Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 2561069, 12 pages
https://doi.org/10.1155/2019/2561069

Research Article

Edge Caching for D2D Enabled Hierarchical Wireless Networks with Deep Reinforcement Learning

Wenkai Li,[1] Chenyang Wang,[1] Ding Li,[1] Bin Hu,[2] Xiaofei Wang,[1] and Jianji Ren[3]

[1] *College of Intelligence and Computing, Tianjin University, 300350 Tianjin, China*
[2] *Department of Network Engineering, Technical College for Deaf, Tianjin University of Technology, 300000 Tianjin, China*
[3] *Institute of Computer Science and Technology, Henan Polytechnic University, 454150 Jiaozuo, Henan, China*

Correspondence should be addressed to Xiaofei Wang; xiaofeiwang@tju.edu.cn

Received 15 November 2018; Revised 16 January 2019; Accepted 10 February 2019; Published 27 February 2019

Guest Editor: Yubin Zhao

Edge caching is a promising method to deal with the traffic explosion problem towards future network. In order to satisfy the demands of user requests, the contents can be proactively cached locally at the proximity to users (e.g., base stations or user device). Recently, some learning-based edge caching optimizations are discussed. However, most of the previous studies explore the influence of dynamic and constant expanding action and caching space, leading to unpracticality and low efficiency. In this paper, we study the edge caching optimization problem by utilizing the Double Deep Q-network (Double DQN) learning framework to maximize the hit rate of user requests. Firstly, we obtain the Device-to-Device (D2D) sharing model by considering both online and offline factors and then we formulate the optimization problem, which is proved as NP-hard. Then the edge caching replacement problem is derived by Markov decision process (MDP). Finally, an edge caching strategy based on Double DQN is proposed. The experimental results based on large-scale actual traces show the effectiveness of the proposed framework.

1. Introduction

With the development of network services and the sharp increasing of mobile devices, severe traffic pressure posed an urgent demand of network operator to explore the effective paradigm towards 5G. Related works show that the requests of top 10% video account for 80% of all traffic, that is, the repeated downloads of the same content [1]. Device-to-Device (D2D) content sharing is an effective method to reduce mobile network traffic. In this way, users can download required content from nearby devices and enjoy data services with low access latency [2], which can improve their service qualities (QoS).

In order to design an efficient caching strategy in mobile networks, we need to achieve the statistical information of the user requests and sharing activities by system learning from the extreme volume of mobile traffic. In previous work, some important factors in mobile networks (such as content popularity, mobile models, user preferences, and user behaviour) are assumed to be well known, which is not rigorous [3]. Recently, a learning-based method is proposed to jointly optimize the mobile content sharing and caching [4, 5]. The authors of [6] calculated the minimum unload loss according to user's request interval and explored content caching of small base station (SBSs). Srinivasan et al. [7] used the Q-learning method to determine the load-based spectrum, optimizing the spectral sharing. However, traditional RL technology is not feasible for the mobile network environment with large state space.

Motivated by this, we studied the D2D edge caching strategy in hierarchical wireless network in order to maximize unloading traffic and reduce pressure through D2D communication. And the cache replacement process is modelled by Markov decision process (MDP). Finally, a Double Deep Q-network (Double DQN) based edge caching strategy is proposed. The contributions of this paper are summarized as follows:

(i) We model the D2D sharing activities by considering both online factor (users' social behaviours) and offline factor (user mobility). The optimization then is proved as NP-hard.

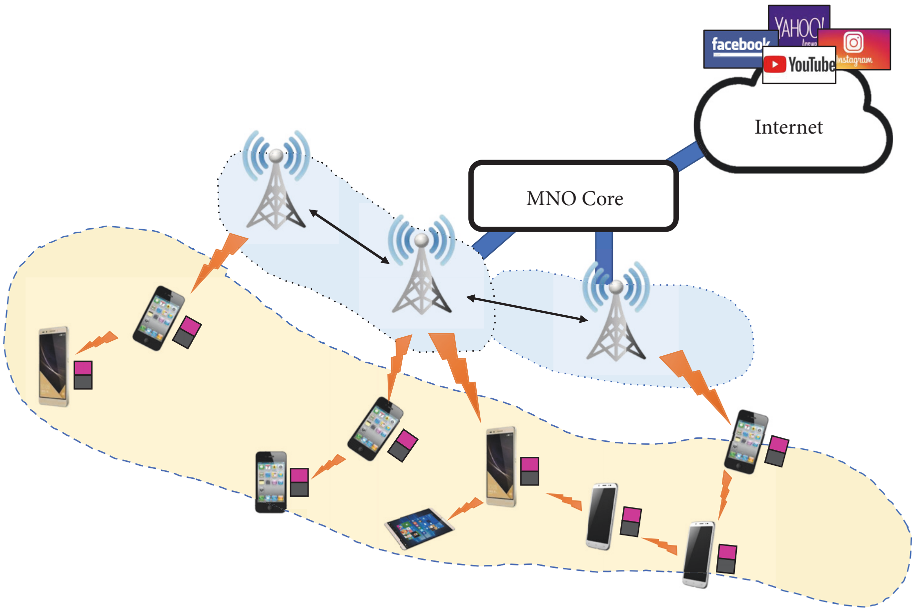


Figure 1: Illustration of edge caching architecture in D2D networks.

(ii) The cache replacement problem is established by Markov decision process (MDP) to address the continuousness of edge caching problem. And we propose a Double DQN-based edge caching strategy to deal with the challenge of action/state spaces explosion.

(iii) Combined with the theoretical model, real trace evaluation, and simulation experimental platform, the proposed Double DQN-based edge caching strategy achieves better performance than some existing caching algorithms, including the least recently used (LRU), least frequently used (LFU), and first-in first-out (FIFO).

The rest of this article is organized as follows. We explained the relevant work in the second part. The third part introduces the system model. The fourth part introduces the cache optimization strategy and raises the relevant problem. The fifth part introduces the details of cache strategy optimization. And in the sixth part, large-scale experiments based on real tracking are carried out.

2. Related Work

There are many researches on edge caching in mobile network. For example, it is studied and proposed in [8–10] that adding caching to mobile network is very promising. Femto caching proposed in [11, 12] and AMVS-NDN proposed by [13] are both committed to adding the cache in BS for the purpose of unloading the traffic. The authors of [14–16] proposed a collaborative caching strategy between BS, which greatly improves the QoS of users. In recent years, the application of intelligence in wireless networks is getting more and more attention. Research in [17, 18] shows that enhanced learning (RL) has great potential in the design of BSs content caching schemes. Particularly, the author proposed the base station caching replacement strategy based on Q-learning and used multiarmed bandit (MAB) to place the cache through RL technology [17]. However, considering the extreme complexity of the actual network environment and the maximum of the state space, traditional RL technology is not feasible. Besides, all of the works mentioned above are focused on single-level caching without considering multilevel caching.

Multitier caching is widely used to exploit the potential of system infrastructure, especially in web caching systems [19–21] and IPTV systems [22]. Reference [23] focused on the theoretical performance analysis of the content cache in HetNets, which assumes that the content is in the same size. However, [22, 23] do not involve the design of caching policies, which required practical considerations in terms of constraints (for instance, limited front-end/backhaul capacity, diversity of content sizes) and specific characteristics of network topologies.

3. System Model

As shown in Figure 1, we consider hierarchical network architecture. The core network communicates with $\mathcal{N}$ base stations via the backhaul link and the base station communicates with the user via the cellular link. N mobile users are uniformly distributed $U = \{u_1, u_2, \ldots, u_N\}$ with a local buffer size $\mathscr{L}^u = \{l_1^u, l_2^u, \ldots, l_N^u\}$, users can establish direct communications with each other via D2D links, and they can also be served by the BSs via cellular links. M files are stored in the content library $\mathscr{F} = \{f_1, f_2, \ldots, f_M\}$, and their content sizes are denoted as $\mathscr{L}_f = \{l_1^f, l_2^f, \ldots, l_M^f\}$. l_f represents the size of the requested content f. The cache state is described by s_{uf}^c. Here, s_{uf}^c is binary, where $s_{uf}^c = 1$ denotes that the user u caches the content f while $s_{uf}^{c}= 0$ means no caching.

3.1. Content Popularity and User Preference. The popularity of content is often described as the probability of a content from the library $\mathscr{F}$ which is requested by all the users. Denote an $N \times M$ popularity matrix $\mathscr{P}$, where $q_{uf} = \mathscr{P}(q_{nm})$ is the probability of user u_n requests for content f_m in the $(n,m)^{th}$ component. In related studies, the content popularity is always described by ZipF distribution as [24].

$$q_{uf} = \frac{R_{uf}^{-\beta}}{\sum_{i\in\mathscr{F}} R_{i}^{-\beta}} \tag{1}$$

where the $R_{uf}^{-\beta}$ is popularity index that user u gives to content f in a descending order and $\beta \geq 0$ is the ZipF exponent.

We measured users' sharing activities by large-scale tracing of D2D sharing based on *Xender*. As shown in Figure 3 [25], in the real world, the matrix $\mathscr{P}$ changes over time (we will introduce the tracking in detail in the sixth part). We assume that the matrix remains constant over time, and our caching strategy refreshes with changes of the popularity matrix $\mathscr{P}$. And the period of user sharing activities can be divided into Peak hours and Peak-off hours. The cache replacement action occurs during the Peak-off hours at each period.

User preference: the user preference, which is denoted as $\mathscr{P}_{uf}$, is the probability distribution of a user's request for each content. According to the content popularity matrix $\mathscr{P}$, each row of the matrix denotes a popularity vector of a user which reflects the preference of a user for a certain content in a statistical way. Assuming that the content popularity and user preference are stochastic, we can obtain the relation:

$$\mathscr{P}_{uf} = \sum_{u=1}^{N} w_u q_{uf} \tag{2}$$

where w_u is the probability of user $u \in U$ sending a request for various contents $f \in \mathscr{F}$, given to a user request distribution $W = [w_1, w_2, \ldots, w_N]$, $\sum_{u=1}^{N} w_u = 1$, $w_u \in [0,1]$, which reflects the request active level of each user.

3.2. D2D Sharing Model. Under the D2D-aid cellular networks, users can select either D2D links model or cellular links model. In the D2D links model, users can request and receive the content from the others via D2D links (e.g., Wi-Fi or Bluetooth) or request the content from the BSs directly in a cellular links manner. In our model, the users select D2D links model in advance. If the requested content is not in their own buffers (or their neighbours'), the cellular links model is chosen.

To model the D2D sharing activities among mobile users, the opportunistic encounter (e.g., user mobility, meeting probability, and geographical distance) and social relationship (e.g., online relations and user preference) are two important factors to be concerned about.

(1) Opportunistic Encounter. It is necessary to ensure that the distance between the two users is less than the critical value d_c,when the user communicates via the D2D link. Since the devices are carried by humans or vehicles, we use the meeting probability to describe the user mobility.

Similar with the prior work [26], we regard λ_{uv} as the contact rate of user u and v, which follows the Poisson distribution and the contact event is independent of the user preference. We can obtain the opportunistic delivery as the Poisson process with rate $\mathscr{P}_{uf}\lambda_{uv}$. If user u caches content f in its buffer, we can derive the probability p_{uv} that user v receives content f from user u before the content expires at time T_f. For a node pair, we can derive that

$$p_{uv} = \int_{T_f}^{\infty} \mathscr{P}_{uf}\lambda_{uv} e^{-\mathscr{P}_{uf}\lambda_{uv} y} dy = 1 - e^{-\mathscr{P}_{uf}\lambda_{uv}T_f} \tag{3}$$

However, if the content f is not cached in user u, $p_{uv} = 0$. Combined with the definition of s_{uf}^{c}, we can overwrite (3), as $p_{uv} = 1 - e^{-\mathscr{P}_{uf}\lambda_{uf}T_f s_{uf}^{c}}$. Hence, the probability that user v cannot receive content f from all the other user $u \in U$ is $\prod_{u\in U}(1-p_{uv})$. Then the probability of user v receiving content f from user u can be expressed by

$$P_{uv} = 1 - \prod_{u\in U} \left(1 - p_{uv}\right) = 1 - e^{-\mathscr{P}_{uf}T_f \sum_{u\in U} \lambda_{uf} s_{uf}^{c}} \tag{4}$$

(2) Social Relationship. In social relationship among users, mobile users with weak social relationship may not be willing to share the content with the others owing to the security/privacy concerns. On the other hand, users sometimes have additional resource and are willing to share the content with others. However, the sharing activities may fail because of the hardware/bandwidth restriction (the content may be too large or the traffic speed is too slow). Thus, we consider the social relationship mainly depends on user preference and content transmission rate condition.

We employ the notion of Cosine Similarity to measure the user preference between two users and the preference similarity factor C_{uv} is defined as

$$C_{uv} = \frac{\sum_{f\in\mathscr{F}} q_{uf}q_{vf}}{\sqrt{\sum_{f\in\mathscr{F}} \left(q_{uf}\right)^2} \sqrt{\sum_{f\in\mathscr{F}} \left(q_{vf}\right)^2}}, \quad \forall u, v \in U \tag{5}$$

Finally, based on the opportunistic encounter and social relationship, we can obtain the probability of D2D sharing between user u and v as follows:

$$P_{uv}^{D2D} = C_{uv} \cdot P_{uv},, \quad \forall u, v \in U,, \forall f \in \mathscr{F} \tag{6}$$

where $\sum_{v\in U} P_{uv}^{D2D} \leq 1, \forall u \in U$. The sum of probability of D2D sharing between each user and other users is less than 1.

3.3. Association of Users and BSs. Users can ask the content directly from the associated local BS when the requested content cannot be satisfied by D2D sharing. Definition P_{u}^{BS} is the cellular serving ratio, which is the average probability that the requests of user u have to be served by local BS via backhaul link rather than D2D communications. Thus, we can obtain $P_{u}^{BS} = 1 - \sum_{v\in U} P_{uv}^{D2D}, \forall u \in U$. In this paper, we consider that the content transmission process can be finished within the user mobility tolerant time, e.g., before the user moves out of the communication range of the local BS. The requested content can be satisfied from the buffer of local BS or obtained from the neighbour BSs via BS-BS link as well as the Internet via backhaul link. Let P_{uB}^{BS} denote the probability of BS n serving user u, then we have

$$P_{uB}^{BS} = \frac{\sum_i T_{un}^{BS}(i)}{\sum_{n\in\mathscr{N}} \sum_i T_{un}^{BS}(i)} \tag{7}$$

where $T_{un}^{BS}(i)$ denote the time period of the i-th cellular serving from BS n to user u during the total sample time T_{tot}. Therefore, we have the probability P_{uv}^{BS} that user u is served by BS n as follows:

$$P_{uB_n}^{BS} = P_u^{BS} \cdot P_{uB}^{BS}, \quad \forall u \in U, \ \forall n \in \mathscr{N} \tag{8}$$

Note that $\sum_{n\in\mathscr{N}} P_{un}^{BS} + \sum_{u\in U} P_{uv}^{D2D} = 1, \forall u \in U$.

3.4. Communication Model. We model the wireless transmission delay between the User and the BS as the ratio between the content size and the downlink data rate. Similar to [27], the downlink data rate from BS n to User u can be expressed as

$$r_{u,n} = w \log_2 \left(1 + \frac{q_u g_{u,n}}{\sigma^2 + \sum_{v\in U\backslash\{u\}} q_v g_{v,n}} \right) \tag{9}$$

where w is channel bandwidth, σ^2represents the background noise power, q_u is transmission power of BS n to User u, and $g_{u,n}$ is the channel gain and is determined by the distance between the User u and the BS n.

3.5. Optimization for D2D-Enabled Edge Caching Problem. Mobile users can share the content via D2D communications. User pair u and v can get the requested content f from u if v has the content (e.g.,$s_{uf}^c = 1$) while v does not under the probability of P_{uv}^{D2D}. Thus, the content offload from the BSs or Internet via D2D link between u and v can be obtained as $l_f P_{uv}^{D2D}$. Whether the user u has the content $f$or not, we can obtain the total content O_{D2D} via D2D sharing as

$$O_{D2D} = \sum_{f\in\mathscr{F}} l_f \sum_{u\in U} P_{uv}^{D2D} s_{uf}^c \left(1 - s_{vf}^c\right) \tag{10}$$

Our aiming is to maximize the total size of content offload at users via D2D sharing while satisfying all the buffer size constraints of mobile users. Formally, the optimization problem is defined as

$$\begin{aligned} \max \quad & O_{D2D} \\ s.t. \quad & \sum_{f\in\mathscr{F}} s_{uf}^c l_f \le L_u, \quad \forall u \in \mathscr{U} \\ & s_{uf}^c \in \{0,1\}, \quad \forall u \in \mathscr{U}, \ \forall f \in \mathscr{F} \end{aligned} \tag{11}$$

where $\sum_{f\in F} s_{uf}^c l_f \le L_u$ is the buffer size constraint of all the mobile users' devices and $s_{uf}^c \in \{0,1\}$ is the caching state in each mobile device.

The optimization problem (11) is NP-hard.

Proof. Let $e_{uv,f} = s_{uf}^c(1 - s_{vf}^c)$, and $e_{uv,f} \in \{0,1\}$.

Thus, we can rewrite Problem (11) as

$$\begin{aligned} \max \quad & \sum_{f\in\mathscr{F}} l_f \sum_{u\in\mathscr{U}} P_{uv}^{D2D} s_{uf}^c \left(1 - s_{vf}^c\right) \\ s.t. \quad & \sum_{f\in\mathscr{F}} s_{uf}^c l_f \le L_u, \quad \forall u \in \mathscr{U} \\ & e_{vu,f}, s_{uf}^c \in \{0,1\}, \ \forall u, v \in \mathscr{U}, \quad \forall f \in \mathscr{F} \end{aligned} \tag{12}$$

where $\sum_{f\in\mathscr{F}} s_{uf}^c l_f \le L_u$ is the cardinality constant of L_u. It is easy to observe that Problem (11) has the same structure with the problem formulated in [28], which has been proved as NP-hard. □

3.6. Cache Replacement Model. We model the cache replacement process as an MDP. Besides, we discuss the details of the related state space, action space, and reward function as follows.

(1) *State Space.* We define s_{uf}^{ci} as the content caching state during each decision epoch i with respect to the content $f \in \mathscr{F}$, which independently picks a value from a state space $\mathbb{P}$. $s_{uf}^{ci} = 1$ means content f is cached in the user u and $s_{uf}^{ci} = 0$ means the opposite. In addition, s_v^{ri} is introduced to denote the current requesting content from other users v in the decision epoch i. The state of an available user during each decision epoch i can be represented by

$$\varkappa_i = \left(s_v^{ri}, s_u^{ci}\right) \in \mathscr{X} \overset{\text{def}}{=} \{1, 2, \dots F\} \times \left\{\times_{f\in\mathscr{F}} \mathbb{P}\right\} \tag{13}$$

(2) *Action Space.* The system action with respect to the state $\varkappa_i$ can be denoted as $\mathscr{A}(\varkappa_i)$. All users possess the same action space $\mathbb{A}$ as

$$\mathbb{A} = \left\{a_i^{D2D}, a_i^{BS}\right\} \tag{14}$$

Namely, the system action $\mathscr{A}(\varkappa_i)$ can be divided into two parts according to their different characters as follows.

(a) *Requests Handled via D2D link.* The available cache control in the adjacent users is represented by $a_i^{D2D} \overset{\text{def}}{=} [a_{i,0}^{D2D}, a_{i,1}^{D2D}, \dots, a_{i,F}^{D2D}]$, where $a_{i,f}^{D2D} \in \{0,1\}(f \in 1, \dots, F)$ indicates that whether and which content in the local user should be replaced by the current requesting content, ($a_{i,0}^{D2D} \in 0,1$) represents whether the local user makes replacement; i.e., the content request is handled by the user itself.

(b) *Requests Handled by BSs.* Certainly, each user can get content directly from BSs when the D2D link fails to meet the requirements. $a_i^{SP} \in \{0,1\}$ is introduced to represent this kind of action, where $a_i^{BS} = 1$ means that the request is chosen to be directly handled by BSs, namely, the User shall fetch the content from BSs.

(3) *Reward Function.* Reward (utility) function $\mathscr{R}(\varkappa, \mathscr{A})$, which determines the reward fed back to the user when performing the action $\mathscr{A}(\varkappa_i)$ upon the state $\varkappa_i$, shall be determined in the interactive wireless environment to lead the DRL agent (we will introduce it later) in users towards achieving ideal performance. Among the QoS metrics, the most important is to improve the hit rate of user-requested

content. Our goal is to maximize the hit rate of user requests. Therefore, in our edge caching architecture, we design the reward function as

$$\mathscr{R}\left(\varkappa_i, \mathscr{A}\left(\varkappa_i\right)\right) = \begin{cases} e^{l_f}, & \mathscr{A}\left(\varkappa_i\right) = a_i^{D2D} \\ e^{-(l_f)}, & \mathscr{A}\left(\varkappa_i\right) = a_i^{BS} \end{cases} \tag{15}$$

where exponential function with respect to the traffic is adopted here to guide the objective of maximizing the traffic.

4. Edge Caching Policy Discussion

In the hierarchical wireless networks with cache-enabled D2D communications, we explore the maximum capacity of the network based on the mobility and social behaviours of users. The goal is to optimize the network edge caching by offloading the contents to users via D2D communications and reducing the system cost of content exchange between BSs and core network via cellular links.

4.1. Problem Formulation. Based on the above analysis and combined with (15), the optimization objective is defined as

$$\begin{aligned} &R^{\text{long}} \\ &\quad = \max E_{\mathscr{A}}\left[\lim_{I \longrightarrow \infty} \frac{1}{I} \sum_{i=1}^{I} \mathscr{R}\left(\varkappa_i, \mathscr{A}\left(\ \varkappa_i\right)\right) \mid \varkappa_1 = \varkappa\right] \end{aligned} \tag{16}$$

which indicates maximizing the expected long-term reward value conditioned on any initial state $\varkappa_1$.

Nevertheless, in general, a single-agent infinite-horizon MDP with the discounted utility (17) can be used to approximate the expected infinite-horizon undiscounted value when $\gamma\epsilon[0,1)$ approaches 1.

$$V\left(\varkappa, \mathscr{A}\right) = E_{\mathscr{A}}\left[\sum_{i=1}^{\infty} (\gamma)^{i\text{-}1} \cdot \mathscr{R}\left(\varkappa_i, \mathscr{A}\left(\varkappa_i\right)\right) \mid \varkappa_1 = \varkappa\right] \tag{17}$$

Further, we can obtain the optimal state value function V($\varkappa$) for any initial state χ as

$$V\left(\varkappa\right) = V\left(\varkappa, \mathscr{A}^*\right), \quad \forall \varkappa \in \mathscr{X} \tag{18}$$

In conclusion, each user is expected to learn an optimal control policy $\mathscr{A}^*$ that maximizes V($\varkappa, \mathscr{A}$), with any initial state $\varkappa$. The optimal control policy can be described as follows:

$$\mathscr{A}^* = \underset{\mathscr{A}}{\operatorname{argmax}}\, V\left(\varkappa, \mathscr{A}\right), \quad \forall \varkappa \in \mathscr{X} \tag{19}$$

5. Double DQN-Based Edge Cache Strategy

5.1. Reinforcement Learning. Reinforcement learning is a machine learning algorithm. In other words, it is a way for an agent to keep trying, to learn from mistakes, and finally to find patterns. RL problems can be described as the optimal control decision making problem in MDP. RL contains many forms, among which Q-learning algorithm based on tabular learning is commonly used. Q-learning is an off-policy learning algorithm that allows an agent to learn through current or past experiences.

In our D2D caching architecture, the agent pertains to the user senses and obtains its current cache state $\varkappa_i$. Then, the agent selects and carries out an action $\mathscr{A}(\varkappa_i)$. Meantime, the environment experiences a transition from $\varkappa_i$ to a new state $\varkappa_{i+1}$ and obtains a reward $\mathscr{R}(\varkappa_i, \mathscr{A}(\varkappa_i))$.

According to the Bellman Equation, the optimal Q-value function $Q(\varkappa, \mathscr{A})$ can be expressed as (20), where $\varkappa = \varkappa_i$ is the state at current decision epoch i, and the next state is $\varkappa' = \varkappa_{i+1}$ after taking the action $\mathscr{A} = \mathscr{A}(\varkappa_i)$.

$$\begin{aligned} Q\left(\varkappa, \mathscr{A}\right) = \mathscr{R}\left(\varkappa, \mathscr{A}\right) + \gamma \cdot \sum_{\varkappa'} Pr\left\{\varkappa' \mid \varkappa, \mathscr{A}\right\} \\ \cdot \max_{\mathscr{A}'} Q\left(\varkappa', \mathscr{A}'\right) \end{aligned} \tag{20}$$

The iterative formula of Q-function can be obtained as

$$\begin{aligned} &Q^{i+1}\left(\varkappa, \mathscr{A}\right) \\ &\quad = Q^{i}\left(\varkappa, \mathscr{A}\right) + \alpha^{i} \\ &\qquad \cdot \left(\mathscr{R}\left(\varkappa, \mathscr{A}\right) + \gamma \cdot \max_{\mathscr{A}'} Q^{i}\left(\varkappa', \mathscr{A}'\right) - Q^{i}\left(\varkappa, \mathscr{A}\right)\right) \end{aligned} \tag{21}$$

where $\alpha^i \epsilon[0,1)$ is the learning rate and the state $\varkappa_i$ will turn to the state $\varkappa_{i+1}$ when the agent chooses action $\mathscr{A}(\varkappa_i)$ along with the corresponding reward $\mathscr{R}(\varkappa_i, \mathscr{A}(\varkappa_i))$. Based on (21), the Q-Table can be used to store the Q value of each state-action pair when the state and action space dimensions are not high in the Q-Learning algorithm. We conclude the training algorithm based on the Q-Learning in Algorithm 1. The complexity of the Q-learning algorithm depends primarily on the scale of the problem. Updating the Q value in a given state requires determining the maximum Q value for all possible actions in the corresponding state in the table. In a given state, if there are n possible actions, finding the maximum Q value requires $n-1$ comparisons. In other words, if there are n states, the update of the entire Q-table requires $m(n-1)$ comparison. Hence, the learning process in Q-Learning becomes extremely difficult when the scenarios are with huge network states and action spaces. Therefore, using neural network to generate Q value becomes a potential solution.

5.2. Double Deep Q-Learning. DQN is the first model that successfully combines Deep Learning with Reinforcement Learning. It replaced the Q-table with the neural network, which effectively solved the complicated and high dimensional RL problems. It comes in many variations, the most famous of which is Double DQN [29]. In our model, we use Double DQN to train our DRL agents in users, which is formed as shown in Figure 2. The Q-function could be approximated to the optimal Q value by updating the parameter τ_i of neural network as follows:

$$Q\left(\varkappa, \mathscr{A}\right) \approx Q\left(\left(\varkappa, \mathscr{A}\right); \tau_i\right) \tag{22}$$

Experience replay is the core component of DQN. It actually is a memory for storing transitions with a finite size N_m, and its stored procedures are overridden by loops. It can effectively eliminate the correlation between training data. The transition sample can be represented as $T_i =$

```
Initialization: Q-Table
Iteration:
for each episode
    Initialize $\mathscr{s}$
    for each step of episode
        Generate $\mathscr{a}$ at random
        if $\mathscr{a} \leq \varepsilon$
            randomly select an action
        else
            choose $\mathscr{A}(\mathscr{s})$ using policy derived from $Q(\mathscr{s}, \mathscr{A})$
        Take action $\mathscr{A}(\mathscr{s})$
        Obtain $\mathscr{R}(\mathscr{s}, \mathscr{A}(\mathscr{s}))$ and $\mathscr{s}'$
        Update Q-Table: $Q(\mathscr{s}, \mathscr{A}) \longleftarrow Q(\mathscr{s}, \mathscr{A}) + \alpha \cdot (\mathscr{R}(\mathscr{s}, \mathscr{A}) + \gamma \cdot \max_{\Phi'} Q(\mathscr{s}', \mathscr{A}') - Q(\mathscr{s}, \mathscr{A}))$
        $\mathscr{s} \longleftarrow \mathscr{s}'$
    end for
end for
```

Algorithm 1: Q-Learning-based content caching algorithm.

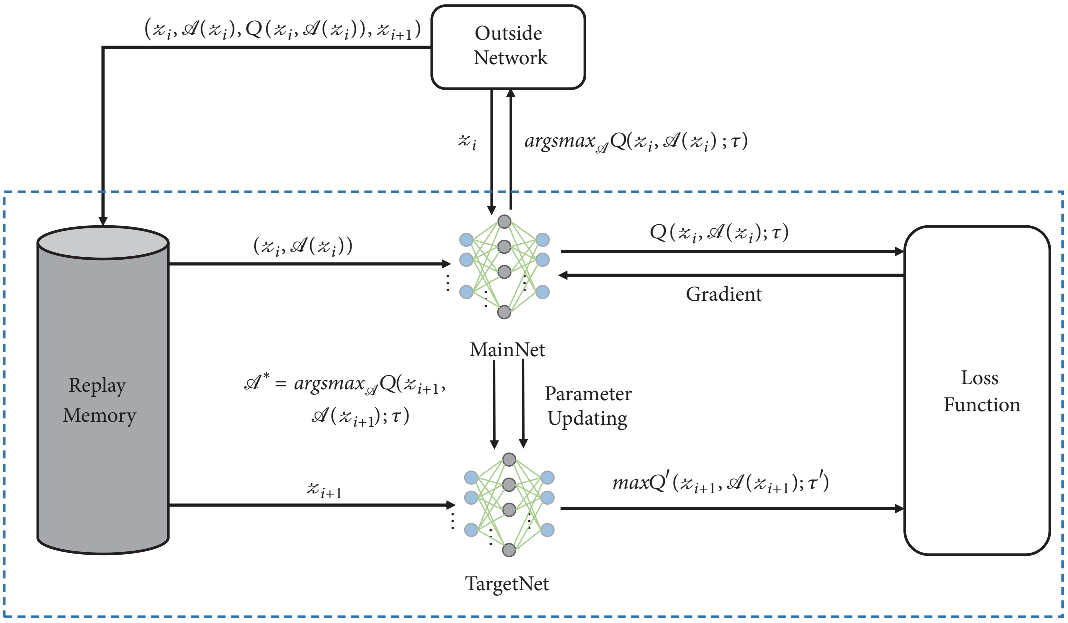

$(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i), Q(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i)), \mathscr{s}_{i+1})$
Outside Network
$\mathscr{s}_i$
$argsmax_{\mathscr{A}} Q(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i); \tau)$
$(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i))$
$Q(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i); \tau)$
Gradient
MainNet
Replay Memory
$\mathscr{A}^* = argsmax_{\mathscr{A}} Q(\mathscr{s}_{i+1}, \mathscr{A}(\mathscr{s}_{i+1}); \tau)$
Parameter Updating
Loss Function
$\mathscr{s}_{i+1}$
$maxQ'(\mathscr{s}_{i+1}, \mathscr{A}(\mathscr{s}_{i+1}); \tau')$
TargetNet

Figure 2: Illustration of training process.

$(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i), R(\mathscr{s}_i, \mathscr{A}(\mathscr{s}_i)), \mathscr{s}_{i+1})$, which represents one state transition. The whole experience pool can be denoted as $\mathscr{M} = \{T_{i-N_m+1}, \ldots, T_i\}$. Note that each DRL agent maintains two Q networks, namely. $Q(\mathscr{s}, \mathscr{A}; \tau_i)$ and $Q'(\mathscr{s}, \mathscr{A}; \tau_i{}')$, with network Q used to choose action and network Q' to evaluate action. Besides, the counterpart τ_i of network Q periodically updates the weight parameters $\tau_i{}'$ of network Q'.

Throughout the training process, the DRL agent randomly samples a minibatch $\mathscr{M}'$ from the experience replay $\mathscr{M}$. Then, at each epoch, the network Q is trained towards the direction of minimizing the loss function as

$$L(\tau_i) = E_{(\mathscr{s}, \mathscr{A}, \mathscr{R}(\mathscr{s}, \mathscr{A}), \mathscr{s}') \in \widetilde{\mathscr{M}}_i} \Bigg[\Bigg(\mathscr{R}(\mathscr{s}, \mathscr{A}) + \gamma \cdot Q' \Bigg(\mathscr{s}, \arg\max_{\mathscr{A}'} Q\left(\mathscr{s}', \mathscr{A}'; \tau_i\right); \tau_i' \Bigg) \Bigg) - Q\left(\mathscr{s}, \mathscr{A}; \tau_i\right) \Bigg)^2 \Bigg] \quad (23)$$

And with (23), the gradient guiding updates of τ can be calculated by $\partial L(\tau_i)/\partial \tau_i$. Hence, Stochastic Gradient Descent (SGD) is performed until the convergence of Q networks for approximating optimal state-action Q-function. We conclude the training algorithm based on the Double DQN in Algorithm 2.

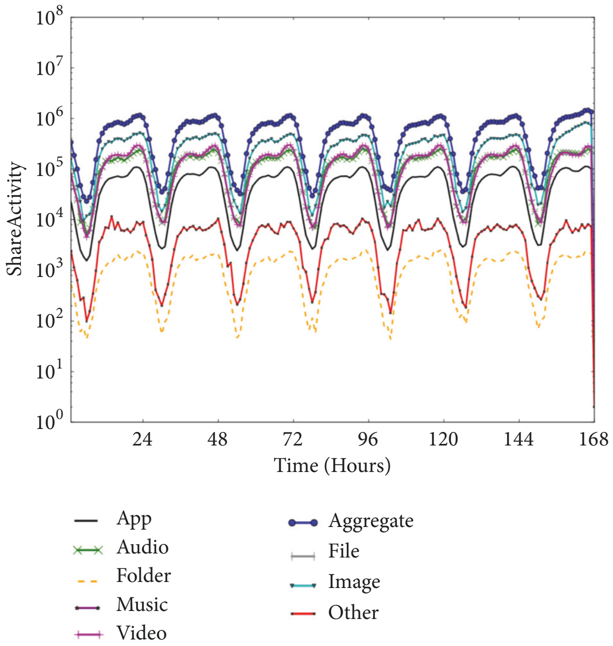


Figure 3: Statistic for all sharing activities [25].

```
Initialization: Experience replay memory $\mathcal{M}$, main Q network with random weights $\boldsymbol{\tau}$, target $Q'$
network with $\boldsymbol{\tau}' = \boldsymbol{\tau}$, and the period of replacing target Q network $\phi$.
Iteration:
for each episode
    Initialize $\varkappa$
    $i \longleftarrow 0$
    for each step of episode
        $i \longleftarrow i + 1$
        Randomly generate $\mathcal{a}$
        if $\mathcal{a} \leq \varepsilon$
            randomly select an action
        else
            $\mathcal{A}(\varkappa) \longleftarrow \arg\max_{\mathcal{A}(\varkappa)} Q(\varkappa, \mathcal{A}(\varkappa); \tau_i)$
        Take action $\mathcal{A}(\varkappa_i)$
        Obtain $\mathcal{R}(\varkappa_i, \mathcal{A}(\varkappa_i))$ and $\varkappa'$.
        Store $T \longleftarrow (\varkappa, \mathcal{A}(\varkappa), \mathcal{R}(\varkappa, \mathcal{A}(\varkappa)), \varkappa')$ into $\mathcal{M}$.
        Randomly sample a mini-batch of transitions $\mathcal{M}' \in \mathcal{M}$.
        Update $\tau_i$ with $\partial L(\tau_i)/\partial \tau_i$.
        if $i == \phi$
            Update $\tau_i{}'$
            $i \longleftarrow 0$
        $\varkappa \longleftarrow \varkappa'$
    end for
end for
```

Algorithm 2: Double DQN-based content caching algorithm.

About algorithm complexity, it mainly includes collecting transitions and executing backpropagation to train the parameters. Since collecting one transition requires $O(1)$ computational complexity, the total computational complexity for collecting K transitions into the replay memory is $O(K)$. Let a and b denote the number of layers and the maximum number of units in each layer, respectively. Training parameters with backpropagation and gradient descent requires the computational complexity of $O(mabi)$ where m and i denote the number of transitions randomly sampled

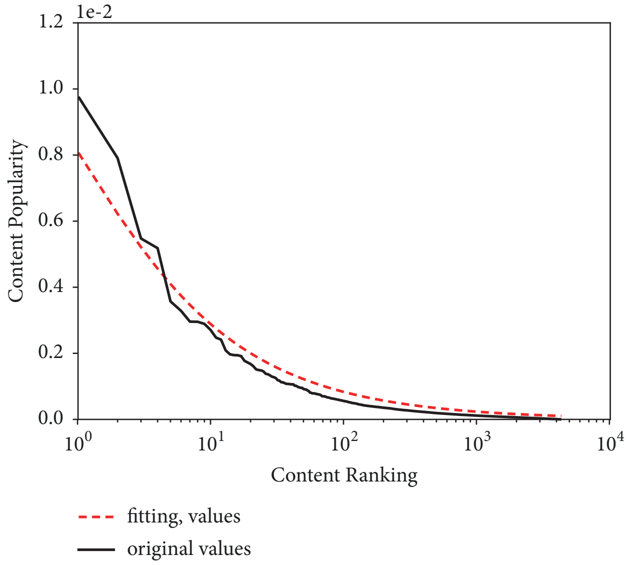


Figure 4: Content popularity.

from the replay memory and the number of iterations, respectively. Furthermore, the replay memory and the parameters of the double deep Q-learning model dominate the storage complexity. Specially, storing K transitions needs the about space complexity of $O(K)$ while the parameters need the about space complexity of $O(ab)$.

6. Experiment

In this section, we evaluate the proposed cache policy based on the experimental results of the mobile application *Xender*.

6.1. DataSet. *Xender* is a mobile APP that can realize offline D2D communication activities. It provides a new way to share diversified content files users are interested in without accessing 3G/4G cellular mobile networks, largely reducing repeated traffic load and waste of network resources, as a result, achieving resource sharing. Currently *Xender* has around 10 million daily and 100 million monthly active users, as well as about 110 million daily content deliveries.

We capture *Xender's* trace for one month (from 01/08/2016 to 31/08/2016), including 450,786 active mobile users, conveying 153,482 content files, and 271,785,952 content requests [30]. As shown in Figure 4, the content popularity distribution in the *Xender's* trace can be fitted by MZipf distribution with a plateau factor of −0.88 and a skewness factor of 0.35.

6.2. Parameter Settings. In our simulations, four BSs are employed with maximum cover range 250 m, $g_{u,n} = 30.6 + 36.7\log_{10} l_{u,n}$ in dB [31] is taken as the channel gain model, and the channel bandwidth of each BS is set as 20 MHz. The delays of D2D link, BS to MNO and MNO to Internet are 5ms, 20ms, and 100ms, respectively. Besides, the total transmit power of BS is 40 W with serving at most 500 Users. With respect to the parameter settings of Double DQN, a single-layer fully connected feed forward neural network, including 200 neurons, is used to serve as the target and the eval Q network. Other parameter values are given in Table 1.

6.3. Evaluation Results. In order to evaluate the performance of our caching strategy, we compared it with three classic cache replacement algorithms.

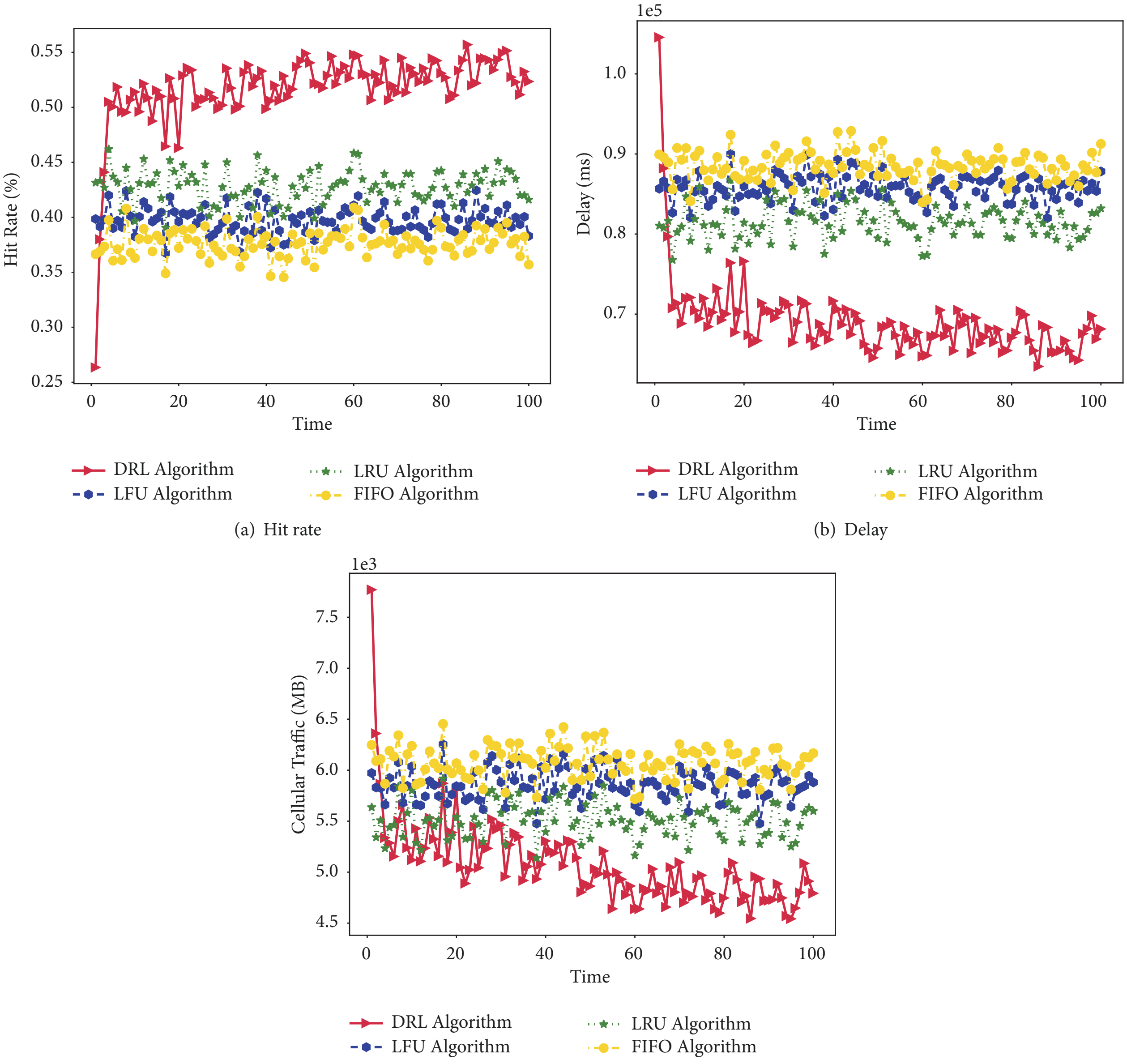
(1) LRU: replace the least recently used content.

(2) LFU: replace the least commonly used content first.

(3) FIFO: replace the first in content first.

Figure 5 shows the performance comparison of cache hit ratio, delay, and traffic at F=1000 and C=100M. As we can see, at the beginning of the simulation, the caching strategy we proposed was surely at a great disadvantage among three aspects. But soon the hit rate increased and stabilized eventually. This is because our reward function is used to increase the cache hit rate; thus, our DRL agent is dedicated to maximizing the system hit rate. It can be seen that our caching strategy is significantly 9%, 12%, and 14% higher than LRU, LFU, and FIFO in terms of hit rate, respectively. At the same time, the improvement of the hit rate has a positive impact on the delay, traffic indicators, and other indexes. The delay of our strategy is 12%, 17%, and 21% lower than that of LRU, LFU, and FIFO, respectively. Besides, the traffic saved is 8%, 10%, and 14%, respectively.

In addition, we explored the effect of content quantity on performance comparison results. We compared the performance when the number of contents is 1000 and 2000. As shown in Figure 6, it can be inferred that when the number of contents increases, the convergence of the algorithm changes and the hit rate decreases. However, it cannot change the overall trend of the algorithm. Our



(a) Hit rate

(b) Delay

(c) Traffic

Figure 5: Performance evaluation in terms of hit rate, delay, and traffic with respect to the time.

Table 1: Parameter Value.

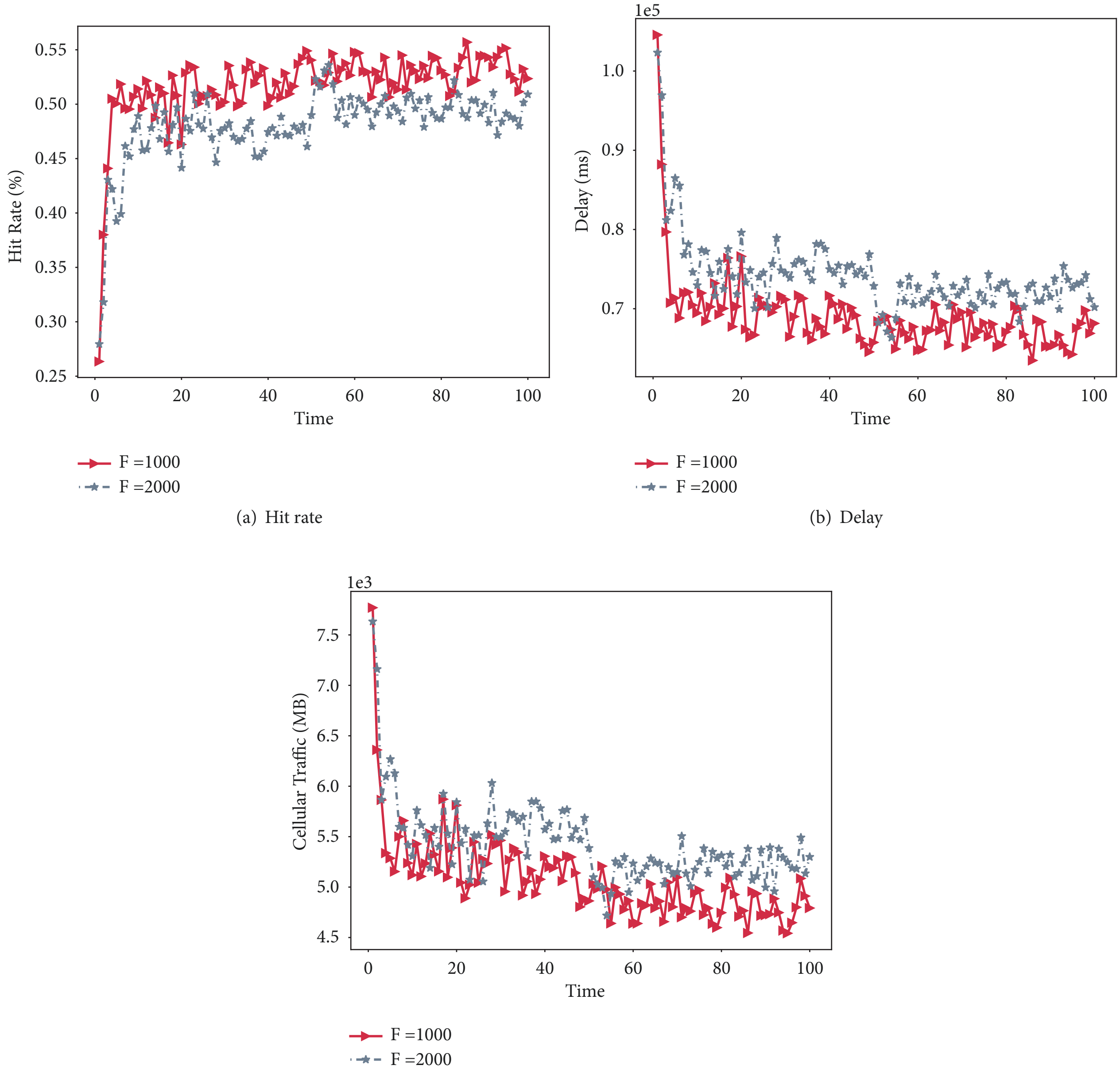
	F	$w[MHz]$	$\sigma^2[dBm]$	$\mathscr{M}$	$\mathscr{M}'$	γ	ϵ	α	ϕ
Range	1000	200	-95	5000	200	0.9	0.1	0.05	250

caching strategy can still perform optimally in these four algorithms.

Finally, we explored the effects of learning rate and exploration probability on our algorithm performances. As shown in Figure 7, learning rate is 0.5 and 0.05 and exploration probability is 0.1 and 0.5, respectively. It can be seen that both of these factors have a great impact on the cache strategy, mainly manifesting in convergence and performance. Thus large numbers of experiments are performed to find an appropriate learning rate and exploration probability for the proposed edge caching scenarios. Hence, in our setting, $\alpha = 0.05$ and $\epsilon = 0.1$ are selected for achieving better performance.

7. Conclusions

In this paper, we study the edge caching strategy of layered wireless networks. Specifically, we use the Markov decision



(a) Hit rate

(b) Delay

(c) Traffic

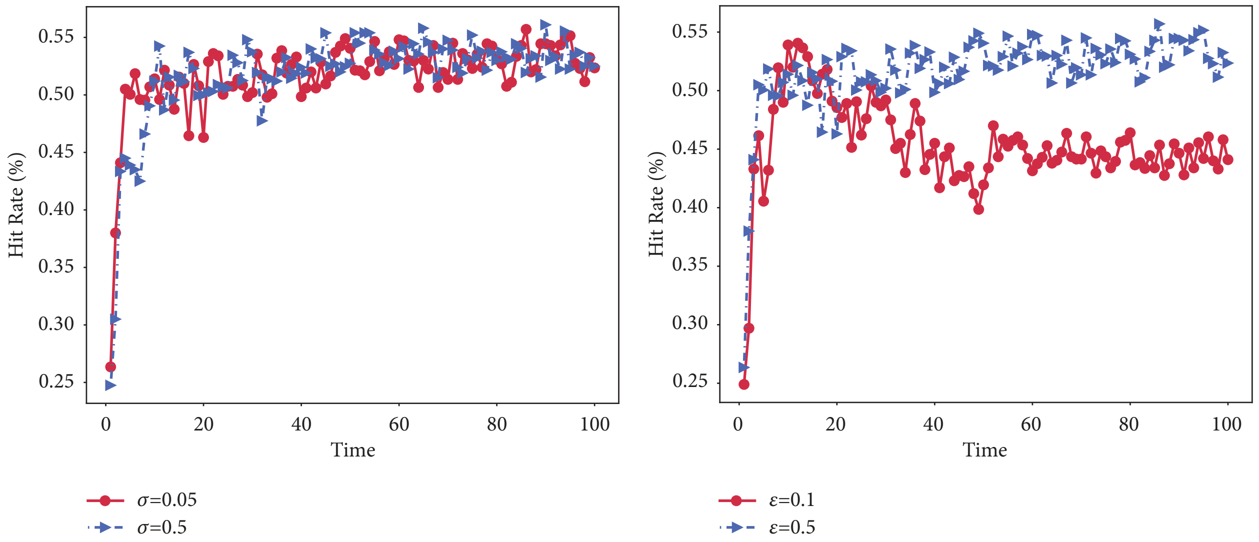
FIGURE 6: Performance comparison between *F=1000, F=2000*.

process and Deep Reinforcement Learning in the proposed edge cache replacement strategy. The experimental results based on actual tracking show that our proposed strategy is superior to LRU, LFU, and FIFO in terms of hit rate, delay and traffic offload. Finally, we also explored the impact of learning rate and exploration probability on algorithm performance.

In the future, we'll focus more on the user layer's impact on cache replacement. (1) In the existing D2D model, the transmission process of files is not persistent, and complex user movement will lead to the interruption of content delivery. In the future, we will consider this factor in the reward function; (2) The cache replacement process requires additional costs, such as latency and energy consumption, all of which should be considered, but how to quantify these factors in the simulation experiment still needs to be explored. (3) The computing resources of user devices are limited. Although Deep Reinforcement Learning can solve the problem of dimensional explosion, it still requires a lot of computing resources. Therefore, we will explore the application of more lightweight learning algorithms in D2D-aid cellular networks.

Data Availability

The data used to support the findings of this study have not been made available because commercial reasons.



(a) Hit rate under $\alpha = 0.05, \alpha = 0.5$

(b) Hit rate under $\epsilon = 0.1, \epsilon = 0.5$

FIGURE 7: Performance of hit rate under different parameters.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The conference version of the manuscript is firstly presented in 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS). Authors have extended the work significantly by exploiting the edge caching problem with deep reinforcement learning framework in this journal version. This work was supported in part by the National Key Research and Development Program of China under grant 2018YFC0809803 and in part by the Natural Science Foundation of China under grant 61702364.

References

[1] X. Wang, M. Chen, Z. Han et al., "TOSS: traffic offloading by social network service-based opportunistic sharing in mobile social networks," in *Proceedings of the INFOCOM*, pp. 2346–2354, 2014.

[2] M. Gregori, J. Gómez-Vilardebó, J. Matamoros, and D. Gunduz, "Wireless content caching for small cell and D2D networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1222–1234, 2016.

[3] T. Rodrigues, F. Benevenuto, M. Cha, K. Gummadi, and V. Almeida, "On word-of-mouth based discovery of the web," in *Proceedings of the 2011 ACM SIGCOMM Internet Measurement Conference, IMC'11*, pp. 381–396, November 2011.

[4] J. Song, M. Sheng, T. Q. Quek, C. Xu, and X. Wang, "Learning based content caching and sharing for wireless networks," *IEEE Transactions on Communications*, vol. 99, pp. 1-1, 2017.

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

[6] B. N. Bharath, K. G. Nagananda, and H. V. Poor, "A learning-based approach to caching in heterogenous small cell networks," *IEEE Transactions on Communications*, vol. 64, no. 4, pp. 1674–1686, 2016.

[7] M. Srinivasan, V. J. Kotagi, and C. S. R. Murthy, "A Q-learning framework for user QoE enhanced self-organizing spectrally efficient network using a novel inter-operator proximal spectrum sharing," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 11, pp. 2887–2901, 2016.

[8] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: exploiting content caching and delivery techniques for 5G systems," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 131–139, 2014.

[9] M. Sheng, C. Xu, J. Liu, J. Song, X. Ma, and J. Li, "Enhancement for content delivery with proximity communications in caching enabled wireless networks: Architecture and challenges," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 70–76, 2016.

[10] E. Zeydan, E. Bastug, M. Bennis et al., "Big data caching for networking: moving from cloud to edge," *IEEE Communications Magazine*, vol. 54, no. 9, pp. 36–42, 2016.

[11] N. Golrezaei, A. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: a new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013.

[12] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: Wireless video content delivery through distributed caching helpers," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 1107–1115, March 2012.

[13] B. Han, X. Wang, N. Choi, T. Kwon, and Y. Choi, "AMVS-NDN: Adaptive mobile video streaming and sharing in wireless named data networking," in *Proceedings of the 2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 375–380, April 2013.

[14] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: wireless content delivery through

distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[15] X. Li, X. Wang, S. Xiao, and V. C. Leung, "Delay performance analysis of cooperative cell caching in future mobile networks," in *Proceedings of the 2015 IEEE International Conference on Signal Processing for Communications (ICC)*, pp. 5652–5657, June 2015.

[16] S. H. Chae, J. Y. Ryu, T. Q. Quek, and W. Choi, "Cooperative transmission via caching helpers," in *Proceedings of the GLOBECOM 2015 - 2015 IEEE Global Communications Conference*, pp. 1–6, San Diego, CA, USA, December 2015.

[17] J. Gu, W. Wang, A. Huang, H. Shan, and Z. Zhang, "Distributed cache replacement for caching-enable base stations in cellular networks," in *Proceedings of the 2014 1st IEEE International Conference on Communications, ICC 2014*, pp. 2648–2653, Australia, June 2014.

[18] C. Wang, S. Wang, D. Li, X. Wang, X. Li, and V. C. Leung, "Q-learning based edge caching optimization for D2D enabled hierarchical wireless networks," in *Proceedings of the 2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 55–63, Chengdu, China, October 2018.

[19] P. Rodriguez, C. Spanner, and E. W. Biersack, "Analysis of web caching architectures: Hierarchical and distributed caching," *IEEE/ACM Transactions on Networking*, vol. 9, no. 4, pp. 404–418, 2001.

[20] H. Che, Y. Tung, and Z. Wang, "Hierarchical web caching systems: modeling, design and experimental results," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 7, pp. 1305–1314, 2002.

[21] K. Poularakis and L. Tassiulas, "On the complexity of optimal content placement in hierarchical caching networks," *IEEE Transactions on Communications*, vol. 64, no. 5, pp. 2092–2103, 2016.

[22] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li, "Collaborative hierarchical caching with dynamic request routing for massive content distribution," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 2444–2452, March 2012.

[23] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: the role of proactive caching in 5G wireless networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, 2014.

[24] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Transactions on Networking*, vol. 16, no. 6, pp. 1447–1460, 2008.

[25] S. Wang, Y. Zhang, H. Wang, Z. Huang, X. Wang, and T. Jiang, "Large scale measurement and analytics on social groups of device-to-device sharing in mobile social networks," *Mobile Networks and Applications*, vol. 23, no. 2, pp. 203–215, 2017.

[26] A. Balasubramanian, B. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proceedings of the ACM SIGCOMM 2007: Conference on Computer Communications*, pp. 373–384, August 2007.

[27] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[28] T. H. Cormen, C. E. Leiserson, R. Rivest et al., *An Introduction to Algorithms*, MIT Press, Cambridge, MA, USA, 2nd edition, 2001.

[29] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with Double Q-learning," in *Proceedings of the AAAI*, pp. 2094–2100, 2016.

[30] X. Li, X. Wang, P. Wan, Z. Han, and V. C. Leung, "Hierarchical edge caching in device-to-device aided mobile networks: modeling, optimization, and design," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1768–1785, 2018.

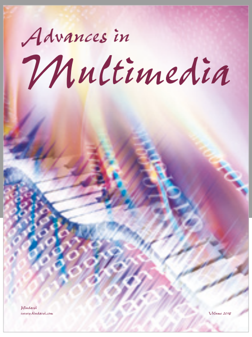
[31] 3GPP, "Further advancements for E-UTRA physical layer aspects (release 9)," Tech. Rep. 36.814 V1.2.0, 2009.

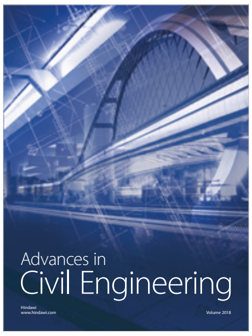
Journal of
Engineering

The Scientific World Journal

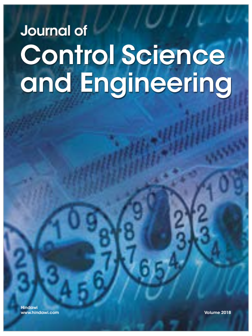
International Journal of
Rotating Machinery

Journal of
Sensors

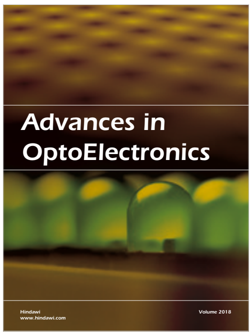

Advances in
Multimedia


Advances in
Civil Engineering

Journal of
Robotics

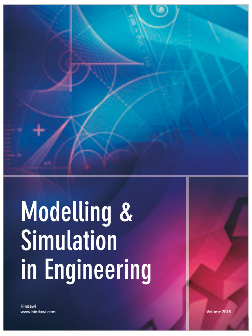

Journal of
Control Science and Engineering

Journal of
Electrical and Computer Engineering

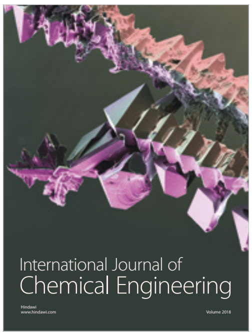

Advances in
OptoElectronics

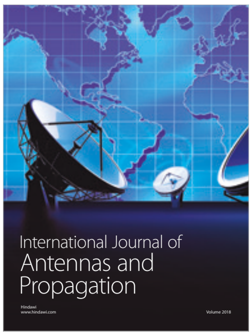
VLSI Design

International Journal of
Navigation and Observation

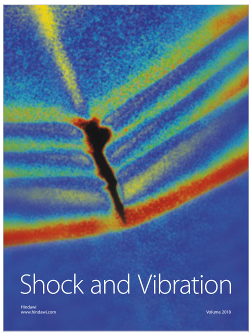

Modelling & Simulation in Engineering

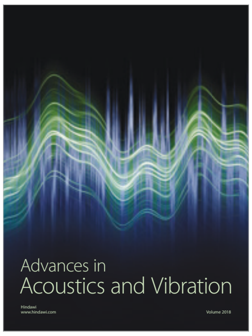
International Journal of
Aerospace Engineering


International Journal of
Chemical Engineering


International Journal of
Antennas and Propagation

Active and Passive
Electronic Components


Shock and Vibration


Advances in
Acoustics and Vibration