Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 4293625, 15 pages
https://doi.org/10.1155/2021/4293625

Research Article

Social-Aware Caching Strategy Based on Joint Action Deep Reinforcement Learning

Jing Yang,[1,2,3] **Zhuowei Song,**[1,2,3] **Peng He**,[1,2,3] **Yaping Cui**,[1,2,3] **Dapeng Wu**,[1,2,3] **and Ruyan Wang**[1,2,3]

[1] *School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China*
[2] *Advanced Network and Intelligent Connection Technology Key Laboratory of Chongqing Education Commission of China, Chongqing, China*
[3] *Chongqing Key Laboratory of Ubiquitous Sensing and Networking, Chongqing, China*

Correspondence should be addressed to Peng He; hepeng@cqupt.edu.cn

Received 19 May 2021; Accepted 7 September 2021; Published 30 September 2021

Academic Editor: Huaqun Wang

Caching in device-to-device (D2D) networks is emerging a promising trend, which enables to reduce backhaul traffic. Moreover, social interaction among users influences the performance of overall system network. Therefore, it is crucial to consider social attributes in the D2D networks to develop a caching strategy to resolve the problem of unbalanced content distributed. In this paper, we consider two types of users according to their activeness, i.e., active users and inactive users. Inactive users assist active users cache contents during off-peak periods and provide the contents to the active users during peak periods to relieve the pressure of base station (BS). In addition, caching system model is divided into physical domain model and social domain model. In physical domain, the quality of communication links is judged by the delay between D2D users. In social domain, based on a real-world dataset, CiaoDVD, we calculate user similarity in three dimensions and obtain user trust by a trust topology to measure user relationships. Finally, in order to maximize the cache hit ratio, a joint action deep *Q*-networks (JADQN) framework is proposed to pair the active users with inactive users and distribute the contents to inactive users. Simulation results indicate that the proposed strategy improves the cache hit ratio by 42.9% and reduces the download delay by 48.8% compared with least frequency used (LFU) algorithm, which validates the effectiveness of our method.

1. Introduction

Recently, the data traffic is exploding with plenty of mobile devices connecting in the wireless network, which brings a heavy burden to the cellular infrastructures. In addition, due to the ever-increasing video applications, the multimedia video services play an important role in our lives and become the majority of Internet traffic [1]. Besides, the unprecedented amount of demand imposes stringent requirements in terms of bandwidth, transmission delay, and communication quality [2]. Caching utilizes the storage capacity of wireless device-assisted traffic offloading [3], which is regard as one of the most promising methods to cope with these problems.

Device-to-device (D2D) communication is recognized as a key technology which can establish communication links without passing through cellular infrastructure [4]. By this way, there are plenty of advantages including high data rates, short delays, and low power consumption [5]. Moreover, D2D communications could extend network capacity, which can increase the cache space of the system. Thus, D2D-assisted caching, which is combined with the advantages of caching and D2D communications, can exploit the cache capacity of user devices during the off-peak periods. In detail, D2D-assisted caching can prefetch contents in the local cache from the adjacent devices during off-peak hours when the network has abundant resources. In this way, the spectrum resources can be efficiently utilized, and the throughput of network can be remarkably improved. In this case, the pressure of base station (BS) can be relieved during the peak periods. However, due to the huge number of contents, the cache resources and the communication resources of users are limited. Moreover, different requests of users enable the unbalanced content distribution and different cache capacity of users. Thus, it is particularly significant to design a caching allocation policy to enhance the cache resources utilization. Some approaches have considered the channel state of the wireless links and the distance of nodes to improve caching performance like the average download delay, throughput, and cache hit ratio. Nevertheless, user relationships influence the quality of communication because the communication link may be broken if the users are not willing to establish the communication links to share their resources.

With the development and popularity of social applications such as Wechat, YouTube, and Facebook, the social relationships among people are extensively broadened and obviously enhanced [6]. In addition, the social interactions among users and mobile devices have a significant impact on wireless network because the devices are carried by users. Thus, the social relationship is regarded as a perspective tool to address the problem in caching, and some efforts on the user social attributes are researched and utilized to resolve the problems on caching. [7, 8]. The social characteristics consist of ties, community, centrality, and bridge. By using the close social ties, the users could easily find the people who have the same interests with them in the MSN. In addition, the users prefer to share contents with their familiar or some people who have common interests in the real world. However, how to match the users and allocate the contents to the users remained challenges. To solve the problem, users' social attributes can be exploited for developing the caching strategy, which enables the users to provide assistance on caching and enhance the cache hit ratio of the system.

In this paper, we divide the users into active users and inactive users depending on their activeness. We establish two layers of networks, which consists of physical layer and social layer, to design the caching strategy. In physical domain, we consider the quality of communication links. Meanwhile, in social domain, we analyze the users social relationships which include users similarity and users' trust. In this case, we present a caching strategy based on a joint action deep Q-networks (JADQN) framework to choose cache nodes and cache contents. The proposed strategy is aiming to maximize the cache hit ratio and reduce the average download delay. We summarize the main contributions of this paper as follows.

We consider the attributes of physical domain and social domain. In the social domain, the user similarity is calculated by three dimensions (i.e., user content rating, user preference for the genre of content, and the number of common friends between users). The trust topology consists of users' direct trust and indirect trust which can be obtained by users' interaction and trust transitivity.

We propose a JADQN framework to match the users and allocate the contents based on users' preference in inactive users remain space. Based on CiaoDVD, the proposed strategy performs a higher cache hit ratio and lower average download delay compared with the classical algorithm.

Table 1: Summary of notations.

Notation	Interpretation
M	The number of active users
N	The number of inactive users
S_n	Remaining space size of inactive user n
τ	Delay of download contents
Γ	Delay threshold
$Tr_{m,n}$	Trust degree between user m and user n
$\mathrm{Sim}_{m,n}$	Similarity between user m and user n
H	Cache hit ratio
s_t	State at decision epoch t
$a_t = (x_t, y_t)$	A joint action at a decision epoch t
r_t	Reward at a decision epoch t
γ	Discount rate
ϵ	Explore rate

The rest of this paper is structured as follows. The related works are introduced by us in Section 2. In Section 3, the system model including physical domain and social domain is described. After that, we formulate the problem in Section 4. Then, we present a JADQN framework to design caching strategy in Section 5. Next, we analyze the CiaoDVD dataset and discuss the simulation results in Section 6. Finally, we give the conclusion and the future work in Section 7. The important notations used in this paper are listed in Table 1.

2. Related Works

A huge number of researches combine D2D-assisted caching and social attributes in the existing literatures. In [9], the authors have utilized the social relationship between users to design the caching scheme at the network edge. In [10], a genetic algorithm-based collaborate caching strategy has been used to motivate the collaboration nodes; further, a submodular function has been used to optimize energy consumption. In [11], an incentive mechanism has been designed to minimize the cost of the BS, and the users who have shared contents with others by D2D communications could obtain a reward. However, these works have modeled the request probability of contents as a Zipf distribution. In fact, users send the requests based on their preference rather than the assumed Zipf distribution. Therefore, a prior knowledge-based learning algorithm for user preference has been exploited to enhance the performance of the caching policy [12]. In order to maximize the caching utility and improve the probability of content sharing between the devices, a caching algorithm based on user preference has been proposed [13]. Nevertheless, user relationships influence the quality of communication because the devices are carried by users. Besides, users are reluctant to share their resources and provide assistance to others who have weak social relationship with them due to the selfish nature.

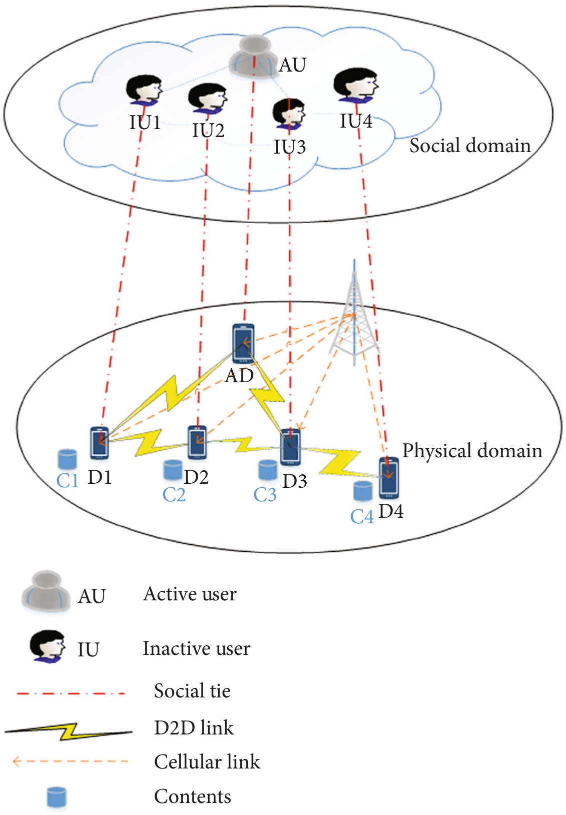

Figure 1: System model.

The problem can be solved by considering the social relationships between users including user intimacy, user similarity, and user trust. All of them can be used to select the nodes as a helper in caching contents. In addition, the users are willing to share the contents to the others in the same community with them. By this way, the cache resources of overall system can be utilized more effectively. In [14], the authors have investigated the caching strategy and cooperative distance to optimize the throughput of the system and improve energy efficiency, where user similarity is calculated by the set of common neighbors among users. In [15], the different time spans have been divided by temporal score information. A new measurement which weights similarity by the network adjacent matrices of diverse time spans has been presented. In [16], the authors have considered the similarity of user interest and adjusted some parameters to control the proportion of social attributes to generate stabler community. However, we should consider not only user similarity but also user trust to ensure transmission reliability. In [17], the authors have argued that the users trust each other if the connection exists between them. Besides, the trust of users can be divided into direct trust and indirect trust. Zhang and Zhong [18] have proposed a method to design trust model by trust transitivity feature. In [19], the authors have combined the direct trust of users and the indirect trust to obtain the trust similarity by the Pearson similarity formula.

In addition, reinforcement learning (RL) is applied to solve the complex communication problems since the environment is becoming more and more complex [20]. Compared with traditional optimization theory, using RL to build environment model can resolve this kind of problem and achieve a significantly effect. In [21], the authors have used a JADQN framework to design an optimal offloading strategy. In [22], the authors have presented a joint framework which composed of mobile edge computing (MEC) and cached-enabled D2D to optimize the energy cost. In [23], the authors have formulated the caching problem as a Markov decision process (MDP) and proposed the RL-based cooperative caching strategy to learn the optimal policy to minimize the delay. In [24], the authors have proposed a model-free reinforcement learning- (RL-) based algorithm. Furthermore, the algorithm has weighted a large set of features including the object size, recency, and frequency of access to maximize the cache hit ratio. However, these works only have considered the influence of physical layer on the communication problems. It is complicated to integrate the social domain and physical domain because the conditions of the network environment (i.e., user similarity, user trust value, and wireless communication links) are dynamic. Thus, the complexity of the integrated network is very high, and it is difficult to solve the formulation optimization problem. Motivated by the previous analysis, we present a model consists of two layers including the physical factors and social attributes. Then, we design a strategy based on a JADQN framework to choose cache nodes and cache contents.

3. System Model

Figure 1 illustrates the considered system model which is divided into physical domain model and social domain model. The social domain and the physical domain are modeled separately; if two users have low social relationship, they may not establish the communication link. In physical domain, we obtain the communication link quality by calculating the download delay. Besides, we introduce the dataset used in this paper. In social domain, we utilize the data from CiaoDVD dataset to judge user relationship by calculating user similarity and user trust. Then, the contents are allotted according to the active user preference.

3.1. Physical Domain Model. In the physical domain, we consider a scenario of D2D-assisted networks, which is composed of mobile users and a base station (BS). In addition, we assume that the BS with huge cache ability could store all contents which can be represented by $\mathscr{F} = \{1, 2, \cdots, f, \cdots, F\}$. The users in the system are denoted by $\mathscr{U} = \{1, 2, \cdots, u, \cdots, U\}$. We suppose that each user takes a mobile smart device with D2D communication capability. The users will be divided into two categories based on their activeness: active users $\mathscr{M} = \{1, 2, \cdots, m, \cdots, M\}$ and and inactive users which expressed by $\mathscr{N} = \{1, 2, \cdots, n, \cdots, N\}$. The inactive users have extra memory size, and they can provide assistance for the active users who are close to them.

The active users send a large number of requests which cause them have no extra memory space. During off-peak periods, the inactive users with different cache size store the contents that active users are interested in, which are represented by $\mathscr{P} = \{\mathbf{1}, \mathbf{2}, \cdots, p, \cdots, P\}$, $\mathscr{P} \subset \mathscr{F}$. In more details, $\mathscr{P}$ is the order of rating contents by active users. Then, inactive users provide the contents to their neighbor during peak periods.

Assuming that the spectrum resources are orthogonality, which means that there is no interference between the inactive users and their neighbors. The active users communicate with the inactive users via D2D links which is subjected to Shannon bound. Thus, when the active user m gets content from the provider π_j, the download delay can be calculated as

$$\tau_{m,\pi_j} = \frac{B}{W \cdot \log_2\left(1 + \left(P_{\pi_j}/N_0\right) \cdot g_{m,\pi_j}\right)}, \tag{1}$$

where B denotes content size, W represents the channel bandwidth, P_{π_j} is the transmitted power of π_j, N_0 is the noise power between π_j and active user m, and g_{m,π_j} is the channel gain which consists of small-scale fading due to multipath effect and shadow fading in large-scale fading. Therefore, g_{m,π_j} can be expressed as follows:

$$g_{m,\pi_j} = \upsilon \vartheta_{m,\pi_j} \xi_{m,\pi_j} d_{m,\pi_j}^{-\alpha}, \tag{2}$$

where υ represents the path loss constant which will be provided at Section 4. ϑ_{m,π_j} represents the small-scale fading, and ξ_{m,π_j} denotes the large-scale fading. $d_{m,\pi_j}^{-\alpha}$ represents the distance from user m to π_j, and α is the path loss factor.

When an active user m sends a content request to the BS, the neighbor of user m has the precedence to provide the contents. The D2D links can be established between the active user m and many inactive users. Thus, the BS selects the inactive user n with the minimum communication delay τ_{m,π_n} from the set of inactive users. Besides, the download delay between the inactive user n and m must satisfy the delay threshold Γ; otherwise, the inactive user n will be discarded. If none of the inactive users adjacent to the active user m can provide the requested content, the user m downloads the content from the BS and the delay denoted by $\tau_{m,MBS}$.

3.2. Dataset Analysis. In this subsection, CiaoDVD dataset is introduced and analyzed. Based on CiaoDVD, Chen et al., utilized temporal score information to calculate user interest similarity [25]. In this paper, the similarity between users is calculated in three dimensions: user content rating, user preference for the genre of content, and the number of common friends between users. Besides, according to the CiaoDVD dataset, we can establish a trust topology for users.

In the dataset, all users rate their favorite contents on a five-point scale: 1, 2, 3, 4, and 5. Moreover, 0 represents the content is not rated by the user. Thus, the score matrix of users is expressed as

$$R_v = \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \\ \vdots \\ U_{m+n} \end{array} \begin{array}{c} \begin{array}{ccccccc} \mathrm{f}_1 & f_2 & f_3 & f_4 & f_5 & \cdots & f_F \end{array} \\ \begin{pmatrix} 5 & 3 & 4 & 3 & 0 & \cdots & 3 \\ 5 & 2 & 4 & 0 & 5 & \cdots & 0 \\ 0 & 4 & 3 & 5 & 2 & \cdots & 4 \\ 0 & 5 & 4 & 3 & 0 & \cdots & 0 \\ 5 & 5 & 4 & 4 & 0 & \cdots & 5 \end{pmatrix} \end{array}. \tag{3}$$

From (3), it can be found that the score of U_1 for the content f_1 is 5, and the score of U_3 for the content f_3 is 3.

Each content in the dataset corresponds to a content type. There are 17 genres of movies in the dataset, and all of them have a fixed label. In addition, users may collect more than one content for the same type. By the statistics, the number of content that user m and user n collected can be expressed as

$$K_v = \begin{array}{c} \\ m \\ n \end{array} \begin{array}{c} \begin{array}{ccccccc} J_1 & J_2 & J_3 & J_4 & J_5 & \cdots & J_{17} \end{array} \\ \begin{pmatrix} 0 & 1 & 3 & 2 & 1 & \cdots & 5 \\ 1 & 1 & 4 & 11 & 2 & \cdots & 0 \end{pmatrix} \end{array}. \tag{4}$$

From (4), it can be found that m collects three contents which belong to type j_3 and two contents which belongs to type j_4. In addition, type j_{17} is the favorite genre of m, and type j_4 is the favorite genre of n.

There are the user trust relationships in the dataset, and the following matrix denotes the trust degree of first five users.

$$R_v = \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{array} \begin{array}{c} \begin{array}{ccccc} U_1 & U_2 & U_3 & U_4 & U_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}. \tag{5}$$

As shown in (5), if a trust relationship exists between two users, the trust value is 1; if there is no direct trust relationship between two users, the trust value is 0. Moreover, the trust relationship between users and users themselves is represented by 1.

3.3. Social Domain Model. In this subsection, CiaoDVD dataset is introduced and analyzed. Based on CiaoDVD, Khan et al., utilized temporal score information calculated user interest similarity [9]. In this paper, the similarity between users is calculated in three dimensions: user content rating, user preference for the genre of content, and the number of common friends between users. Besides, according to the CiaoDVD dataset, we can establish a trust topology for users.

3.3.1. User Similarity Analysis. The Jaccard similarity algorithm is utilized to calculate the similarity in this paper. We consider three dimensions different from other literatures [26, 27], including user collected contents, user favorite content genre, and the number of common friends of the users. By this way, the characteristic set of user m and user n is expressed by $F_{m,n} = \{f^i_{m,n}, J_{m,n}, fr^u_{m,n}\}$. In detail, $f^i_{m,n}$ is the favorite list of the mth and the nth user, $J_{m,n}$ is the movie genre that both user m and user n collected, and $fr^u_{m,n}$ denotes the common friend list of user m and user n.

We suppose that once a user has rated the content, the content is considered to be put on their favorite lists. In this case, the content matric of user m and user n can be expressed as

$$R_m = \begin{array}{ccccccc} fr^1_{m,n} & fr^2_{m,n} & fr^3_{m,n} & fr^4_{m,n} & fr^5_{m,n} & \cdots & fr^i_{m,n} \\ (0 & 1 & 0 & 1 & 1 & \cdots & 1) \end{array}, \tag{6}$$

$$R_n = \begin{array}{ccccccc} fr^1_{m,n} & fr^2_{m,n} & fr^3_{m,n} & fr^4_{m,n} & fr^5_{m,n} & \cdots & fr^i_{m,n} \\ (0 & 1 & 0 & 1 & 1 & \cdots & 1) \end{array}, \tag{7}$$

where R_m is the set of favorite contents for user m and R_n is the set of favorite contents of user n. From (6) and (7), we can see that the contents collected by user m are IDs 1, 2, 3, 4, and 5. For user n, the content IDs with 2, 4, and 5 are collected.

K_m represents the movie genre list rated by the user m, and there are 17 genres in total dataset. $K^{J_1}_m = 1$ represents user m prefers the J_1; otherwise, $K^{J_1}_m = 0$. It is noted that each user has only one favorite content genre.

$$K_m = \begin{array}{ccccccc} J_1 & J_2 & J_3 & J_4 & J_5 & \cdots & J_{17} \\ (0 & 0 & 1 & 0 & 0 & \cdots & 0) \end{array}, \tag{8}$$

$$K_n = \begin{array}{ccccccc} J_1 & J_2 & J_3 & J_4 & J_5 & \cdots & J_{17} \\ (0 & 1 & 0 & 0 & 0 & \cdots & 0) \end{array}. \tag{9}$$

From (8), it is obvious that the favorite content of user m is genre J_3. In the same way, from (9), it can be seen that user n is genre J_2.

T_m and T_n denote the friend list of user m and user n, respectively. $fr^u_{m,n}$ denotes the user list involving the friends of user m and the friends of user n, i.e., $fr^u_{m,n} = fr^A_m \bigcup fr^B_n$, where fr^A_m is the friend list of user m who has A friends and fr^B_n is the friend list of user n who has B friends. If user u is the friend of user m, $fr^u_m = 1$; otherwise, $fr^u_m = 0$. Similarly, if user u is the friend of user n, $fr^u_n = 1$. Otherwise, $fr^u_n = 0$. Thus, the friend matrix of user m and user n can be expressed as follows:

$$T_m = \begin{array}{cccccc} fr^1_{m,n} & fr^2_{m,n} & fr^3_{m,n} & fr^4_{m,n} & \cdots & fr^u_{m,n} \\ (1 & 0 & 1 & 1 & \cdots & 0) \end{array}, \tag{10}$$

$$T_n = \begin{array}{cccccc} fr^1_{m,n} & fr^2_{m,n} & fr^3_{m,n} & fr^4_{m,n} & \cdots & fr^u_{m,n} \\ (1 & 1 & 0 & 0 & \cdots & 0) \end{array}. \tag{11}$$

In (10), obviously, user 1, user 3, and user 4 are familiar with user m, and user 2 is not the friend of user m. Similarly, in (11), user 1 and user 2 are familiar with user n, and user 3 and user 4 are not the friend of user n. Moreover, user 1 is the common friend between m and n.

In that way, the feature set of user m can be expressed as $F_m = \{R_m, K_m, T_m\}$. Similarly, the feature set of user n can be expressed as $F_n = \{R_n, K_n, T_n\}$. Thus, Jaccard similarity algorithm can be used as

$$\mathrm{Sim}_{m,n} = \frac{|F_m \bigcap F_n|}{|F_m \cup F_n|}. \tag{12}$$

3.3.2. User Trust Analysis. If m and n contact directly $m \leftrightarrow n$, it is believed that there is a direct trust relationship, which can be denoted as $DT_{m,n}$. However, not all users exist direct trust, which contributes to the trust matrix sparse. In fact, although there is no direct contact between the two users, it does not mean that users do not trust each other, which can be denoted as $m \sim n$. At this point, the indirect trust can be calculated by using the transitivity of trust, denoted by $PT_{m,n}$. In other words, if there is at least one set of intermediate users between m and n, $\mathcal{O} \subset \mathcal{U} = \{o_1, o_2, \cdots, o_v\}$, which enables the users trust each other: $m \longrightarrow o_1 \longrightarrow o_2 \longrightarrow \cdots \longrightarrow o_v \longrightarrow n$. Nevertheless, there are more than one intermediate user groups, and the trust network is regarded as a resistance network [26].

$$\frac{1}{PT_{m,n}} = \frac{1}{DT_{m,o_1}} + \frac{1}{DT_{o_1,o_2}} + \frac{1}{DT_{o_1,o_v}} + \frac{1}{DT_{o_v,n}}, \tag{13}$$

where $PT_{m,n} \in (0, 1)$. According to six degrees of separation theory [27], a person can be able to know a stranger through six friends. In case of intermediate users more than six, $PT_{m,n} = 0$. In details, based on the data which expressed like Equation (5), we can know whether the two users trust each other directly. If two users have no direct trust and the intermediate users of them are less than six, we can utilize Equation (13) to calculate the indirect trust value between them. Otherwise, we regard the two users trust value is 0. Thus, the trust calculation formula can be given by

$$Tr_{m,n} = \begin{cases} DT_{m,n}, & \text{if } m \leftrightarrow n, \\ PT_{m,n}, & \text{if } m \sim n \quad \&(\exists \mathcal{O} \& |\mathcal{O}| \leq 6), \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

BS selects user n from the set of inactive users adjacent to the active user m by weighing the link quality in physical domain and the relationship in social domain.

4. Problem Formulation

The caching strategy can be modeled as a Markov decision process (MDP) which is regarded as the states in the future depending only on the present state rather than the past history. In this paper, we use deep Q-networks (DQN) framework to design the caching strategy, including matching the users and allotting the contents to the helper nodes, which can be expressed by $\psi = (u_m^n, y_n)$, where u_m^n denotes the inactive user m paired with active user n. That is, BS puts the contents in the remaining cache space of user n. y_n is the content list which cached in user n.

The cache hit ratio in this paper is the ratio of the hit times of the active user all requested during a period of time T, which can be formulated as

$$H = \frac{\sum_{t=1}^{T} \sum_{p=1}^{P_t} h_{m,n}^t(p)}{\sum_{t=1}^{T} P_t} = \frac{1}{P} \sum_{t=1}^{T} \sum_{p=1}^{P_t} h_{m,n}^t(p), \qquad (15)$$

where $h_{m,n}^t \in \{0, 1\}$ indicates the cache status by the active user m at time slot t. It is noted that each content has a corresponding content genre. We suppose that the active user m is paired with the inactive user n and sends a request to download the content p of genre J_f. It is called a hit if J_f exists in the cache space of the inactive user n, denoted by $h_{m,n}^t = 1$. In addition, P_t expresses the count of request during the period.

Both matching users and choosing contents which should be cached in the remaining space of inactive users can affect the cache hit rate. Therefore, the goal is to obtain an optimal policy ψ^* to find the proper content placement location and cache the active user prefer contents for maximizing the cache hit rate H.

$$\begin{aligned}
\max_{\psi} \quad & \frac{1}{P} \sum_{t=1}^{T} \sum_{p=1}^{P_t} h_{m,n}^t(p), \\
s.t. \quad & C1 : \sum_{p} c_{n,p} \le S_n, \\
& C2 : \tau_{m,\pi_j} < \Gamma, \\
& C3 : h_{m,n}^t(p) \epsilon \{0, 1\}, \\
& C4 : 0 \le \mathrm{Sim}_{m,n} \le 1, \\
& C5 : 0 \le Tr_{m,n} \le 1,
\end{aligned} \qquad (16)$$

where p represents the content that user n cached and $c_{n,p}$ is the size of p. Furthermore, $C1$ denotes that $c_{n,p}$ shall be smaller than the remain storage space S_n of inactive user n. $C2$ denotes that the delay of caching content for user m should be less than the delay threshold Γ. Otherwise, the communication link cannot be established. $C3$ represents two cache states, $h_{m,n}^t(p) = 1$ means the content can be provided by the inactive users nearly, and $h_{m,n}^t(p) = 0$ means the content is not cached by the adjacent inactive users. $C4$ means the trust value located in the range of $[0, 1]$, and $C5$ means the similarity value located in the range of $[0, 1]$.

5. Joint Action DQN-Based Caching Strategy

In this section, we propose a caching strategy based on a JADQN framework. We treated the BS as an agent who knows the cache status of all users. When an active user sends a content request, the agent observes the state space to make a joint action, including selecting inactive users and allotting the caching content placement process. Finally, the goal of maximizing cache hit is achieved.

It is well known that Q-learning is a widely used model-free reinforcement learning. However, due to the complex environment, the state space is huge, and one of the challenges in Q-learning is to store the state in Q table. Thus, DQN who combines the advantages of Q-leaning and neural network is proposed. The DQN framework consists of an agent, an environment, and three crucial elements including the state space $\mathscr{S}$, the action space $\mathscr{A}$, and a reward function $\mathscr{R}$. The agent learns and makes decisions through interacting with the environment in discrete time steps. In each decision time slot, the agent observes the current state of the environment $s_t \in \mathscr{S}$ and takes an action $a_t \in \mathscr{A}$, then the agent will obtain a reward $r_t \in \mathscr{R}$. We utilize the DQN framework to cope with the problem and define the key elements according to our system model as follows.

5.1. State Space. The environment is defined as the communication conditions of D2D users, which consists of physical condition and social condition. In this case, the state s_t includes the inactive users remaining cache space $S_n(t)$ which expresses the users caching ability in the physical domain, the favorite movie genre of user m is denoted by J_m; the trust degree and the similarity between user m and user n are denoted by $Tr_{m,n}$ and $\mathrm{Sim}_{m,n}$, respectively. It is noted that the J_m, $Tr_{m,n}$, and $\mathrm{Sim}_{m,n}$ express the characteristics of users in social domain. In addition, $S_n(t)$, $Tr_{m,n}$, and $\mathrm{Sim}_{m,n}$ are detailed described in Section 4. At the beginning of each decision slot t, the agent selects different users whose remaining space is different and the social relationships between them change dynamically. Thus, s_t can be written as

$$s_t = \{\mathrm{Sim}_{m,n}, Tr_{m,n}, S_n(t), J_m\}. \qquad (17)$$

5.2. Action Space. There are N inactive users and P contents to be cached. In order to restrict the size of the action space, it is assumed that only one inactive user can be selected to place cache contents when an active user requests content. The agent chooses a joint action a_t to decision where and what to cache based on the observed environment state s_t of time slot t. That is, the agent completes the pairing process between active user m and inactive user n. At the same time, the contents are allocated to the remaining cache space of the inactive user n paired with the active user m. Thus, the action can be expressed by

$$a_t = \{(x_1(t), y_1(t)), (x_2(t), y_2(t)), \cdots, (x_n(t), y_n(t))\}. \qquad (18)$$

The dimension of the action space can be expressed as $N \times P$, where $x_n(t) \in \{0, 1\}$ represents the paring status of active user m. $x_n(t) = 0$ denotes that BS selects other inactive

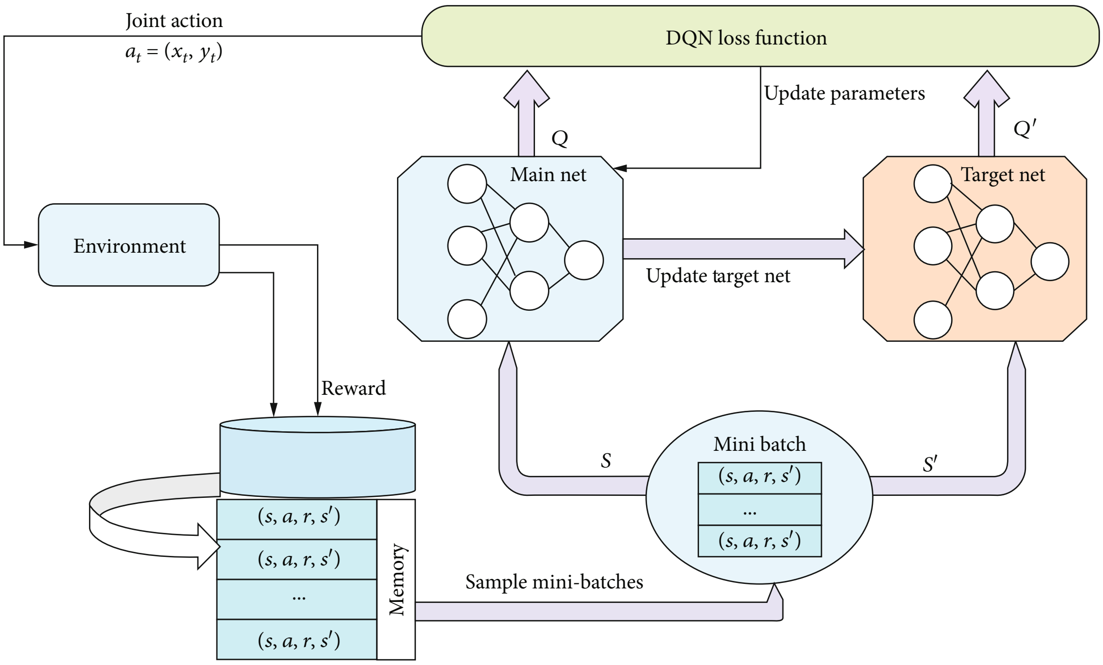

Figure 2: A joint action DQN framework.

user rather than inactive user n. $x_n(t) = 1$ denotes that the inactive user n is selected by BS to help the active user cache contents.

When the active user m is paired with the inactive user n, the agent will allocate contents to the user n according to the active user m preference and the remaining cache space of the selected user n. Besides, $y_n(t) = \{1, 2, \cdots, k\}$, $k \in \mathscr{P}$ represents a cache list of the inactive user. However, the inactive users have limited cache space; the inactive user selected by the agent may not cache the content that the active user requests. If the content genre does not exist in the memory of the inactive users adjacent to the active user, the active user downloads the content from the BS.

5.3. Reward Design. In the caching strategy described in Section 4, the objective is to maximize the cache hit ratio within a certain time constraint T. When we give a particular state s_t and a joint action a_t at time slot t, the agent obtains a larger reward when the selected action results in a higher cumulative hit times before the next decision epoch. Hence, the reward is defined as a sum of hit count on each cache slot, and it can be formulated as

$$r_t = \sum_{i}^{P} h_{m,n}^{t}(t+1) - h_{m,n}^{t}(t). \tag{19}$$

The joint action DQN framework is shown in Figure 2, which consists of two neural networks which have the same structure but their parameters are different, that is, target-Q network with θ and main network with θ^{-}. The value of main network is represented as $Q = (s_t, (x_t, y_t), \theta)$ obtained by $a_t = (x_t, y_t)$. Note that a joint action DQN and Double DQN are different. JADQN used in this paper obtains the target-Q value by greedy algorithm. Double DQN obtains the target-Q by two steps, that is, decouple the selection of target-Q and the calculation of target-Q value. The target-Q value of JADQN is calculated like the DQN apart from making a joint action in a decision slot, and the target-Q value can be calculated by

$$Q' = r_{t+1} + \gamma \max_{(x_t, y_t)} Q(s_{t+1}, (x_t, y_t)\,; \theta^{-}). \tag{20}$$

Since exploration and exploitation are contradictory, the most general way to balance the tradeoff between them is by using ϵ-greedy policy. Specially, the agent chooses the actions that have been tried and shown to be effective with probability $1-\epsilon$ or the action is chosen randomly with probability ϵ to explore the state space and action space. We train the network through experience replay to learn the caching strategy. In detail, the agent observes the environment state s_t to make a joint action (x_t, y_t), which obtains a reward and affects the state of next moment s_{t+1}. After that, the agent stores the transition tuple $(s_t, (x_t, y_t), r_t, s_{t+1})$ in the replay memory $\mathscr{D}$. The transition in $\mathscr{D}$ consists of the current state s_t, the current action a_t, the reward r_t calculated by formula (19), and the state of the next decision epoch after the environment receives the action s_{t+1}. Furthermore, at each episode, minibatches are tokens from $\mathscr{D}$. The network updates parameter θ and minimizes the mean square error as follows:

$$\mathscr{L} = \left\| Q' - Q \right\|. \tag{21}$$

The proposed solution is shown in Algorithm 1, and the algorithm is divided into six steps as follows.

```
Input: inactive users remaining space: $S_N$, transmission distance of users: $d_N$, delay threshold: $\Gamma$, the parameter of main $Q$ network: $\theta$,
the parameter of target $Q$ network: $\theta^-$ and read the data from dataset.
Output: the cache hit ratio $H$.
Initialize: $\theta^- \longleftarrow \theta$, memory size: $\mathscr{D}$, leaning rate $\beta$, explore rate: $\epsilon$, discount rate: $\gamma$, the number of iterations:$E$, the number of mini-
batches:$M$.
if$\tau_{m,n} < \Gamma$ then
        for episode=$1, 2, \cdots E$ do
              Update $S_N$; $\mathscr{P}$
              for t $=1, 2, \cdots T$ do
                 Random generation probability: $\rho$
                       if $\rho < \epsilon$:
                 Randomly generated action: $(x_t, y_t)$
                       else:
                             Obtained a joint action by according to $(x_t, y_t) = \arg \max_{(x_t,y_t)} Q(s_t, (x_t, y_t), \theta)$
                       end if
                 Observe $s_{t+1}$
              Store the transaction $(s_t, a_t, s_{t+1}, a_{t+1})$ in replay memory $\mathscr{D}$, uniformly sample minibatches from $\mathscr{D}$
              Optimize error between $Q$-network and learning targets, using variant of stochastic gradient descent $\theta = \theta - \beta(d\mathscr{L}/d\theta)$
              Each step $C$ updates the parameters of target $Q$-network $\theta^- \longleftarrow \theta$
              end for
        end for
end if
```

Algorithm 1: Caching strategy based on a JADQN.

Step 1. Input the variables and initialize them including the remaining space set of inactive users $S_N = \{S_1, S_2, \cdots, S_n\}$, the communication delay set between inactive users and the active user $d_N = \{d_1, d_2, \cdots, d_n\}$, and the delay threshold Γ. Besides, we set the parameters of neural networks as θ and θ^-, respectively, the number of iterations as E, the minibatch size as M, and the replay memory size as $\mathscr{D}$. At the same time, we read the data from the CiaoDVD dataset to obtained the user preference $\mathscr{P}$ and calculate the user similarity $\mathrm{Sim}_{m,n}$, user trust $Tr_{m,n}$.

Step 2. The communication delay between the inactive user and the active user determines whether to discard the inactive user. In detail, the agent discards the inactive user and chooses another one if the cache download delay is greater than the threshold Γ.

Step 3. In each episode, update the users preference $\mathscr{P}$ and the remaining space of inactive users S_N. The agent observes the current state space when the active user sends the content request based on their preference $\mathscr{P}$. Then, in the each training step, the agent takes the joint action by the value network and ϵ-greedy policy. We used the replay memory $\mathscr{D}$ to store the transition tuple $(s_t, a_t, r_t, s_{t+1},)$ in order to break up the correlation between the data.

Step 4. Sample minibatches from $\mathscr{D}$ randomly and uniformly. The networks are training by utilizing variant of stochastic gradient descent.

Step 5. Update the target-Q network parameters for each C steps and train the network by calculating the loss in formula (21).

Table 2: Simulation parameters.

Parameter	Value
Transmission power of BS	46 dBm
Transmission power of users	23 dBm
Noise power	-114 dBm
Path-loss constant υ	0.01
Path-loss exponent α	4.0
Multipath fading ϑ	8 db
Shadowing ξ	-114 dBm
System bandwidth	10 MHz
Discount rate γ	0.9
Learning rate β	0.001
Minibatch size M	2000
γ	Discount rate
Explore rate ϵ	0.02

Step 6. Calculate the cache hit ratio once the active user sends requests until the algorithm converges. Then, output the cache hit ratio H.

6. Simulation Results

In this section, we analyze the performance of the proposed strategy by utilizing the dataset CiaoDVD. We divide the dataset into two parts, one part is used as the historical request data, the other is regarded as the user requests. In addition, the preference of users is calculated based on the historical request data. We compare our strategy with several benchmark algorithms including least recently used

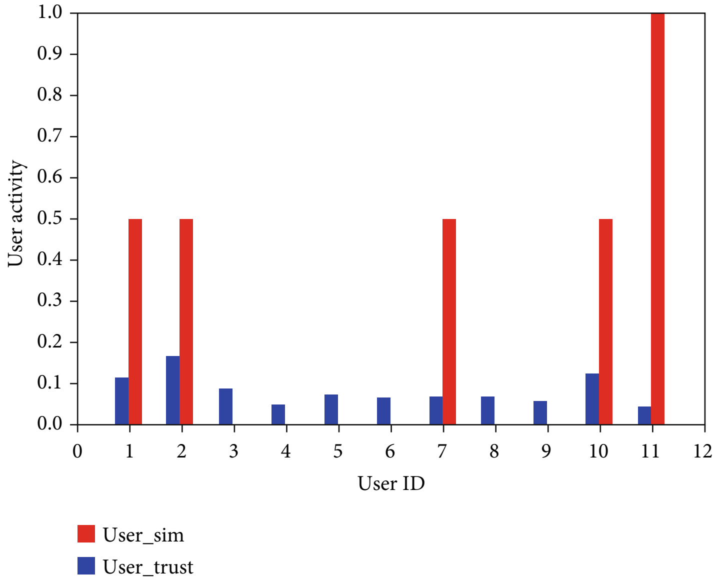

Figure 3: User similarity and user trust analysis.

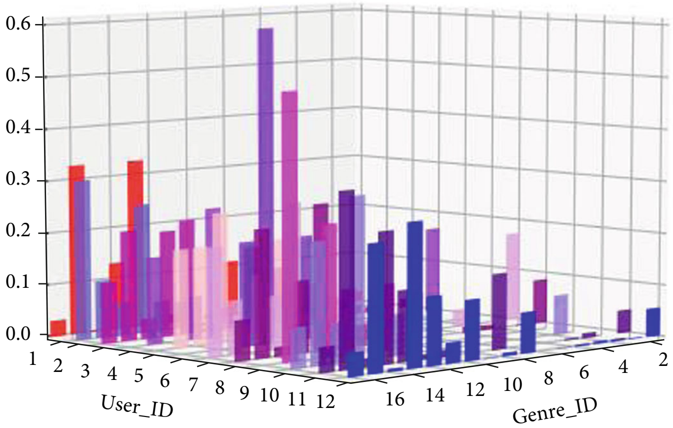

Figure 4: User preference analysis.

(LRU) algorithm, least frequently used (LFU) algorithm, and first input first output (FIFO) algorithm to validate the effectiveness of the proposed caching strategy.

In detail, when the cache space is filled with the contents, LRU algorithm evicts the earliest last used time content and caches the request content. For LFU algorithm, the least frequently request content is replaced. For FIFO algorithm, the content input first is replaced first. Moreover, we compare two algorithms (i.e., randomly select users and randomly cache: RU-RC, randomly select users and select users according to user preference: RU-PC) in order to show the performance advantages brought by selecting users. Table 2 lists the main simulation parameters [28, 29].

We select an active user and 11 inactive users from the CiaoDVD dataset randomly, which concludes 72665 users, then we analyze their social relationship. It is easy to get the trust value of an active user and eleven inactive users by trust topology network in (4). As shown in Figure 3, the value range of similarity and trust is between 0 and 1. It unable to establish trust relationship with the trust topology of six close friends between the user m and the user whose IDs are 3, 4, 5, 6, 8, and 9, respectively. Thus, the trust value of them is 0. Based on all users' historical information, the preference for movie genres can be obtained. We calculate the similarity by formula (5) and obtain the value in the range from 0.1 to 0.2. Besides, it can be seen that the inactive user 2 has the highest similarity to the active user. In addition, the value of the similarity is about 0.16. The trust between user 11 and the selected active user is the highest, with a value of 1.

Figure 4 shows user preferences obtained from historical movie requests of different users. According to the dataset, we divide the users into active users and inactive users depending on their activeness. In detail, from user 1 to user 11 are inactive users, and user 12 is active user. Besides, the dataset includes a total of 17 movie types such as action, comedy, and love. In addition, the users in the dataset have different genre preference. The value represents the degree of

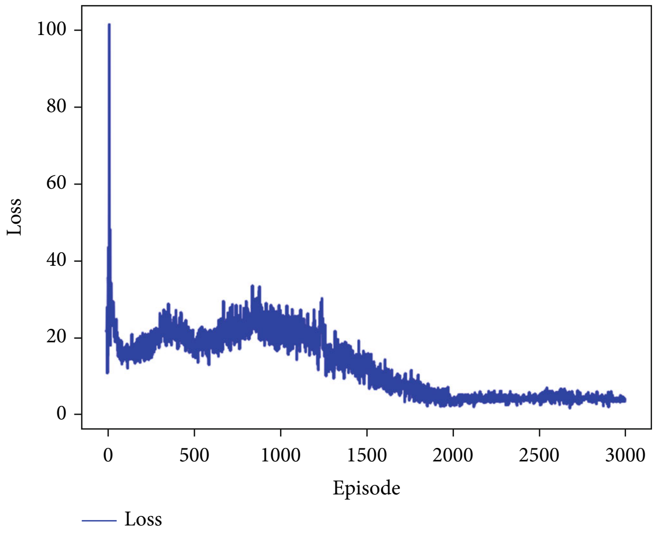

Figure 5: Loss for each training episode with increasing iterations.

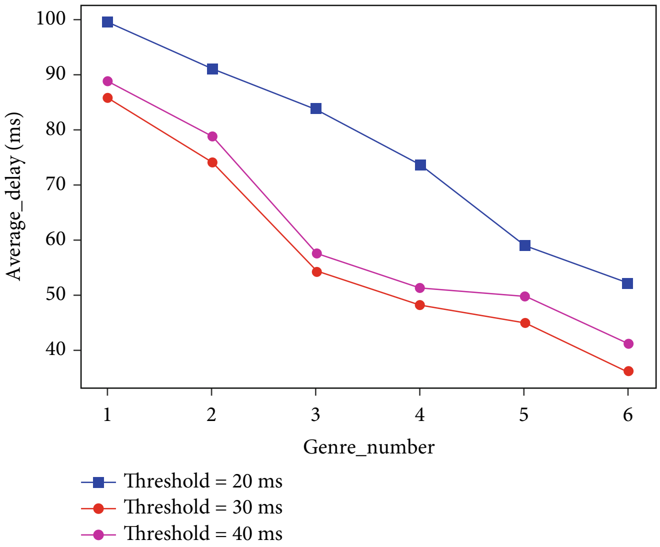

Figure 6: User cache genre vs. average delay.

user preference for the different genres which is defined as the ratio of contents collected by the user in this paper. That is, in all the contents that the user collected, the user preference is the proportion of the contents which belongs to the genre to the total amount of contents. It can be seen that user 2's preference value for genre 17 is about 0.3. Moreover, user 2's favorite movie genre is 17, and user 12's preference value for movie genre 16 is about 0.2, and user 12's favorite movie genre is 15. Based on the analysis of the users preference, we can utilize them in the caching strategy.

The DQN framework consists of 3 hidden layers, and the rectified linear unit (ReLU) is employed as an activation function. The learning rate is set as 0.001 by PRMProp optimizer to update network parameters. Besides, this is set as 0.02. We show the loss of each training episode as the number of training iterations increasing in Figure 5. It can be observed that the neural network continues to explore actions to get better reward in the early stages of training. Therefore, it has a large loss. However, as the number of episodes increases, the loss decreases in each training episode which demonstrates the effectiveness of our algorithm. In addition, when the number of training episode reaches 2000, the performance gradually converges and the DQN framework explores all the possible states and enables the agent select the best joint action depended on the states and rewards. Based on such an observation, we set the training episode as 3000 to guarantee convergence of DQN.

The average download delay performance of caching strategies with different cached genre number is shown in Figure 6. It can be observed that the number of cache genre influences the average delay. With the cache genre number increases, the cache average delay decreases obviously. When the user space only caches fewer genres, the average download delay is higher. Since the number of user cache genres is limited, it is difficult for active users to obtain the requested content from their neighbor. If they want to get

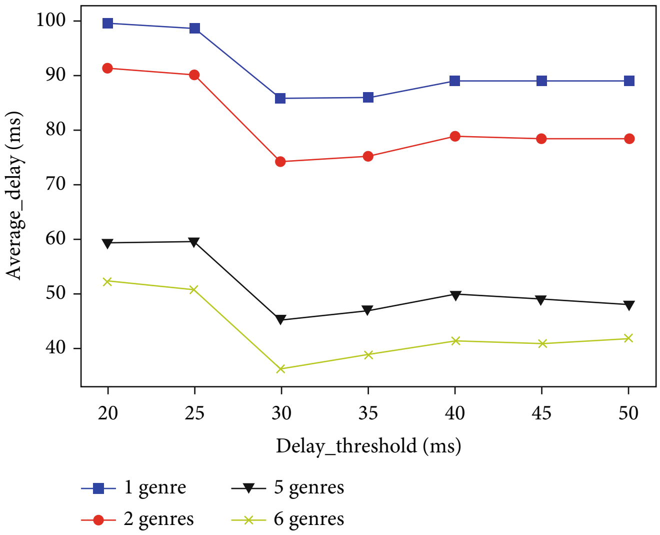

Figure 7: Delay threshold vs. average delay.

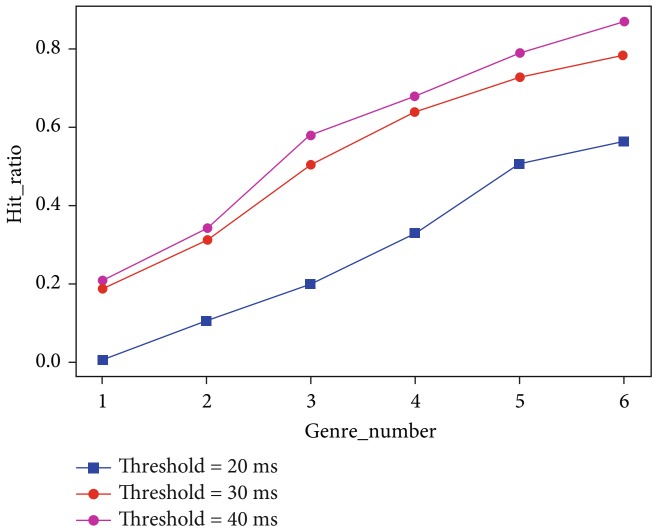

Figure 8: User cache genre vs. cache hit ratio.

the content only cache from the BS where far away from them, it decides the higher cache download delay. As the genres of contents cached by the inactive users increase, the more contents that active users interested in are cached. In this case, the average download delay gradually decreases. For the genre of content that inactive user could cache is 5 and the delay threshold is 30 ms, the average download delay of presented strategy is 46 ms. When the delay threshold is set as 20 ms, the average download delay of presented strategy is 46 ms. When the delay threshold is set as 40 ms, the average download delay of presented strategy is 52 ms. Based on these results, it can be found that the delay threshold effects the cache download delay. For instance, the cache download delay decreases around 32.6% when the delay threshold is set as 30 ms compared with 40 ms. However, the cache download delay decreases around 13.0% when the delay threshold is set as 30 ms compared with 20 ms. Based on this observation, we believe that the larger delay threshold does not mean the lower cache delay, there may be an optimal value for the delay threshold.

Figure 7 depicts the cache download delay performance of the system under different delay threshold. With the delay threshold increases, the average download delay curve of active user cache contents shows decreases firstly, then the value increases over threshold, and the average download delay tends to be stable finally. The main reason is that our proposed caching strategy is affected by not only social attributes but also physical factors, i.e., the similarity and trust degree between users in social domain and the quality of communication links in physical domain. However, users with high similarity and trust may be far away from active users. When the value of delay threshold is set small, part of inactive users will be weeded out. This is because the distance between them to the active user is too far. They are

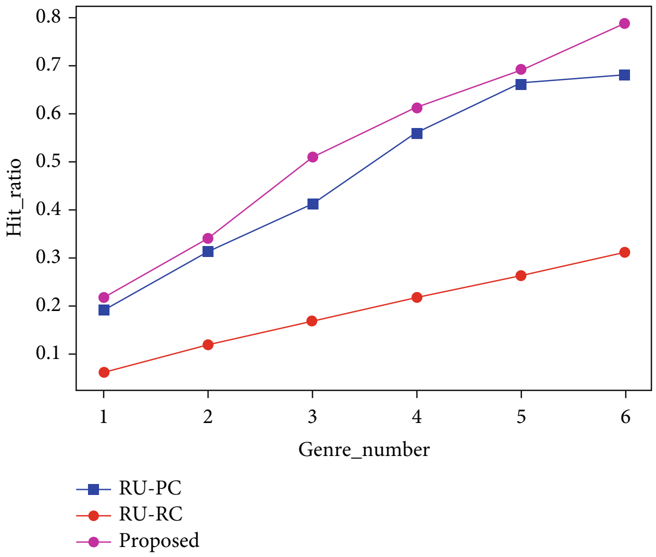

Figure 9: User cache genre vs. cache hit ratio.

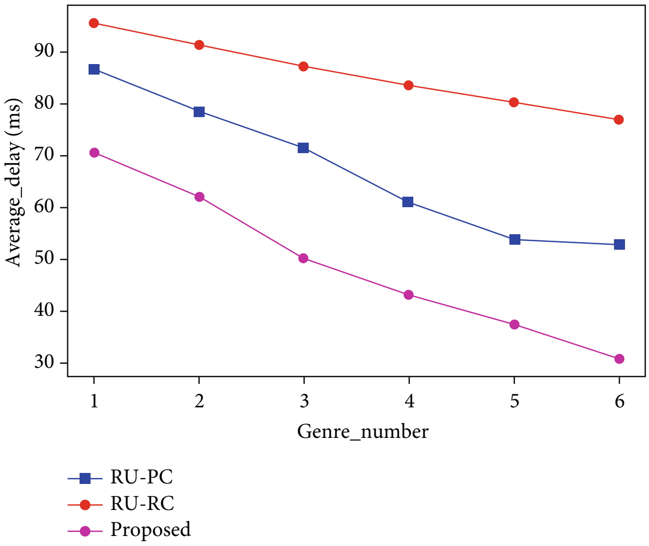

Figure 10: User cache genre vs. average delay.

unable to establish a communication link even if they have a high degree of similarity and trust to the active user. It can be seen that the delay threshold is 30 ms; a concave point is reached. When the delay threshold increases from 30 ms to 40 ms, the average download delay of the content requested by active users shows a trend of slow increase. The main reason is the number of users satisfying constraint in the physical domain. In this case, the agent prefers selecting inactive users who have closed relationships with the active user in the social domain to assist cache contents. In this way, the security and reliability of the communication can be ensured. However, the quality of communication links between the active user and the inactive users who are selected by the agent may not be optimal even if they satisfied the delay threshold. In addition, when the delay threshold is set as 45 ms, all inactive users who can provide services for active users meet the requirements of the physical domain. Therefore, if the download delay continues to rise, the average download delay of active users remains unchanged.

In Figure 8, we show the tendency of cache hit ratio with the different cached genre number. It can be observed that the cache hit ratio is increasing when the cache genre number inactive users can cache additionally changes from 1 to 6. In addition, as the delay threshold increases, the cache hit ratio increases accordingly. As the delay threshold is set as 20 ms, only a small part of inactive users satisfied the physical condition. Thus, not all inactive users could provide services and cache contents for active users, which results in low cache hit ratio. However, with the delay threshold increases, more and more users meet the condition result that the cache hit ratio is improved. For example, when the delay threshold is set as 40 ms, and the inactive user can store 5 movie genres, the cache hit ratio is 0.78. When the

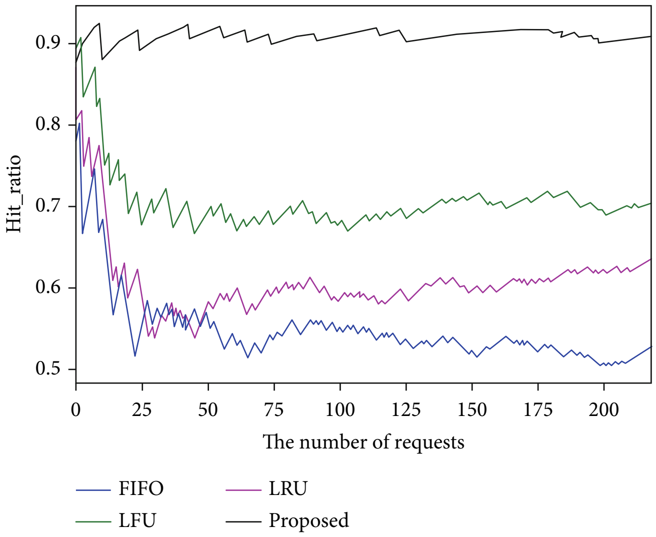

Figure 11: Request number vs. cache hit ratio.

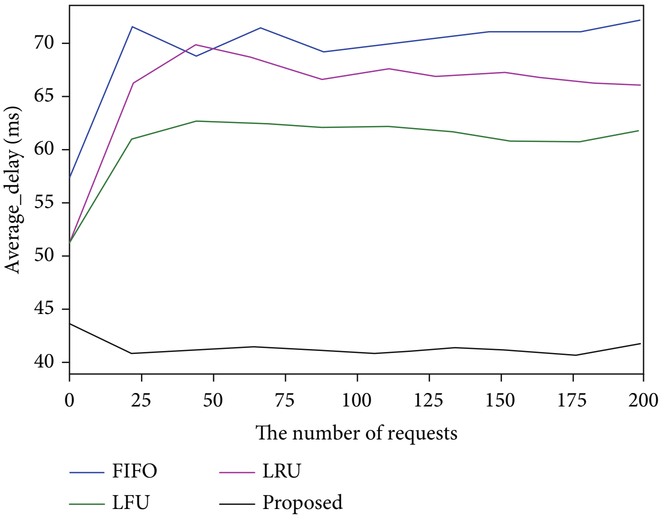

Figure 12: Request number vs. average delay.

delay threshold is set as 20 ms, the cache ratio is 0.47. Hence, we can know that the cache hit ratio is improved by about 47.7%, when the delay threshold increases from 20 ms to 30 ms.

Figure 9 demonstrates that the cache hit ratio brought by our caching strategy is significantly better than other caching strategies, when the leaning-based cache algorithm converges. The main reason is that the proposed strategy considers to select users according the social relationship between users and the quality of communication links. With the algorithm converges, the agent has the ability to choose the inactive user who is much proper than other users and provide the assistance to the active user. Thus, our strategy can obtain the better performance when the number of inactive users cached genre that active user prefers changes from 1 to 6. In addition, when the extra space of inactive user is small, the contents that the user cached are sparse. All the algorithms show the low cache hit ratio because the user cannot satisfy all demands. For example, when the delay threshold is set as 40 ms and the active users on the number of cache content genre is 3, the proposed caching strategy makes the cache hit ratio reach 0.51. The cache hit ratio of RU-PC caching strategy is 0.37, and RU-PC caching strategy is 0.16 under the same conditions. In this case, the cache hit ratio of a joint action based on DQN framework to select users and the contents is improved 37.8% than that of RU-PC.

In Figure 10, we show the impact of the genre number the inactive user could cache additionally on the average download delay. When the delay threshold is set as 40 ms, the average download delay of the algorithm proposed in this paper decreases gradually with the number of genres that inactive users cached increases. The main reason is that the neighboring inactive users provide assistant instead to obtain the contents from the BS is increasing. The content is placed closer and closer to active users; thus, the average download delay is reduced when the active users send the requests. However, the download delay performance fluctuates in the case of randomly cached content and randomly selected users. For RU-PC, the active users' interested contents which are obtained by historical requests are more likely to be cached. Therefore, its performance is better than RU-RC, but worse than the proposed strategy in this paper. When the inactive users could extraly cache 4 genres, the average download delay of the proposed algorithm is 46 ms, and the average download delay of RU-PC algorithm is 63 ms. Compared with RU-PC algorithm, the average download delay of the proposed caching strategy is reduced by about 36.9%.

Figure 11 illustrates the trend of cache hit ratio with the increasing number of content requests. Obviously, the cache hit ratio of LRU, LFU, and FIFO appear a tendency to decline and then level off gradually. However, the cache hit ratio of the proposed strategy fluctuates with the number of content requests increases. If the users can only cache four content genres, these three benchmark algorithms update the content when the preset storage space is filled with contents. It can be seen that our strategy shows much better performance than other algorithms. In more details, it is shown that the cache hit ratio of our strategy floats around 0.9. When the request number is 125, the cache hit ratio of our strategy improves by 49.1% compared with LRU algorithm.

As shown in Figure 12, with the increasing of content requests, the average download latency of content appears a trend that climbs up and then declines. Compared with the benchmark algorithms, the proposed strategy achieves better performance. If the inactive users nearby have no cached content, the content will be obtained from the BS, which results in much higher communication delay. As the D2D communication delay threshold is 30 ms, the download delay of the proposed strategy fluctuates between 40 ms and 45 ms. When the active user sends 100 content requests, compared with FIFO algorithm, the delay is reduced by 38.2%.

7. Conclusions

In this paper, we studied a JADQN framework to resolve the caching problem that includes choosing the cache location and allotting cache contents. To utilize the limited resources rationally, all the users in the network were divided into active users and inactive users according to their activeness. In addition, the inactive users could help active users cache contents with their extra cache space during the peak periods and provide the contents in the off-peak periods. In more details, we considered the physical domain and the social domain to design our caching strategy. In the physical domain, we calculated the communication delay to judge the quality of D2D links. In the social domain, we obtained users' social relationships from CiaoDVD dataset including user similarity and user trust. BS matched active users and inactive users who were willing to provide assistance. Furthermore, inactive users got information of active user preference from the dataset and used their remaining cache space to assist them cache contents. Numerical results showed that the cache hit rate of the proposed strategy was improved distinctly compared with the classical LFU caching strategy. In future work, we are going to design an incentive mechanism based on pricing with user service willingness to inspire more users as auxiliary nodes to participate in caching.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

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

Acknowledgments

This work was partially supported by Natural Science Foundation of China (Grants 61901070, 61801065, 61771082, 61871062, and U20A20157), in part by the Science and Technology Research Program of Chongqing Municipal Education Commission (Grants KJQN202000603 and KJQN201900611), in part by the Natural Science Foundation of Chongqing (Grant cstc2020jcyj-zdxmX0024), and in part by the University Innovation Research Group of Chongqing (Grant CXQT20017).

References

[1] X. Zhang, T. Lv, Y. Ren, W. Ni, N. C. Beaulieu, and Y. J. Guo, "Economical caching for scalable videos in cache-enabled heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 7, pp. 1608–1621, 2019.

[2] X. Zhao, P. Yuan, H. li, and S. Tang, "Collaborative edge caching in context-aware device-to-device networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 10, pp. 9583–9596, 2018.

[3] S. Han, F. Xue, C. Yang, J. Liu, and F. Lin, "Data-supported caching policy optimization for wireless D2D caching networks," *IEEE Transactions on Communications*, 2021.

[4] M. Lee, H. Feng, and A. F. Molisch, "Dynamic caching content replacement in base station assisted wireless D2D caching networks," *IEEE Access*, vol. 8, pp. 33909–33925, 2020.

[5] M. Waqas, Y. Niu, Y. Li et al., "A comprehensive survey on mobility-aware D2D communications: principles, practice and challenges," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1863–1886, 2020.

[6] J. Li, M. Liu, J. Lu et al., "On social-aware content caching for D2D-enabled cellular networks with matching theory," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 297–310, 2019.

[7] Y. Wang, M. Ding, Z. Chen, and L. Luo, "Caching placement with recommendation systems for cache-enabled mobile social networks," *IEEE Communications Letters*, vol. 21, no. 10, pp. 2266–2269, 2017.

[8] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. YANG, and W. Wang, "A survey on mobile edge networks: convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017.

[9] K. S. Khan, A. Naeem, and A. Jamalipour, “Incentive-based caching and communication in a clustered D2D network,” *IEEE Internet of Things Journal*, p. 1, 2021.

[10] S. Wang, H. Chen, and Y. Wang, “Collaborative caching for energy optimization in content-centric internet of things,” *IEEE Transactions on Computational Social Systems*, vol. 7, pp. 173407–173418, 2021.

[11] Z. Chen, Y. Liu, B. Zhou, and M. Tao, “Caching incentive design in wireless D2D networks: a Stackelberg game approach,” in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kuala Lumpur, Malaysia, 2016.

[12] B. Chen and C. Yang, “Caching policy for cache-enabled D2D communications by learning user preference,” *IEEE Transactions on Communications*, vol. 66, no. 12, pp. 6586–6601, 2018.

[13] T. Zhang, H. Fan, J. Loo, and D. Liu, “User preference aware caching deployment for device-to-device caching networks,” *IEEE Systems Journal*, vol. 13, no. 1, pp. 226–237, 2019.

[14] S. G. Hasson, J. Piorkowski, and I. McCulloh, “Social media as a main source of customer feedback,” in *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 829–832, 2019.

[15] J. Chen, L. Wei, L. U, and F. Hao, “A temporal recommendation mechanism based on signed network of user interest changes,” *IEEE Systems Journal*, vol. 14, no. 1, pp. 244–252, 2020.

[16] K. Gu, D. Liu, and K. Wang, “Social community detection scheme based on social-aware in mobile social networks,” *IEEE Access*, vol. 7, pp. 173407–173418, 2019.

[17] C. Ma, M. Ding, H. Chen et al., “Socially aware caching strategy in device-to-device communication networks,” *IEEE Access*, vol. 67, no. 5, pp. 4615–4629, 2018.

[18] S. Zhang and H. Zhong, “Mining users trust from E-commerce reviews based on sentiment similarity analysis,” *IEEE Access*, vol. 7, pp. 13523–13535, 2019.

[19] A. Kang, “Collaborative filtering algorithm based on trust and information entropy,” in *2018 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*, pp. 262–266, Bangkok, Thailand, 2018.

[20] A. Sadeghi, G. Wang, and G. B. Giannakis, “Deep reinforcement learning for adaptive caching in hierarchical content delivery networks,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1024–1033, 2019.

[21] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, “Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning,” *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4005–4018, 2019.

[22] J. Tang, H. Tang, X. Zhang et al., “Energy minimization in D2D-assisted cache-enabled internet of things: a deep reinforcement learning approach,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5412–5423, 2020.

[23] P. Lin, Q. Song, J. Song, A. Jamalipour, and F. R. Yu, “Cooperative caching and transmission in CoMP-integrated cellular networks using reinforcement learning,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5508–5520, 2020.

[24] V. Kirilin, A. Sundarrajan, S. Gorinsky, and R. K. Sitaraman, “RL-cache: learning-based cache admission for content delivery,” *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2372–2385, 2020.

[25] Y. Shao and C. Wang, “HIBoosting: A recommender system based on a gradient boosting machine,” *IEEE Access*, vol. 7, pp. 171013–171022, 2019.

[26] M. Taherian, M. Amini, and R. Jalili, “Trust inference in web-based social networks using resistive networks,” in *2008 Third International Conference on Internet and Web Applications and Services*, pp. 233–238, Athens, Greece, San Francisco, 2008.

[27] X. Ke, “A social networking services system based on the “six degrees of separation” theory and damping factors,” in *2010 Second International Conference on Future Networks*, pp. 438–441, Sanya, China, 2010.

[28] J. Iqbal and P. Giaccone, “Interest-based cooperative caching in multihop wireless networks,” in *2013 IEEE Globecom Workshops (GC Wkshps)*, pp. 617–622, Atlanta, GA, USA, 2013.

[29] L. Liang, H. Ye, and G. Y. Li, “Spectrum sharing in vehicular networks based on multi-agent reinforcement learning,” *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2282–2292, 2019.