$

where $\xi > 0$ is a weighting parameter between the transmission latency and storage cost.

D. Optimization Objective

One of the most critical issues in edge-enabled IoT for content-centric caching performance is QoE, which can be affected by many factors. Specifically, taking QoE into consideration from two perspectives is an emergency: transmission latency and storage cost are negatively proportional to QoE. Finally, the objective of maximizing the QoE is as follows:

$$\text{Maximize} \quad QoE(La, Co). \tag{4}$$

IV. Learning-Based Caching Strategy

In this section, we resort to the RL method to obtain the Q-value by learning a stochastic policy $\pi()$. Then, the improved DNN, based on fixed target network [20], experience replay [21], and adaptive learning rate [22] methods, is proposed to balance the Q-value accuracy and DRL-accelerated stability. Note that, Q-value denotes the value of QoE.

A. RL-Based Problem Formulation

In this paper, we consider an arbitrary state space. Therefore, the above stochastic optimization problem can be formulated as an MDP. We use the RL method to learn the best policy, with the objective of maximizing QoE through large-scale training. This training is in the context of the present state, which can be transferred, based on the transition probabilities to the other states, incompletely. The trait of RL architecture appears in that the decision-making by the agent is in a non-discretized approach. Therefore, BS can realize the RL agent and can gather all information, containing states, and decision-making for all requests. Note that, the following parts give descriptions of state and action.

State: In t, the M requests of m_t can be acquired by the BS. A series of SINR and bit-rate are included in every request. The RL agent obtains the state, value of popularity, and size for each m_t. Therefore, the state in our problem is denoted by $s_t \in S$, including

- M: the amount of requested m_t in the case of the management fields of the BS.
- p_{m_t}: the popularity of m_t.
- $a_{m_t}^{ca}$: whether m_t is cached or not.
- $\Upsilon_{m_t,M}$: the $M+1$ size-vector produced by requesting $m_t \in M$.
- $L_{m_t}^{c}$: bit-rate of the cached m_t. If m_t is cached, then $L_{m_t}^{c} = 0$.

Action: Decision-making through the agent gives the certain m_t, which can be served by a certain BS. The purpose of the agent is to achieve the minimum average transmission latency and storage cost, based on the caching conditions of the m_t. Therefore, the action of RL is defined as $a_t \in A$, including

- $a_{m_t}^{ca}$: whether m_t is cached or not. If $a_{m_t}^{ca} = 1$, the storage units cache it. If $a_{m_t}^{ca} = 0$, the units do not cache it.
- $a_{m_t}^{bs}$: the size-vector $M+1$ of m_t, indicating which m_t can be served by the BS.
- a_{m_t}: bit-rate selection of m_t (i.e., L_{m_t}).

Reward: In the state of taking the action a_t, the reward r_t is obtained by the system. In the stochastic process, the value-expectation of a maximum long-term value which is accumulated does not refer to the immediate value-maximum. Therefore, the action-state is the acceptance of considering a more and more timely reward. In the approach of RL, the next rewards in the next situations and the immediate rewards are affected by the actions. With the purpose of considerable rewards, an RL agent can incline towards actions that efficiently have attempted at reward-production. The issue needs to be externalized in the transmission latency and storage cost manner, and the objective is to maximize the value of QoE. Therefore, the reward can be written as follows:

$$r_t = -QoE. \tag{5}$$

Value: Decision-making in t is intended to optimize the long-term performance. Suppose that, in the condition of the stochastic policy $\pi(a|s) = Pr(a_t = a|s_t = s)$, the action a_t can be obtained. Taking the probability of taking actions in the system states constructs this policy, referring to a mapping. Due to the value function, $\pi(a|s)$ can be evaluated and improved by the RL agent. Note that, the function refers to the cumulative value-expectation in the discounted rewards. The rewards can be received following $\pi(a|s)$ in the entire procedure. The value function of state-action for an agent with $\pi(a|s)$ is given by

$$Q^{\pi}(s, a) = E\left\{\sum_{K=0}^{\infty} \beta^k r_{t+k} | \pi, s_t = s, a_t = a\right\}, \tag{6}$$

which refers to (as in the above) the long-term reward of expectation when started at state s. The value $\beta \in (0, 1)$ is the discount factor to index decisions in the way that can be expected. Then, the optimal Q-value $Q^*(s, a)$ can be denoted and calculated by the Bellman optimality equation [23] as follows:

$$Q^*(s_t, a_t) = E\{r_t + \beta \max_{a_{t+1}} Q(s_{t+1}, a_{t+1} | s_t, a_t)\}. \tag{7}$$

As the state spaces are very large, it is impossible to calculate all Q values using the Bellman equation. Consequently, a neural network can be used to estimate value functions in RL, as the function approximator.

B. Q-Value Estimation with Improved DNN

According to the above-mentioned function approximators, DNN acts as the most efficient architecture for representation or feature learning [24]. Therefore, to estimate the value function, DNN is required.

A fully-connected DNN can be shown through the Q-value $Q_w(s, a)$. A series of weights $w = \{w_1, w_2, \ldots, w_n\}$ are used to parameterize the DNN. Consisting of a variety of neurons in each hidden layer, we compute some elements. The elements in the condition of a non-linear activation function can output a value through transferring the weight-input. In the layer i, the definition in the value-output of the j-th neuron is y_{ij}, which refers to

$$y_{ij} = f_{\text{act}}(w_i \cdot x_i + b_{ij}), \tag{8}$$

where f_{act} is the activation function. In layer i, w_i, $i = 1, 2, \ldots, n$, is the weight-input, x_i, $i = 1, 2, \ldots, n$, is the value-input, and b_{ij} is the bias.

The parameters of the system are optimized using a loss function with updating w. The loss function is denoted by $L(w)$. Recall that, the loss function shows the mean-squared error of the value-target, that is,

$$L(w) = E[r_t + \beta \max_{a_{t+1}} Q_w(s_{t+1}, a_{t+1}) - Q_w(s_t, a_t)]^2, \tag{9}$$

where $r_t + \beta \max_{a_{t+1}} Q_w(s_{t+1}, a_{t+1})$ is the value-target, and $r_t + \beta \max_{a_{t+1}} Q_w(s_{t+1}, a_{t+1}) - Q_w(s_t, a_t)$ is called the temporal-difference error.

However, due to the conflict between the targets and the relevance in a non-stationary environment, DNN can lead to instability, or cannot be able to learn effectively and cannot resolve the inconsistency of the RL algorithm. As a consequence, we utilize the fixed target network (F), experience replay buffer (E), and adaptive learning rate (L) methods to construct an FEL-DRL algorithm with the goal of accelerated stability in DRL.

The following proposal is the issue of the target in a condition of non-stationarity: at each iteration of the training procedure, the parameters of the value-target and the present parameter-estimation of DNN can synchronously change. If the variational value-target is changed with tuning parameters, the tendency of value-estimation is not to be inconsistent and the procedure of learning tends to be destabilized. Aiming to alleviate the condition of non-stationarity, another neural network is selected by the fixed target network to utilize as the target. The parameters of the target neural network are fixed by w^* at each procedure of the iterations. In addition, the parameters which are fixed by the target network in the time slot can update the values of the network parameter-estimation in the characteristics of the slower cycle.

Combining RL process with DNN estimation, the proposed FEL-DRL algorithm can be summarized. The main algorithm steps of FEL-DRL, with experience replay buffer, fixed target network, and adaptive learning rate $a_{t,h}$, are as follows.

The experience replay buffer is used to terminate the relativity of time within different training slots. The dataset is built because the experience of the agent is stored. To train the network, the batches of dataset are delivered in a random way. As a result, learning about what it is doing in a timely manner of the network is prevented. Then, the RL algorithm can be allowed to learn from a variety of past experiences. The replay buffer of experience stores $I = \langle s_t, a_t, r_t, s_{t+1} \rangle$ as a tuple. Finally, the parameter w can be updated by a mini-batch sample of I tuples in the DNN.

The transition probability and reward, following the distribution, are created by samples in the past t. The hth experience sample is $\langle s_t^h, a_t^h, r_t^h, s_{t+1}^h \rangle$. We propose the adaptive learning rate as follows:

$$\alpha_{t,h} = \alpha_0 \beta_0^{t-t_h}, \tag{10}$$

where the constant parameters α_0 and β_0 satisfy $0 < \alpha_0,\ \beta_0 < 1$.

Sample-failure is updated exponentially by the adaptive learning rate $a_{t,h}$ in t. The shorter the sample which is generated, the smaller the mistakes are in the transition probability and the average reward made. In the extreme case, when the sample is produced, the adaptive learning rate can achieve the maximum value of α_0.

The loss function appears in the following, with the acceptance techniques of the fixed target network and the experience replay buffer.

$$L(w) = E_D[r_t + \beta \max_{a_{t+1}} Q_{w^*}(s_{t+1}, a_{t+1}) - Q_w(s_t, a_t)]^2, \tag{11}$$

where w^* is the network-target parameter, D is the experience replay buffer, and the value-expectation is a mini-batch of samples coming from D (i.e., $\langle s_t, a_t, r_t, s_{t+1} \rangle \sim D$).

To calculate the loss function in the process of each iteration, the target neural network with the adaptive learning rate $a_{t,h}$ and the old fixed parameter is proposed. In addition, to minimize the loss function, the technology of the gradient descent method is given to update the parameter w, as follows:

$$\triangle_w = a_{t,h} [Q_t - Q_w(s_t, a_t)] \nabla_w Q_w(s_t, a_t), \tag{12}$$

where $Q_t = r_t + \beta \max_{a_{t+1}} Q_{w^*}(s_{t+1}, a_{t+1})$ denotes the value-target, and ∇_w refers to the partial derivative with respect to w.

C. FEL-DRL Algorithm

Combining RL process with DNN estimation, the proposed FEL-DRL algorithm can be summarized. The main algorithm steps of FEL-DRL, with experience replay buffer, fixed target network, and adaptive learning rate $a_{t,h}$, are as follows:

Step 1: The stochastic policy $\pi(a|s)$ with parameters is initialized by the agent. Then, the definition of the value function approximation $Q_w(s, a)$ is proposed to parameterize. The network-target is initialized using the weights $w_t,\ t = 1, 2, \ldots, n$.

Step 2: The agent produces an action a_t, in light of current state s_t and policy $\pi(a|s)$.

Step 3: The next state s_{t+1} in the environment and the reward r_t are observed in the RL. Then, the RL is employed to the experience replay buffer D for storing the tuple $I = \langle s_t, a_t, r_t, s_{t+1} \rangle$.

Step 4: A mini-batch of I is sampled from the replay buffer at random.

Step 5: For any sample $h \in I$, the DNN estimates the Q-value $Q_w(s_t^h, a_t^h)$. Then, the DNN calculates the error and adaptive learning rate $a_{t,h}$. Finally, the DNN with the averaged value updates its parameter w over the mini-batch, minimizing the loss function.

V. Performance Evaluation

In this section, a performance evaluation is conducted to validate the QoE model and the caching strategy algorithm.

A. Experimental Settings

In this evaluation, the caching capacity is set as $M = 100$. Each requested m_t is divided into $4s$ and is encoded at 5 different bit rates. Then, η is set as 0.03×10^{-3}. Finally, the average SINR is discretized into the set $\{1, 4, 9, 16, 32\}$ in t. A variety of content chunks can be set, with the total number being $F = 1{,}001$. The Zipf distribution is appropriate for the popularity of these F. Each edge user, with the random probability, selects the m_t which is required. Note that, the probability is in direct proportion to the popularity of m_t. The transmission latency, using a wired-transmission round-trip, is set as $d = 1$.

Considering low complexity for good performance, the amounts of neurons are set as 301 in the hidden layers of the DNN. To regularize the learning algorithm, the network-target is generated by the agent. The agent can stand in place of the parameters for the network-target, using the current parameter-estimation of their primary networks. Note that, the adaptive learning rate $\alpha_{t,h}$ is set as $\{0.00005, 0.00008, 0.0001, 0.00012, 0.00015\}$, once every 301 iterations. The size of an experience replay buffer is set as 10,001 for the training DNN. The buffer can randomly return a mini-batch of experiences chosen by the agent. The maximum beam is set as 1,001. In addition, the size of the minimum batch is set as 64. The maximum amount of steps in t is set to be 1,001.

Finally, we implement the algorithms named RL, DRL, AC-DRL, FE-DRL, and FEL-DRL in Matlab, and evaluate them using Tensorflow. Recall that, FEL-DRL is the scheme proposed in this paper. RL, DRL, AC-DRL, FE-DRL, and FEL-DRL are introduced in other studies. The useful simulator is adopted since it is designed to input realistic traces from all varieties of data sets in databases. We also use the Federal Communications Commission (FCC) [25] dataset, consisting of millions of nationwide broadband performance records in an edge-enabled IoT.

B. Experimental Results

Figure 2 compares the performance of the QoE models with different metrics. We can see that the QoE model using storage cost and transmission latency metrics obtains the best performance. The one-metric QoE model using the storage cost or

the transmission latency achieves a low performance. When we adopt three metrics, the rewards of the QoE model are the lowest. It appears that the performance of the QoE model relies, in a certain way, on those two metrics.

Figure 3 compares the performance of the schemes with different parameters. The ratios of RL, AC-DRL, and FE-DRL are 0.9, 0.6, and 0.8, respectively. However, the average ratio of FEL-DRL shows the best performance with changes of the adaptive learning rate $\alpha_{t,h}$. That is, the FEL-DRL scheme has the highest stability. Figure 4 shows the comparison of the transmission latency in the condition of the diverse mechanisms. With an increase of SINR, the average transmission latency decreases. Specifically, the average values of transmission latency under the RL, DRL, AC-DRL, FE-DRL, and FEL-DRL mechanisms are 6.0, 5.0, 4.8, 4.5, and 4.6 seconds, respectively.

Figure 5 compares the transmission latency, based on content popularity under different schemes. To clearly see the content-centric caching on the basis of FEL-DRL, five caching strategy categories are compared, with an increase of caching capacity from 100 to 1,000. We can see that the caching strategy, based on FEL-DRL, outperforms the other four mechanisms. Note that, the smaller the caching capacity is, the better our proposed method performs. The performance of the caching strategy, based on the content popularity, is very close to the proposed method, subject to the caching capacity being larger than 501. In other words, in an edge-enabled IoT, half of the content chunks can be cached. The average transmission latency using the caching method with FEL-DRL is inversely proportional to an increase in the caching capacity. Not all content-centric caching strategies has different performance when the caching capacity is over 1,001. That is to say, any m_t can be cached in the edge.

Figure 6 compares the storage cost under different schemes. Randomly-selected traces are set to be 5,000, from the FCC dataset. The average bandwidth is from 1 to 5 Mbps, avoiding simple cases so the maximum bit-rate can be chosen. We can see that the FEL-DRL approach outperforms the other baseline approaches. Especially, our proposed method achieves an average improvement of storage cost, compared to FE-DRL.

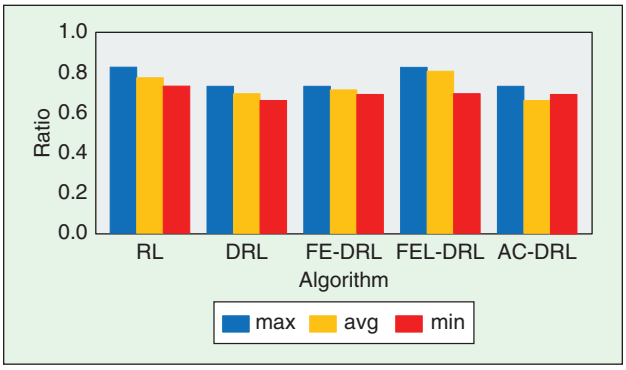

FIGURE 3 Algorithm performance.

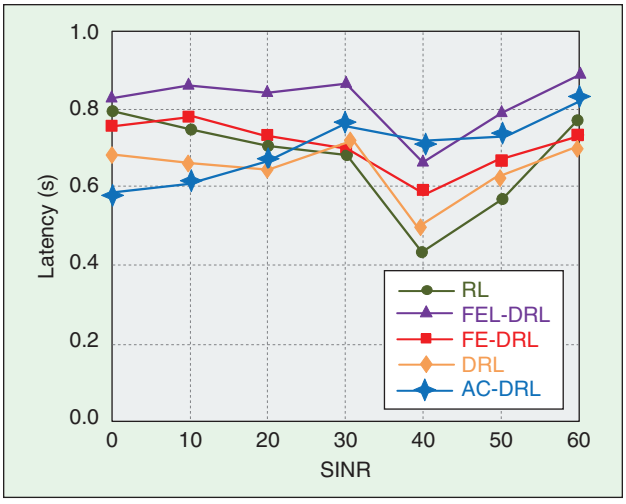

FIGURE 4 SINR versus transmission latency.

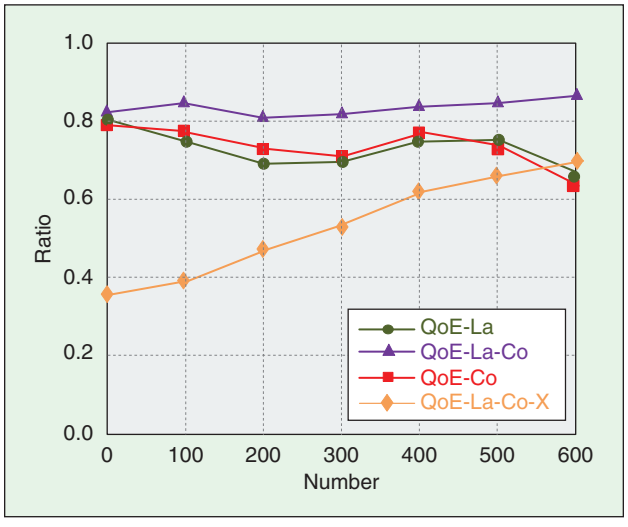

FIGURE 2 QoE model performance.

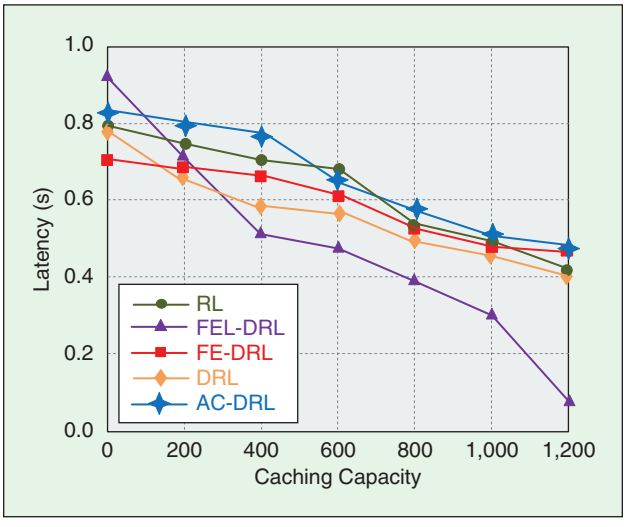

FIGURE 5 Caching capacity versus transmission latency.

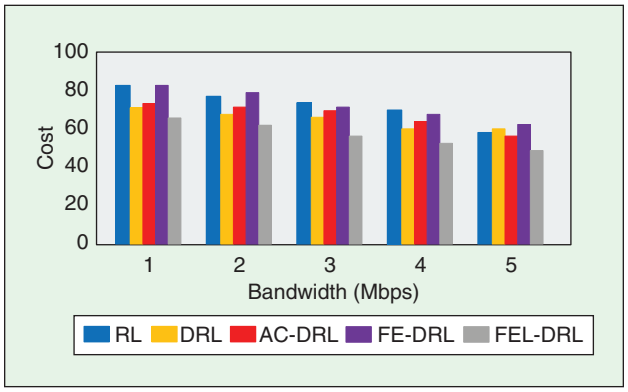

FIGURE 6 Bandwidth versus storage cost.

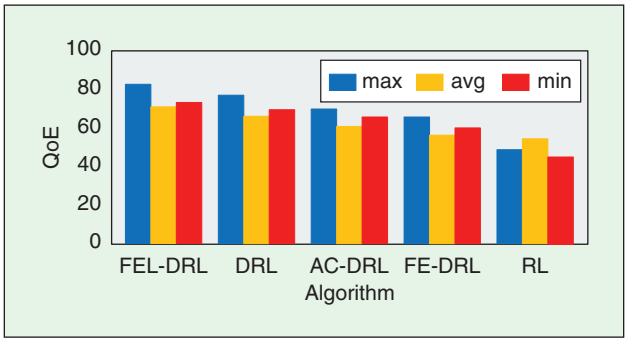

FIGURE 7 QoE versus different mechanisms.

With an increase in bandwidth, the improvement of the storage cost contributes to our proposed method significantly.

In the literature [10], it has emerged that the value of QoE in a satisfactory environment is usually above 60. The content-centric caching method, based on the FEL-DRL algorithm with the best choice of the transmission latency *La* and storage cost *Co*, brings the edge user a satisfactory QoE. In detail, we can see, from Figure 7, that the average QoE of FEL-DRL is 64, the maximum QoE of FEL-DRL is 70, and the minimum QoE of FEL-DRL is 62. In contrast to the other algorithms, the FEL-DRL algorithm generates a better QoE. From what has been discussed above, the QoE values are negatively proportional to transmission latency and storage cost.

VI. Conclusions

In this paper, we study content-centric caching in an edge-enabled IoT with the constraint of improvement of QoE. The main influencing factors for the QoE are analyzed. An RL method is used for Q-value decision-making, considering transmission latency and storage cost. Moreover, an FEL-DRL method is proposed to balance the Q-value accuracy and the DRL-accelerated stability, due to the large action spaces and states. Finally, the experimental results evaluate the high value of QoE given by the FEL-DRL algorithm.

Acknowledgment

This work was supported in part by National Natural Science Foundation of China under Grant 61872195, Grant 61572262 and in part by the U.S. National Science Foundation under Grant 1718375.

References

[1] X. Yang, X. Wang, Y. Wu, L. P. Qian, W. Lu, and H. Zhou, "Small-cell assisted secure traffic offloading for narrowband internet of thing (NB-IoT) systems," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1516–1526, June 2018.

[2] L. Jalal, M. Anedda, V. Popescu, and M. Murroni, "QoE assessment for IoT-based multi sensorial media broadcasting," *IEEE Trans. Broadcast.*, vol. 64, no. 2, pp. 552–560, June 2018.

[3] N. Javaid, A. Sher, H. Nasir, and N. Guizani, "Intelligence in IoT-based 5G networks: Opportunities and challenges," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 94–100, Oct. 2018.

[4] W. Liu, G. Qin, Y. He, and F. Jiang, "Distributed cooperative reinforcement learning-based traffic signal control that integrates V2X networks' dynamic clustering," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 8667–8681, Oct. 2017.

[5] T. Li, M. Ashraphijuo, X. Wang, and P. Fan, "Traffic off-loading with energy-harvesting small cells and coded content caching," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 906–917, Feb. 2017.

[6] N. Morozs, T. Clarke, and D. Grace, "Distributed heuristically accelerated Q-learning for robust cognitive spectrum management in LTE cellular systems," *IEEE Trans. Mobile Comput.*, vol. 15, no. 4, pp. 817–825, Apr. 2016.

[7] Y. Wei, F. R. Yu, M. Song, and Z. Han, "Joint optimization of caching, computing, and radio resources for fog-enabled IoT using natural actor-critic deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2061–2073, Apr. 2019.

[8] Y. Cui, W. He, C. Ni, C. Guo, and Z. Liu, "Energy-efficient resource allocation for cache-assisted mobile edge computing," in *Proc. IEEE 42nd Conf. Local Computer Networks (LCN)*, Singapore, Oct. 9–12, 2017, pp. 640–648.

[9] Z. Liu, M. Dong, B. Gu, C. Zhang, Y. Ji, and Y. Tanaka, "Fast-start video delivery in future internet architectures with intra-domain caching," *Mobile Netw. Appl.*, vol. 22, no. 1, pp. 98–112, Feb. 2017.

[10] C. Wu, Z. Liu, D. Zhang, T. Yoshinaga, and Y. Ji, "Spatial intelligence toward trustworthy vehicular IoT," *IEEE Commun. Mag.*, vol. 56, no. 10, pp. 22–27, Oct. 2018.

[11] P. Fazio, F. D. Rango, and M. Tropea, "Prediction and QoS enhancement in new generation cellular networks with mobile hosts: A survey on different protocols and conventional/unconventional approaches," *IEEE Commun. Surv. Tuts.*, vol. 19, no. 3, pp. 1822–1841, Mar. 2017.

[12] R. K. P. Mok, R. K. C. Chang, and W. Li, "Detecting low-quality workers in QoE crowdtesting: A worker behavior-based approach," *IEEE Trans. Multimedia*, vol. 19, no. 3, pp. 530–543, Mar. 2017.

[13] S. Tasaka, "Bayesian hierarchical regression models for QoE estimation and prediction in audiovisual communications," *IEEE Trans. Multimedia*, vol. 19, no. 6, pp. 1195–1208, June 2017.

[14] S. Mu, Z. Zhong, D. Zhao, and M. Ni, "Joint job partitioning and collaborative computation offloading for internet of things," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1046–1059, Feb. 2019.

[15] Z. Pang, L. Sun, Z. Wang, W. Hu, and S. Yang, "CP-operated dash caching via reinforcement learning," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME)*, Hong Kong, July 10–14, 2017, pp. 487–492.

[16] J. Lee and S. Kim, "Filter data cache: An energy-efficient small L0 data cache architecture driven by miss cost reduction," *IEEE Trans. Comput.*, vol. 64, no. 7, pp. 1927–1939, July 2015.

[17] Q. Jia, R. Xie, T. Huang, J. Liu, and Y. Liu, "Energy-efficient cooperative coded caching for heterogeneous small cell networks," in *Proc. IEEE Conf. Computer Communications Workshops (INFOCOM WKSHPS)*, Atlanta, GA, May 1–4, 2017, pp. 468–473.

[18] C. Mueller, S. Lederer, R. Grandl, and C. Timmerer, "Oscillation compensating dynamic adaptive streaming over HTTP," in *Proc. IEEE Int. Conf. Multimedia and Expo (ICME)*, Turin, June 29–July 3, 2015, pp. 1–6.

[19] H. Zhao, Q. Zheng, W. Zhang, B. Du, and H. Li, "A segment-based storage and transcoding trade-off strategy for multi-version VoD systems in the cloud," *IEEE Trans. Multimedia*, vol. 19, no. 1, pp. 149–159, Jan. 2017.

[20] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-Q-learning-based transmission scheduling mechanism for the cognitive internet of things," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2375–2385, Aug. 2018.

[21] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, S. U. Khan, and P. Li, "A double deep Q-learning model for energy-efficient edge scheduling," *IEEE Trans. Services Comput.* doi: 10.1109/TSC.2018.2867482.

[22] A. Anand and N. B. Mehta, "Quick, decentralized, energy-efficient one-shot max function computation using timer-based selection," *IEEE Trans. Commun.*, vol. 63, no. 3, pp. 927–937, Mar. 2015.

[23] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. Int. Conf. Learning Representations (ICLR)*, San Juan, May 2–4, 2016, pp. 1–21.

[24] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. 33rd Int. Conf. Machine Learning (ICML)*, New York, NY, June 19–24, 2016, pp. 1928–1937.

[25] M. J. Marcus, "Unlicensed cognitive sharing of TV spectrum: The controversy at the federal communications commission," *IEEE Commun. Mag.*, vol. 43, no. 5, pp. 24–25, May 2005.