Heterogeneous Edge Caching based on Actor-Critic Learning with Attention Mechanism Aiding

Chenyang Wang, *Member, IEEE,* Ruibin Li*, *Student Member, IEEE,* Xiaofei Wang, Tarik Taleb, *Senior Member, IEEE,* Song Guo, *Fellow, IEEE,* Yuxia Sun, *Member, IEEE,* and Victor C. M. Leung *Life Fellow, IEEE*

***Abstract*—In recent years, the explosive growth of network traffic has placed significant strain on backbone networks. To alleviate the content access delay and reduce additional network resource consumption resulting from large-scale requests, edge caching has emerged as a promising technology. Despite capturing substantial attention from both academia and industry, most existing studies overlook the heterogeneity of the environment and the spatial-temporal characteristics of content popularity. As a result, the potential for edge caching remains largely unexploited. To address these challenges, we propose a neighborhood-aware caching (NAC) framework in this paper. The framework leverages the perimeter information from neighboring base stations (BSs) to model the edge caching problem in heterogeneous scenarios as a Markov Decision Process (MDP). To fully exploit the environmental information, we introduce an improved actor-critic method that integrates an attention mechanism into the neural network. The actor-network in our framework is responsible for making caching decisions based on local information, while the critic network evaluates and enhances the actor's performance. The multi-head attention layer in the critic network enables integration of environmental features into the model, reducing the limitations associated with local investigation. To facilitate comparison from an engineering perspective, we also propose a heuristic algorithm, Neighbor-Influence-Least-Frequently-Use (NILFU). Our extensive experiments demonstrate that the proposed NAC framework outperforms other baseline methods in terms of average delay, hit rate, and traffic offload ratio in heterogeneous scenarios. This highlights the effectiveness of the neighborhood-aware caching approach in enhancing the performance of edge caching systems in such scenarios.**

***Index Terms*—Edge Caching, Attention Mechanism, Actor-Critic Learning, Multi-agent Caching**

This work was supported in part by the National Science Foundation of China under Grant No. 62072332, China NSFC (Youth) through grant No. 62002260; the China Postdoctoral Science Foundation under Grant No. 2020M670654; the Haihe Lab of ITAI (Grant number 22HHXCJC00002); and the Technical Project Funding of State Grid Corporation of China under Contract No. 1400-202055132A-0-0-00; and the Chinese Government Scholarship (NO. 202006250167) awarded by China Scholarship Council. This research was also supported by fundings from the Key-Area Research and Development Program of Guangdong Province (No. 2021B0101400003), the Guangdong Basic and Applied Basic Research Foundation No. 2021A1515012297; Hong Kong RGC Research Impact Fund (No. R5060-19), Areas of Excellence Scheme (AoE/E-601/22-R), General Research Fund (No. 152203/20E, 152244/21E, 152169/22E), and Shenzhen Science and Technology Innovation Commission (JCYJ20200109142008673). This research work was also conducted in ICTFICIAL OY and is partially supported by the European Union's Horizon Europe program for Research and Innovation through the aerOS project under Grant No. 101069732. It was also partially supported by the Academy of Finland 6Genesis project under Grant No. 318927 and the Academy of Finland IDEA-MILL project under Grant No. 352428.

This work was presented in part at the IEEE International Conference on Communications (ICC), June 2021, Virtual / Montreal, Canada. (**Ruibin Li is the corresponding author.*)

C. Wang and X. Wang are the are with College of Intelligence and Computing, Tianjin University, Tianjin, 300072 China (e-mail: {chenyangwang, xiaofeiwang}@tju.edu.cn).

R. Li and S. Guo are with Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: ruibin.li@connect.polyu.hk, song.guo@polyu.edu.hk)

Yuxia Sun is with the Department of Computer Science, Jinan University, Guangzhou, China (e-mail: tyxsun@email.jnu.edu.cn).

T. Taleb is with the Information Technology and Electrical Engineering, University of Oulu, Oulu, 90570 Finland, and the Department of Computer and Information Security, Sejong University, Seoul, 05006 South Korea (e-mail: tarik.taleb@oulu.fi)

V. C. M. Leung is with the College of Computer Science & Software Engineering, Shenzhen University, Shenzhen, China 518060, and with the Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC, Canada V6T 1Z4 (e-mail: vleung@ieee.org).

I. Introduction

Recently, the development momentum of network technology is swift and skyrocketing, which leads to the deepening of digital transformation in all walks of life. According to Cisco's forecast, global data traffic will increase three times faster than it did four years ago [1]. The enormous data traffic has become a severe challenge for mobile network operators (MNOs), pushing the urgent revolutions of network architecture and high-level communication technologies (e.g., 5G). As a promising technology, mobile edge caching (MEC) is widely considered to palliate the traffic stress of MNOs [2]. By storing the diverse contents at edge nodes like base stations (BSs) or local devices approximating users, the data transmission delay can be reduced [3], thus improving the quality of the user experiences (QoE) and the quality of service (QoS).

Fig. 1 exhibits the architecture of heterogeneous edge networks (HetENets), which are densely designated to the edge joints (e.g., BSs, devices, and relays). The content cached in different BSs is exceedingly varied because the network operations at the locations are heterogeneous. For example, the social intentions of users who move from school to the hospital are completely different from those who are going to the living areas. The MEC technique can efficiently handle the high capacity demand for different species of node communications. Nevertheless, storing all the contents at HetENets is impossible and impractical. As a result, it is critical to envision and build a proper edge caching strategy to fully utilize network infrastructures [4].

Conventional caching replacement methods, such as Least Recently Used (LRU) and First Input First Output (FIFO), are based on static precepts, ignoring the vibrant intercommunication with context [5]. With the development of artificial intelligence (AI), more and more researchers have begun to

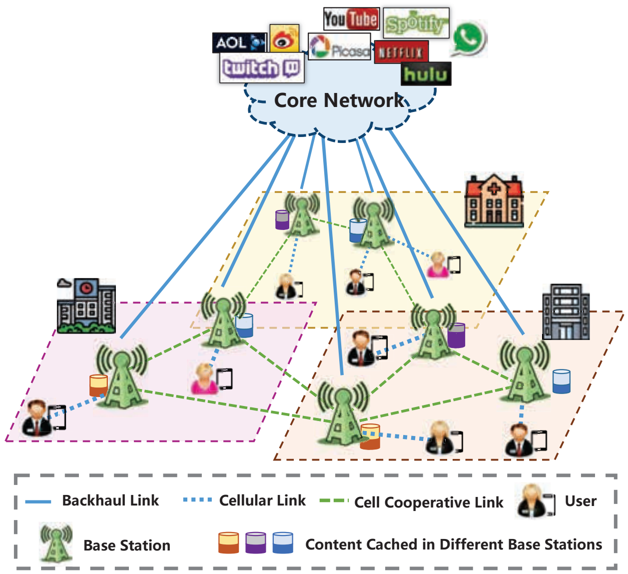


Fig. 1. Heterogeneous Edge Caching Architecture

use adaptive AI models to make edge caching strategies that can fully utilize the environmental awareness capacity of the network system [4]. Based on the previous discussions, there are several challenges that need to be considered as follows.

- **Dynamic Environment is Changing**: Existing studies are hard to make adjustment decisions during the information exchange of environmental interactions.
- **Contextual Information is Ignored**: Contextual information from neighbor BSs is always ignored, leading to low network performance when designing the caching strategy in the heterogeneous environment.

The attention mechanism is widely used to fix this gap since it is proficient in blending the elements according to their degree of influence [6]. By using the attention mechanism, the latent influences between the entities can be found. This helps reach the goal of coordination and balance from different points of view at the same time. Inspired by the MAAC algorithm proposed in the work [7], which selects appropriate information to estimate the execution effect of actors based on a multi-agent system by attention mechanism, we propose a multi-agent neighbor-aware actor-critic framework, nominated *NAC*. In this method, we consider not only the knowledge of the current area but also the global information of neighboring nodes. The actor-network in our framework is responsible for making caching decisions based on the local information, while the critic network is responsible for evaluating and correcting the actor's performance. The introduced multi-head attention layer in the critic network helps integrate the environment features into the model and eliminate the limitations caused by the local perspective. The main contributions of this paper are listed as follows:

- To address the problems of a dynamic environment and limited cache resources, we utilize a multi-head attention technique to effectively capture system changes and the status of neighbor BSs, therefore addressing the issue of adaptation and cooperation.
- We model the heterogeneous edge caching issue as the Markov Decision Process (MDP) and introduce the NAC framework for neighbor-aware actor-critic collaborative edge caching. In addition, we develop an engineering solution based on information by enhancing the classic Least-Frequently Used (LFU) algorithm for scenarios that cannot be integrated by neural network capabilities.
- Simulation results indicate that, compared to current baseline approaches, the proposed framework can effectively enhance hit rate, and decrease average transmission latency, and traffic offload ratio in a heterogeneous environment.

The rest of this paper is organized by the following: Section II introduces the related studies in the aspects of edge caching optimization and attention mechanisms. We specifically describe the goals of the system model and algorithm in Section III. The optimization of the edge caching strategy and the two proposed methods are introduced in Section IV. The simulation results and a case study are conducted in Section V. Finally, Section VI summarizes the full work.

II. Related Work

A. Edge Caching

Traditional caching replacement methods have made significant contributions in this field, such as Least-Recently-Used (LRU), Least Frequently Used Dynamic Aging (LFUDA) [8] and Greedy Dual Size Frequency (GDSF) [9]. Many studies sharpened edge caching by combining popular methods with probability. In [10], the authors envisaged proactive caching systems and distributed caching replacement processes based on the popularity of the content. However, this method neglected the reality of user variability and the high diversity of users' interests to a certain extent.

Furthermore, with the continuous efforts of researchers, artificial intelligent (AI) methods have been used to solve the problem of caching replacement and have played a milestone role. In deep learning (DL), the neural network was used to investigate user inclinations precisely and predict the request mode of BS [11]. Wang *et al.* paid attention to the design of the distributed caching strategy in [12], considered the cache capacity of the edge server layer, and proposed a distributed caching replacement method according to the popularity of the content to perform collaborative edge caching between BSs, and finally, minimize the transmission cost. Besides, deep reinforcement learning (DRL) was also a methodology that has made some achievements in caching policy optimization. It supported the network objects to learn and build knowledge and interact with the environment. In [13], the authors built an incentive mechanism, which was used for the distributed caching system and optimized via Deep Q-learning (DQN). In some studies, multi-agent learning centred on getting an overall reward [14], which designed a centralized critic network. However, a singular and particular aggregator leads to an additional transmission cost, and adaptability is also one of the challenges.

B. Attention Mechanism

In recent years, the attention mechanism was used for predicting the click-through rate (CTR) in e-commerce [15], [16] by discerning the influential characteristics of other entities to the current commodity or user. Especially in [17], Zhou *et al.*, authors investigated the distribution of influence, which was captured from the historical operation on the present products. Literatures of [18], [19] concentrated on finding the state's intrinsic composition and relationship in the complex environment of the multi-agents system.

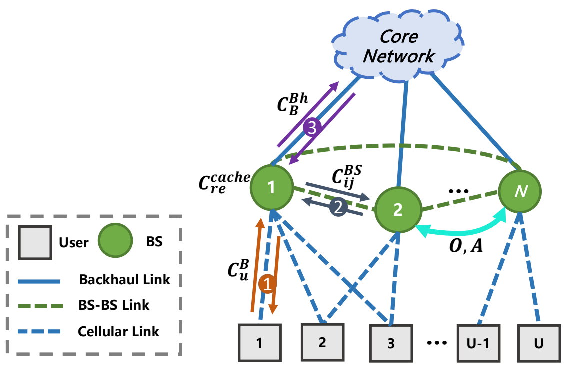


Fig. 2. Heterogeneous caching model.

There were many variants of the attention mechanism, such as hard attention [15], but this method did not pay attention to all the input content, but only paid attention to a specific aspect of information. Another variant was the multi-head attention [20], which aimed to allow the model to obtain more levels of information about the sentence from different representation spaces and improved the feature expression ability of the model. It has made contributions in the field of sentiment analysis and machine translation [21], [22]. To this end, we design the edge caching strategy by considering the multi-head attention mechanism for better extracting the underlying system information.

III. System Model

In this section, we introduce the models of the proposed framework, including the system architecture, user request model, and communication model. Then the objective problem is formulated.

A. System Overview

Fig. 2 shows the heterogeneous caching model consisting of various users, BSs and a core network. Let $\mathcal{B} = \{B_1, B_2, ..., B_b\}$ denote the BSs, and the users are randomly distributed in the coverage area of the BS B_b, which can be donated as $\mathcal{U}_b = \{U_1, U_2, ..., U_b\}$, where $U_b = \{1, 2, \cdots, u\}$. The union of C_b is a set of cached contents under all BSs denoted as $\mathcal{C} = \{C_1, C_2, ..., C_b\}$, and $\mathcal{N} = \{n_1, n_2, ..., n_b\}$ to signify the storage capacity of BSs. The backhaul link is the bridge between the core network and each BS. We consider a content pool $\mathcal{F} = \{f_1, f_2, ..., f_F\}$ in the core network, which stores all of the contents. Let $req(u_b)$ express the requested content sent by user u to BS B_b via the cellular links. We only consider the cost that the request of adjoining BS is in the last hop (i.e., a two-hop requested scenario). If the request is not available in the neighbor BS B_b[1]. The request will be sent to the cloud centre through backhaul if the content cannot be satisfied by its neighbor.

As mentioned in the above section, the network is strengthened by arranging BSs in buildings, such as shopping malls, schools, hospitals, etc., or on outdoor facilities (e.g., parks and traffic lights.) In real life, these scenes are close to each other in various combinations, and the distribution of these BSs is also close and intersects. In the proposed caching framework, the communication from a BS to others is composed of the observation value and action of each BS. To describe the problem clearly in the modelling, we only consider the transfer between neighbors of one hop in the part of communication between BSs. Observation o_b and action a_b of BS B_b thereafter are transferred between it and the one-hop neighboring BSs of it.

B. User Request Model

As verified in [23], Mandelbrot-Zipf (MZipf) is usually used to model the distribution of content popularity as a basis of request sending and transmission in the caching process. In our model, the generation of user request content is also derived from MZipf distribution. Motivated by this, the probability that the content f is requested by users in BS B_b is defined by:

$$\omega_{fb} = \frac{(R_f + q)^{-\alpha}}{\sum_{i \in \mathcal{F}} (R_i + q)^{-\alpha}}, \tag{1}$$

where R_f represents the popularity rank of content f in descending order in local. Among the parameters, α is the skewness factor and q donates the plateau factor.

C. Communication Model

The communication process mainly includes three processes, as shown in Fig. 2. Process ① indicates the communication between the local BS and user which is established by the cellular links. If the local BS B_i stores the requested content f, it can provide the quick content access service for user u, thus, the cost of this process is represented as $C_{uf}^{B_i}$. However, it is impractical to put all the content in the cache of the local BS B_i, the request will be sent to another neighbor BS. This process is illustrated by ②, and the transmission cost C_{ij}^{BS} between the current BS B_i and the neighbor BS B_j is measured by the distance [24], expressed as $C_{ij}^{BS} = \nu d(B_i, B_j), i, j \in \{1, 2, ..., b\}$, where the non-negative constant ν is the cost coefficient, and $d(B_i, B_j)$ is the network distance between the two BSs B_i and B_j.

If the neighbor BSs cannot provide the service for the required content through the process ③, i.e., the backhaul link, the indicated request will be sent to the core network. According to the experience of existing research [25], we use a positive parameter μ to represent the unit transmission cost in the backhaul link. Related to formula (1), the request of

[1]We consider the BSs are connected by fibres in the networks.

the user for content f is not available[2] in BS B_b and the probability of the request to the core network is:

$$\omega_{f_{Bh}} = 1 - \omega_{fb}. \tag{2}$$

The transmission cost of backhaul is:

$$C_{Bf}^{Bh} = \mu\omega_{f_{Bh}}. \tag{3}$$

The main costs incurred during the entire content request process include not only transmission costs. It also includes the caching replacement cost, which is closely related to the size of the content [26]. The overhead incurred during the caching replacement is defined as follows:

$$C_{rc}^{cache} = \delta(c_r - c_c), r, c \in \{1, 2, ..., N\}, \tag{4}$$

where the content replacement factor $\delta > 0$, c_c and c_r respectively represent the size of two contents before and after the replacement.

TABLE I
NOTATION VALUES

Notation	Meaning
$\mathcal{B}$	Set of BSs
$n_i \in \mathcal{N}$	Storage capacity of BS B_i
$\mathcal{F}$	Content pool
U_b	User set under the coverage of BS B_b
C_b	Cached content set of BS B_b
$req(u_b)$	requested content sent by user u to BS B_b
S_b	State space of BS B_b
A_b	Action space of BS B_b
o_b	Current observation state of BS B_b
a_b	Current action made by AC agent
ω_{fb}	Probability that the content f is requested
α, q	skewness factor, plateau factor
$C_{uf}^{B_i}$	Cost when user's request satisfied by local BS B_i
C_{ij}^{BS}	Cost when fetch content from neighbor BS
C_{Bf}^{Bh}	Cost when fetch content from Core network
r_i^t	Reward which used for training RL model
θ_i	Actor's network parameter
d	Numbers of feature
e	The embedding feature used for critic
$\mathcal{Q}$	Query embedding for different feature
$k_{d,j}$	Key value for calculating attention weight
$\zeta_{d,j}$	Calculated attention weight
$v_{d,j}$	Value that contain caching information
ψ_b	Parameter of BS B_bś target critic network
γ	Decay factor of RL model

D. Problem Formulation

In this part, we introduce the goals which are obtained based on the details and limitations of the modeling described in the past section. The objective of the proposed framework is to keep the costs of the system as low as possible. The system saved costs can be obtained as follows.

We use G to represent the revenue of the entire system:

$$G = \sum_{B_i \in \mathcal{B}} \sum_{u \in U_i} (x_{req(u)i} C_{uf}^{B_i} \\ +(1 - x_{req(u)i})\Delta). \tag{5}$$

[2]This may be caused by no content f cached in BS B_b or the link congestion, etc.

The Δ in the formula is expressed as the following:

$$\Delta = \begin{cases} \dfrac{\sum_{j \neq i}^{|\mathcal{B}|} (x_{req(u)j}(C_{Bf}^{Bh} - C_{ij}^{BS} - C_{req(u)f}^{cache}))}{\sum_{j \neq i}^{|\mathcal{B}|} x_{req(u)j}}, & \sum_{j \neq i}^{|\mathcal{B}|} x_{req(u)j} \neq 0 \\ 0, & \sum_{j \neq i}^{|\mathcal{B}|} x_{req(u)j} = 0 \end{cases}. \tag{6}$$

The parameter $req(u)$ means the content requested by the user u, $x_{req(u)i}$ represents whether the content requested by u is stored in the BS B_i, and $x_{req(u)i} = 1$ means that the content f has been cached, and there are no replacement actions required in this case. If $x_{req(u)i} = 0$, the modified content needs to be satisfied through other processes, e.g., directly from neighbor BSs or the core network.

Maximizing the reduction of the transmission and caching cost is the goal of our work:

$$max \sum_{B_i \in \mathcal{B}} \sum_{u \in U_i} (x_{req(u)i} C_{uf}^{B_i} \tag{7}$$
$$+ (1 - x_{req(u)i})\Delta) \tag{8}$$
$$s.t. \sum_{c_n \in C_i} c_n \leq n_i, \tag{8a}$$
$$x_{req(u)i} \in \{0, 1\}, \tag{8b}$$
$$d(B_i, B_j) > 0, \tag{8c}$$
$$\nu, \mu, \delta \geq 0. \tag{8d}$$

The constraint shown in (8a) indicates that the sum of the size of the cached content is limited by the storage capacity of the BS. Furthermore, (8b) describes the parameter table of the token replacement behavior, (8c) and (8d) ensure the positive value of the distance between BSs, BS-BS cost coefficient, backhaul unit transmission cost, and replacement factor.

IV. ATTENTION-BASED MULTI-AGENT CACHING REPLACEMENT STRATEGY

Each BS has its own service area, in which users have different preferences and relationships with others. Meanwhile, BSs communicate with each other and cooperate. To achieve an efficient caching strategy, we use the influence and information between BSs to a greater extent. The actor-critic algorithm is an effective combination of strategy-based and value-based methods. We propose a multi-agent framework based on actor-critic to solve the edge caching problem.

A. Multi-Agent Actor-Critic

In our architecture, each BS elaborates one actor-network and one critic-network, where different users have their own interests and preferences. Meanwhile, due to the difference in geographic spaces and social functions as well as the heterogeneous services, BSs may have diverse influences on the neighbors. As shown in Fig. 3, for the BS B_1 in the blue area, the BS B_2 in the dark orange area has more influence on its caching decision compared with B_b in the light orange area. Agents can also learn from other BSs which help make the intelligent caching decision. In this way, we assume that the

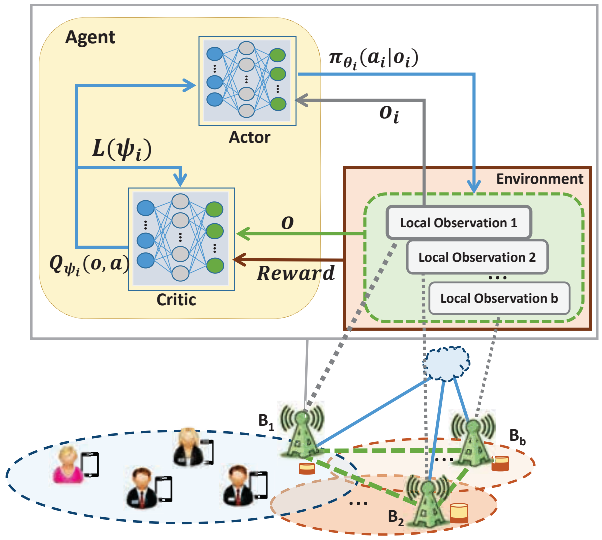


Fig. 3. Illustration of actor-critic mechanism in each BS.

caching state can be passed among BSs for better information (e.g., the size/type of cached content) interactions.

State Space: We use $S = \{S_1, S_2, ..., S_b\}$ to describe the state space. The state space of each agent is made up of the cached list state and the information of the cached content, for example, the size and the type of the cached content.

Action Space: Let $A = \{A_1, A_2, ...A_b\}$ represent the action space of all BSs, where A_i means the action space of B_i. The size of the action space of each BS is equal to the amount of content currently cached plus one as the agent can also choose the action that does not replace existing content in the current caching list.

Reward: Based on the definition system model, we design the system reward as:

$$r_i^t = \begin{cases} C_{Bf}^{Bh} - C_{ij}^{BS} - C_{rc}^{cache} \\ C_{Bf}^{Bh} \end{cases} , \tag{9}$$

where the above expression represents the reward of receiving the requested content from adjacent nodes, and the other one is the reward of receiving it from the current BS.

Observation: Let the $o = \{o_1, o_2, ..., o_b\}$ denote the observation, agents can obtain the local caching state and the user's requests under the coverage of its serving area. Besides, as different BSs can establish communication links, agents can also obtain some information about neighbor BSs as a result. In the next subsection, we will introduce the specific temporal and spatial factors involved in the observation.

Actor Network: Each local agent observes its serving area's local state and then makes the caching decision based on its local state. In the actor-network, the optional actions include all currently cached contents and choose action mainly based on its local observation, the policy gradient therein is used to update the parameter. We use θ to represent the content replacement parameter:

$$a_i = \pi_{\theta_i}(o_i) \ , \tag{10}$$

where o_i means the local caching states of the BS B_i.

Critic Network: The decisions of caching actions are selected by the actor-network deployed in each agent based on local information. After the process of actions selection in the actor-network, the critic-network evaluates the expected reward of the policy according to the observation and actions of all BSs through the value function. Furthermore, an attention mechanism is adopted to model the complicated impact of other agents, and we describe our specific method in the following sections.

B. Critic Network with Attention Mechanism

We use time and space factors as observations, and mine the influence of BSs based on the critic network with the attention mechanism. As an agent, each BS has its own critic network and can observe the history caching status of its nearby BSs. The whole process can be regarded as a combination of distributed local training and centralized global learning. In order for the critic to learn globally from the information of neighboring BSs, as shown in Fig. 4, we set up three modules of embedding layer, attention layer, and output layer in the critic network.

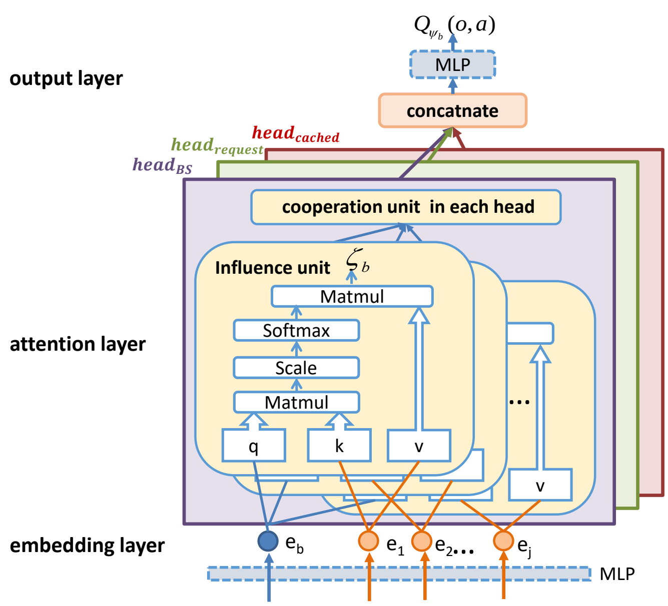


Fig. 4. Design of the attention-based critic network.

Embedding Layer: The obtained numerical features will be fed into the multilayer perceptron (MLP) in this layer. Because different BSs provide users with a variety of services in different scenarios, resulting in the uncertain number of services provided by each BS. This results in the length of the corresponding encoding vector list is not a fixed value.

As described in the previous subsection, we use one-hot or multi-hot encoding operations to express the types. We know that the one-hot feature is too sparse, and even the encoding of the caching category with multi-hot encoding may be a very sparse vector, which is not suitable for direct input into the subsequent neural network for training. To this end, it is necessary to convert this sparse one-hot vector into a denser embedding vector by connecting it to the embedding layer.

The embedding vector e_{bi} of i-th content cached in the BS B_b is obtained through the embedding operation. Observation of BS B_b can be embedded and transformed into embedding vector e_b, and e_j is defined as the observation of the adjacent BS B_j linked to the current BS via the BS-BS link. Due to the different caching capabilities of BSs and the different sizes of content, the amount of content in each BS may be different, resulting in different lengths of feature vectors. However, a fully connected network can only handle fixed-length input. To solve this problem, the pooling method is generally adopted to map the original features to a fixed-length vector and ensures the immutability of features. In this way, we have the pooling vector $e = pooling(e_1, ..., e_j)$ of the neighboring BSs.

Attention Layer: The multi-head attention mechanism is deployed in this layer, which uses multiple queries $\mathcal{Q} = \{q_1, .., q_d\}$, where d means the number of kinds of features, to calculate in parallel to select multiple information from the input information, and each attention focuses on a different part of the input information. The different feature subspaces are shown in Table. II. The cached content, request content, and BS feature in the table correspond to $head_{cached}$, $head_{request}$, and $head_{BS}$, respectively. The requested feature is similar to the feature of the cached content and will not be repeated. Our critic is not centralized, in each head the query is a kind of character in the current BS. As described in the previous subsection, the head includes the size, type, and request frequency of the cached content, caching capacity, and the users who are under the service of the local BSs. Key $k_{d,j}$ and value $v_{d,j}$ are the information of the adjacent BS B_j on the feature d. After completing the process of information input, we get the attention to aspect d of adjacent node j, which is named the influence unit $\zeta_{d,j}$:

TABLE II
FEATURE SUBSPACES CORRESPOND TO MULTI-HEADS.

feature subspaces	feature		data collection
cached content	content size		numeric
	content type		one-hot
	frequency	long-term	numeric
		mid-term	numeric
		short-term	numeric
request feature	content size		numeric
	content type		one-hot
	frequency	long-term	numeric
		mid-term	numeric
		short-term	numeric
BS features	storage capacity		numeric
	type of cached content		multi-hot

$$\zeta_{d,j} = \frac{q_{d,j}^T k_{d,j}}{\sum_{k \neq i} \exp(q_{d,j}^T k_{d,k})}, \tag{11}$$

where d is used to mark the features observed by the current head.

After getting the influence unit $\zeta_{d,j}$, we calculate cooperation unit $Att_{d,b}$ through the weighted sum of attention weight $\zeta_{d,j}$ and value $v_{d,j}$. In this way, we can obtain the $k_{d,j}$ and $v_{d,j}$ through different linear embedding layers where $k_{d,j}$ is used to calculate weights and $v_{d,j}$ is used to capture features. $Att_{d,b}$ is defined as:

$$Att_{d,b} = \sum_{j \neq b} \zeta_{d,j} v_{d,j} \ . \tag{12}$$

Output Layer: Based on the attention operation of the previous layer, at this stage, we concatenate these weights in columns, and then perform a linear transformation with a new weight matrix W^a to get the final attention output. The concatenate function is used to aggregate the attention of multi-heads:

$$Att_b = concat(Att_{1,b}, ..., Att_{d,b}) \ . \tag{13}$$

The critic network of each BS estimates the action-value function by attention mechanism which can be expressed by:

$$Q_{\psi_b}(o, a) = \sigma_b(e_b, Att_b) \ , \tag{14}$$

where ψ is the parameter of the critic network, σ_b is an MLP layer, and the ψ_b is the parameter of the target critic network which is associated with the agent B_b.

C. The Algorithm process

Reinforcement learning (RL) is an iterative process. Each iteration needs to be given a strategy evaluation function and update the strategy according to the value function. There are few studies on the convergence of actor-critic, the main reason is that the update has a strong network and status relevance. Therefore, in addition to the original ac network, a target ac network is also established. The target network has the same structure and initialization as the original network. While training parameters of the networks, the target network estimates future actions.

In the RL scheme, the objective is to find the optimal policy. To this end, the Q-function is used to represent the expectation of the total reward that the agent can obtain in the future after taking action a in state s:

$$Q_i(o, a) = \mathbb{E}_\pi[\sum_{t=0}^{+\infty} \gamma^t r_{t+1} | o_0 = o, a_0 = a] \ . \tag{15}$$

To get the optimal Q-value, the input corresponding state and all possible actions are required to be traversed and then find the largest Q-value. We use equation (14) to approximate equation (15).

The attentive critic network calculates the advantage function to get the advantage of the current action compared to other actions [7], and the advantage function is applied to the gradient update process of individual strategy as follows:

$$A_i^a(o, a) = Q_{\psi_i}(o, a) - \sum_{a_i' \in A_i} \pi_{\theta_i}(a_i' | o_i) Q_{\psi_i}(o, (a_i', a_i)) \ . \tag{16}$$

This additional item is a baseline that considers an additional benefit in a multi-environment interaction, and it does not change the decision-making process of the agent. The calculation of the updated gradient of our policy is shown as:

$$\nabla_{\theta_i} J(\pi_{\theta_i}) = \mathbb{E}_{\pi_{\theta_i}}[\nabla_\theta log(\pi_{\theta_i}(a_i | o_i)) A_i(o, a, \psi_i)] \ . \tag{17}$$

The update of the critic network is based on the following loss function:

$$L(\psi_i) = \mathbb{E}[(y_i - Q_{\psi_i}(o, a))^2]. \tag{18}$$

In the above equation, the y_i is defined as:

$$y_i = r_i + \gamma \mathbb{E}_{\pi_{\theta_i}}[Q_i^{\overline{\psi}}(o', a') - \rho log(\pi_{\overline{\theta}}(a_i'|o_i'))], \tag{19}$$

where γ is the decay factor, $\overline{\psi_i}$ and $\overline{\theta_i}$ are the past parameters before update, ρ determines the balance between maximizing entry and rewards [27].

The parameter $\bar{\psi}$ of critic network and $\bar{\theta}$ of actor-network are updated after each iteration:

$$\bar{\psi} = \psi - lr_c \nabla L(\psi_i) \ , \tag{20}$$

$$\bar{\theta} = \theta - lr_a \frac{\nabla_{\theta_i} J(\pi_{\theta_i})}{\nabla \theta_i} \ , \tag{21}$$

where lr is the learning rate, which is set in the experiment according to the actual situation.

Note that the above two equations are used to increase the randomness of our strategy which can decentralize the probability of each action, in case the agent just chooses one action. The whole process is as shown in Algorithm 1. Like in [4], the time complexity of back-propagation is $\mathcal{O}(abN^2)$, in which a and b are the number of layers and unit, respectively.

Algorithm 1 The procedure for caching by NAC

Initialization:
  Parameter of each actor network θ_i
  Parameter of each critic network ψ_i
  Reset the parallel environments with B_b agents
$T_{update} \leftarrow 0$
for $t = 1, T$ **do**
  each BS receives requests R_t=$\{r_1, r_2, ..., r_b\}$
  obtain BSs' observations $o_t = \{o_1, o_2, ..., o_b\}$
  each BS selects by the policy $\pi_{\theta_b}(a_b|o_b)$
  each BS sends action and state to adjacent BSs
  $T_{update} = T_{update} + 1$
  if $T_{update} \geq$ min steps per update **then**
    calculate $Q_{\psi_i}(o_{1...b}, a_{1...b})$ for all BSs
    calculate $a_i' = \pi_i^{\overline{\theta_i}}(o_i')$ using target policies
    calculate $Q_{\overline{\psi_i}}(o_{1...b}', a_{1...b}')$ for all BSs
    update critic using $\nabla L_Q(\psi_i)$
    calculate $a_i = \pi_i^{\overline{\theta_i}}(o_i')$ for all BSs
    calculate $A_b^a(o, a)$ for all BSs
    update policies using $\nabla_{\theta_i} J(\pi_{\theta_i})$
    update the parameters for all BSs
    update state of each BS
    $T_{update} \leftarrow 0$
  end if
end for

D. An Information-based Engineering Solution

Considering the deployment cost of neural networks embedded in all scenarios, inspired by the core idea of the attention mechanism, we abstract the information of the **N**eighbors' **I**nfluence (NI) and design a heuristic method by improving the traditional LFU, named NI-LFU, from the engineering perspective.

Based on the temporal locality of traffic, each content that has been cached is assigned a value $\mathcal{F}(f, b) = \sum_{j=1}^{k} m_f^k, j \in \{1, 2, \ldots, k\}$, where m_f^k means if the current BS meets the requested content f, and k is the number of requests since the content f was cached. Taking into account the impact of multi-granularity unit time and surrounding neighbor nodes, we change the value as:

$$\mathcal{F}_{NI}(f, b) = \frac{1}{B_b} \sum_{i=0}^{B_b} (\mathcal{F}_m(f, i) + \mathcal{F}_w(f, i) + \mathcal{F}_d(f, i)) \ , \tag{22}$$

where $\mathcal{F}_m(f, i), \mathcal{F}_w(f, i), \mathcal{F}_d(f, i))$ indicates the request frequency in the different three modes under the service range of the BS B_i respectively. The long-term frequency of request content f is defined as its requested times in a month $\mathcal{F}_m(f)$, the mid-term frequency is its requested times in a week $\mathcal{F}_w(f)$, and the short-term frequency is represented by the content request frequency in a day $\mathcal{F}_d(f)$. Let $\mathcal{F}_{NI}(f, b)$ denote the elimination priority on which content will be replaced. The workflow of the NI-LFU algorithm is shown in Fig.5.

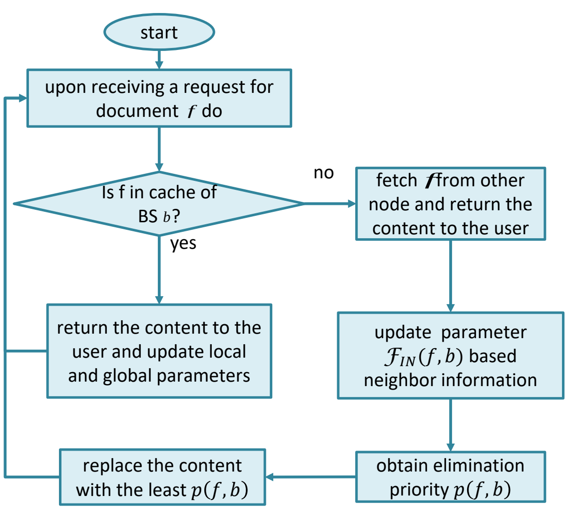


Fig. 5. The workflow of NI-LFU algorithm.

V. Case Study and Experiment

In this section, we design a small scratch for the adaptability of heterogeneous environments and evaluate the performance of the proposed framework based on the simulation results.

A. Simulation Configuration

In order to reflect the feasibility of the theory, we designed a small-scale simulation experiment, and use the stochastic

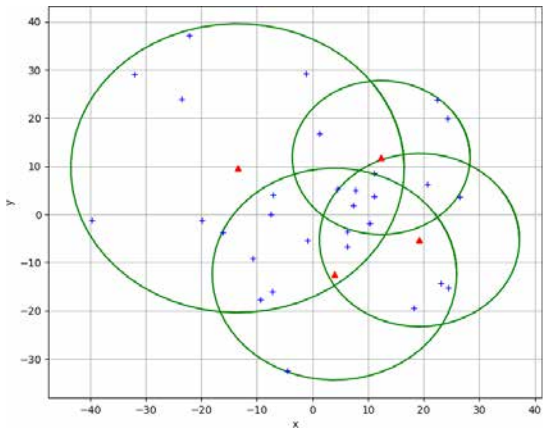


Fig. 6. Simulation environment generated by stochastic geometry.

geometry [28] to model the potential correlation between BSs and the service area of BSs in a multi-agents environment. Stochastic geometry has proven to be a powerful tool for evaluating wireless network performance. Fig. 6 simulates the relationship between BSs, where the red triangles indicate BSs and the blue stars are users.

The total number of contents is $F = 10000$, and different contents have a normalized size of different values which range between 1MB and 10MB. In order to conduct the case study experiments, we set the total number of BSs as 4, and the cache size of each BS is set to 100MB as default. Users are distributed in the serving range of each BS randomly. In addition, we use a parameterized network to solve the problem, and we confirm some of the parameters (i.e., the learning rate of actor-network lr_a, the learning rate of critic lr_c) involved in the experiment.

We compare our methods with the following state-of-the-art baseline algorithms:

*1) **Actor-critic (AC)**:* A TD method that has a separate memory structure to explicitly represent the policy independent of the value function [29].

*2) **Deep Q Networks (DQN)**:* In addition to the difference in network structure, the method differs from the proposed framework in that it is based on local characteristics and does not add influencing factors in the environment of neighbor nodes. DQN and AC are mainly used to compare with NAC to verify the stability of the NAC algorithm [30].

*3) **First-Input-First-Output (FIFO)**:* FIFO is one of the oldest caching methods. This algorithm always deletes the content with the longest time [5].

*4) **Least-Frequently-Used (LFU)**:* In the LFU algorithm, the least frequently used content is knocked out [31].

*5) **Least-Recently-Used (LRU)**:* LRU is a common caching replacement algorithm that eliminates the most recently unused content [5].

*6) **Least-Frequently-Used-Dynamic-Aging (LFUDA)**:* LFUDA builds upon simple LFU by accommodating shifts in the set of popular objects in the cache [31].

*7) **Greedy-Dual-Size-Frequency (GDSF)**:* A strategy based on the value relationship of all cache objects in the cache node [9].

*8) **Neighbor-Influence-Least-Frequently-Used (NILFU)**:* From the perspective of information, NI-LFU is an engineering solution formed by the idea of our approach. It combines the request frequency in the current area and the neighbor environment via multiple angles to make replacement decisions.

We take the following indicators to evaluate the performance of different methods:

Hit Rate: it is the ratio of hits and total requests. Hit means that the user can get the required data directly through the cache. At the same time, the performance index we use when conducting parameter selection experiments is also a hit rate. **Average Transmission Delay:** it is calculated based on the delay between the cloud data centre and the BS, between BSs, and between the user and BS. **Traffic Offload Ratio:** it means the rate of BS offloaded traffic to the total traffic.

B. Case Study

A key challenge in highly heterogeneous services is making full use of complex spatial and temporal characteristics [32]. We designed this caching replacement optimization strategy with the attention mechanism and used artificial intelligence technology to perceive the environment of neighbor BSs.

To prove the practicability of the proposed framework in a heterogeneous environment, we design a small demo to compare the algorithm's performance in a heterogeneous environment and a homogeneous environment.

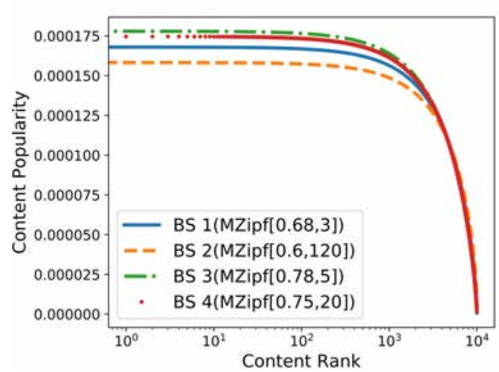
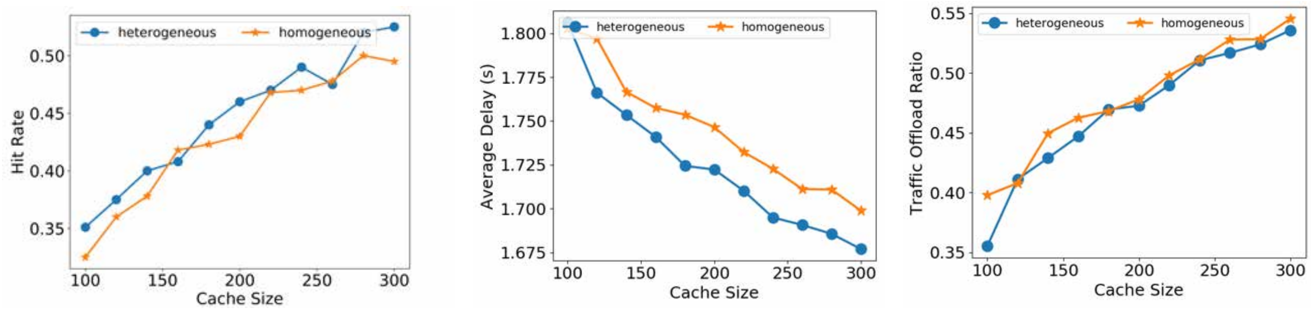


Fig. 7. Distributions of Content popularity for 4 BSs.

In this case, as expressed in Fig. 6, we consider 4 BSs, and the storage size of BSs is [300, 340, 380, 420]. The user number of each BS is [10, 7, 5, 8], and the initial state of the user is randomly generated. At the same time as agents, their training parameters are also different, the experience pool sizes of these agents are [2000, 2500, 1500, 2000], the episode numbers of BSs are set as [200, 150, 250, 150], the batch sizes are [128, 128, 256, 256].

We have the following two cases, i.e., heterogeneous and homogeneous cases. (1) For the heterogeneous case, we simulate heterogeneity of different services by setting the MZipf factor and plateau factors in equation (1), the popularity distributions are shown as dotted lines in Fig. 7. (2) For the homogeneous case, the content popularity follows the same MZipf, shown



(a) Hit rate (b) Average delay (c) Traffic offload ratio

Fig. 8. Performance demonstration of hit rate, average delay, and traffic offload ratio with heterogeneous services and homogeneous services

as the solid line in Fig.7, we set the value of q and α based according to the reference [33] which originally proposed the Mandelbrot–Zipf model for content popularity.

In the homogeneous environment, the agent can easily obtain the network information while the spatial and temporal characteristics in heterogeneous are dynamic and more complex. As shown in Fig. 8(a) and Fig. 8(c), the hit rate and traffic offload ratio of the two kinds of environments is relatively close. It indicates that the proposed NAC can achieve similar performance in terms of hit rate and traffic offload ratio. This is mainly because the proposed framework learns the information from the neighbor BSs which improves the network performance. Besides, the curves interact several times due to the dynamic environment changes. Especially in Fig. 8(b), when the cache size is 180, the average delay reduces by 2.1% at best in the heterogeneous situation compared with the homogeneous one. This shows our method has a great capacity for neighborhood awareness and adaptability in heterogeneous scenarios.

C. Parameters Settings

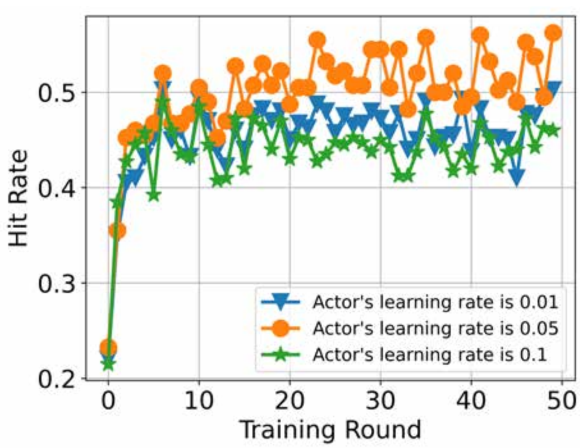


Fig. 9. Performance of the hit rate under different learning rates of actor-network.

Before the performance comparison simulation, we first compare and select the experimental parameters of our method. We use the training round as the abscissa to show the effect of the different parameters, Fig.9 compares the performance with different learning rates lr_a of the actor-network. The comparative learning rate is $lr_a = [0.01, 0.05, 0, 1]$. To maintain the algorithm's stability and efficiency, we defined the learning rate $lr_a = 0.05$ as an empirical value.

We demonstrate the network performance of the hit rate in the following Fig.10. We compare the performance for hit rate under different learning rates of critic network $lr_c = [0.01, 0.05, 0.1]$. We can see that if $lr_c = 0.05$, the proposed framework produces the best hitting effect. Therefore, $lr_c = 0.05$ is used as an algorithm parameter of the NAC in our simulation.

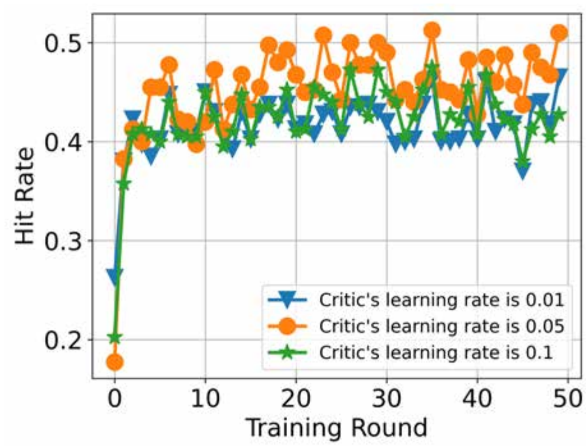


Fig. 10. Performance of the hit rate under different learning rates of critic network.

D. Comparison of Different Training Methods

We compared *NAC*, AC, DQN, FIFO, LFU, LRU, LFUDA, GDSF, and NILFU under different BSs' buffer sizes. The corresponding simulation results are shown in Fig. 11-Fig. 13.

Fig. 11 shows the performance of the hit rate. The hit rate of NACE is obviously the highest. Compared with several commonly used traditional methods, it has increased by 26.81% (FIFO), 26.14% (LFU), 29.04% (LRU), 22.62% (LFUDA), 20.45% (GDSF), 11.23% (AC) and 27.43% (DQN) under different parameters. Moreover, although the NILFU we designed is not based on machine learning, after adding the influence of neighboring cells, compared with other traditional

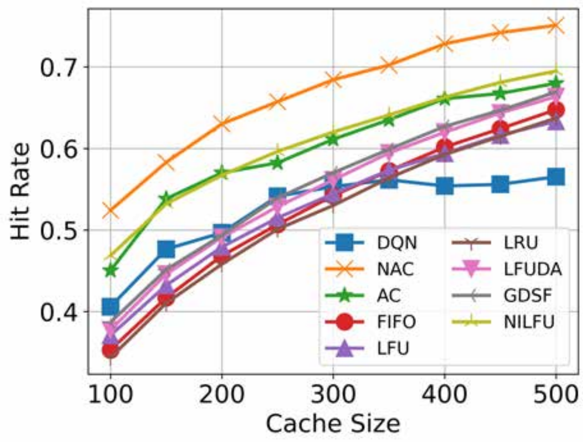


Fig. 11. Performance demonstration of hit rate with a different algorithm.

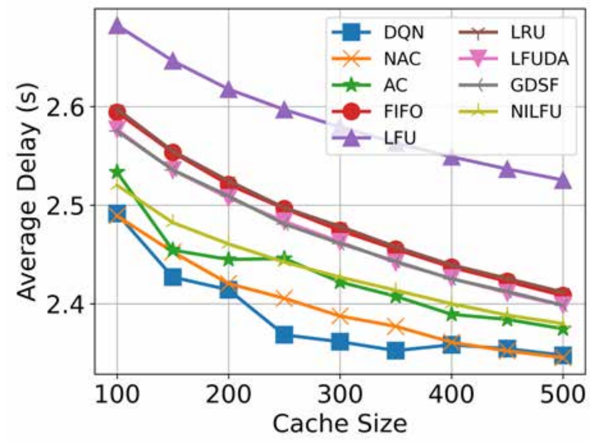


Fig. 12. Performance demonstration of average delay with different algorithms.

non-learning caching replacement algorithms, the advantage hit rate of it is also obvious. Compared with AC, DQN, FIFO, LFU, LRU, LFUDA, and GDSF, the hit rate increased by 1.26%, 16.00%, 15.44%, 14.83%, 17.47%, 11.31% and 9.65%, respectively.

As shown in Fig. 12, compared with several traditional replacement methods, NAC can effectively reduce the delay by 0.030s (AC), 0.086s (FIFO), 0.190s (LFU), 0.086s (LRU), 0.072s (LFUDA), 0.072s (GDSF). Although NAC's performance in latency is slightly inferior to DQN, it is better in terms of hit rate and traffic unloading. And from the figure, we can see that compared to DQN, as the cache capacity increases, the performance of NAC will improve. As an algorithm based on a value function, DQN itself is not easy to find the greatest Q-value among many values when facing a larger action space. With the enhancement of the caching capacity, the action space becomes larger. This shortcoming is also exposed, and on the other hand, it also illustrates the advantages of NAC. And the NILFU average delay performance is excellent. As shown in Fig. 12, the average delay of NILFU is reduced by 0.050s, 0.154s, 0.053s, 0.037s, and 0.037s compared to FIFO, LFU, LRU, LFUDA, and GDSF, evenly and respectively. When the BS cache capacity is large to some degree, NILFU can be equal to or even surpass the AC algorithm. This also fully proves that NILFU not only uses neighborhood information to improve accuracy but also can overcome some of the disadvantages of complex algorithms.

At the same time, as depicted in Fig. 13, NAC and NILFU also show good advantages in offload. From the figures, we can see that with the enhancement of the cache capacity, the traditional AC does not have a higher performance improvement than NAC. With the enhancement of caching capabilities, the content cached by BSs has a higher diversity, only considering local information, so AC cannot make a good global decision. Agents do not consider the dependency of the surrounding node cache, so its performance is relatively poor. It can be seen from the resulting graph that the broken line of the neighborhood-aware method will be more smooth and more stable. But it will have relatively large fluctuations in the traditional AC method. This shows to a certain extent that

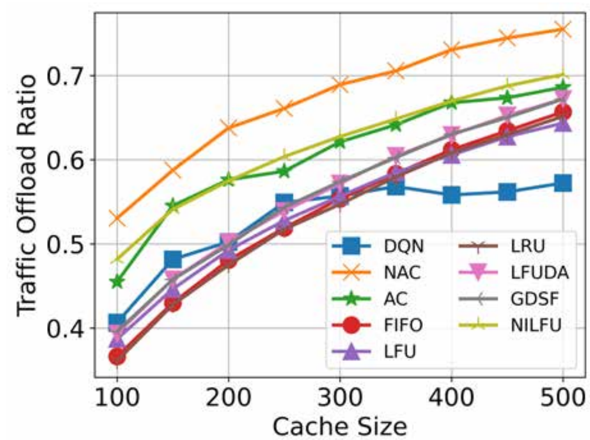


Fig. 13. Performance demonstration of traffic offload ratio with different algorithms.

considering the characteristics and help of the surrounding area can increase the stability of the entire model.

VI. Conclusion

In this paper, we have proposed an edge caching framework called *NAC* by considering the neighborhood-aware information, where the Actor-Critic method is employed to solve the complicated edge caching problem. The AC agent can make efficient caching decisions after gathering all the global and local information, leading to excellent backhaul traffic reduction and QoS/QoE improvement. Each AC agent contains two networks: the actor-network chooses actions according to the local information. The critic network evaluates the impact of surrounding BSs. What is more, to integrate the information transmitted between BSs, we have employed a multi-head attention mechanism to quantify the impact of neighboring BSs on the current node with each head capturing the characteristics under a subspace. Besides, we also propose a non-learning caching algorithm NILFU, which can achieve considerable performance in some scenarios without the use of AI services. In the section of experiments, the proposed *NAC* algorithm shows the feasibility and effectiveness compared to several existing caching algorithms. In the future, We will continue to improve our algorithm and architecture based on the actual situation.

REFERENCES

[1] G. M. D. T. Forecast, "Cisco visual networking index: global mobile data traffic forecast update, 2017–2022," vol. 2017, p. 2022, 2019.

[2] X. Wang, R. Li, C. Wang, X. Li, T. Taleb, and V. C. Leung, "Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 154–169, 2020.

[3] Z. Cai and Q. Chen, "Latency-and-coverage aware data aggregation scheduling for multihop battery-free wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1770–1784, 2021.

[4] X. Wang, C. Wang, X. Li, V. C. Leung, and T. Taleb, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE IoT Journal*, 2020.

[5] A. Dan and D. Towsley, "An approximate analysis of the lru and fifo buffer replacement schemes," in *SIGMETRICS*, 1990, pp. 143–152.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning." PMLR, 2019, pp. 2961–2970.

[8] B. Jia, R. Li, C. Wang, C. Qiu, and X. Wang, "Cluster-based content caching driven by popularity prediction," *CCF Transactions on High Performance Computing*, pp. 1–10, 2022.

[9] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin, "Evaluating content management techniques for web proxy caches," *ACM SIGMETRICS Performance Evaluation Review*, vol. 27, no. 4, pp. 3–11, 2000.

[10] S. Tuli, G. Casale, and N. R. Jennings, "Pregan: Preemptive migration prediction network for proactive fault-tolerant edge computing," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 670–679.

[11] S. Shekhar, A. Singh, and A. K. Gupta, "A deep neural network (dnn) approach for recommendation systems," in *Advances in Computational Intelligence and Communication Technology*. Springer, 2022, pp. 385–396.

[12] H. Tian, X. Xu, T. Lin, Y. Cheng, C. Qian, L. Ren, and M. Bilal, "Dima: Distributed cooperative microservice caching for internet of things in edge computing by deep reinforcement learning," *World Wide Web*, vol. 25, no. 5, pp. 1769–1792, 2022.

[13] R. Zhang, F. R. Yu, J. Liu, T. Huang, and Y. Liu, "Deep reinforcement learning (drl)-based device-to-device (d2d) caching with blockchain and mobile edge computing," *IEEE TWC*, 2020.

[14] C. Zhong, M. C. Gursoy, and S. Velipasalar, "Deep multi-agent reinforcement learning based cooperative edge caching in wireless networks," in *IEEE ICC*, 2019, pp. 1–6.

[15] U. Ahmed, G. Srivastava, and J. C.-W. Lin, "Reliable customer analysis using federated learning and exploring deep-attention edge intelligence," *Future Generation Computer Systems*, vol. 127, pp. 70–79, 2022.

[16] Y. Feng, F. Lv, W. Shen, M. Wang, F. Sun, Y. Zhu, and K. Yang, "Deep session interest network for click-through rate prediction," *arXiv preprint arXiv:1905.06482*, 2019.

[17] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *KDD*, 2018, pp. 1059–1068.

[18] E. Barati and X. Chen, "An actor-critic-attention mechanism for deep reinforcement learning in multi-view environments," *arXiv preprint arXiv:1907.09466*, 2019.

[19] X. Shen, C. Yin, and X. Hou, "Self-attention for deep reinforcement learning," in *ICMAI*, 2019, pp. 71–75.

[20] C. Tao, S. Gao, M. Shang, W. Wu, D. Zhao, and R. Yan, "Get the point of my utterance! learning towards effective responses with multi-head attention mechanism." in *IJCAI*, 2018, pp. 4418–4424.

[21] C. Xi, G. Lu, and J. Yan, "Multimodal sentiment analysis based on multi-head attention mechanism," in *Proceedings of the 4th International Conference on Machine Learning and Soft Computing*, 2020, pp. 34–39.

[22] Z. Sun, S. Huang, H.-R. Wei, X.-y. Dai, and J. Chen, "Generating diverse translation by manipulating multi-head attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 8976–8983.

[23] M.-C. Lee, M. Ji, A. F. Molisch, and N. Sastry, "Throughput–outage analysis and evaluation of cache-aided d2d networks with measured popularity distributions," *IEEE Transactions on Wireless Communications*, vol. 18, no. 11, pp. 5316–5332, 2019.

[24] C. Li, J. Tang, H. Tang, and Y. Luo, "Collaborative cache allocation and task scheduling for data-intensive applications in edge computing environment," *Future Generation Computer Systems*, vol. 95, pp. 249–264, 2019.

[25] S. Wang, X. Zhang, K. Yang, L. Wang, and W. Wang, "Distributed edge caching scheme considering the tradeoff between the diversity and redundancy of cached content," in *2015 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2015, pp. 1–5.

[26] T. Ma, J. Qu, W. Shen, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, "Weighted greedy dual size frequency based caching replacement algorithm," *IEEE Access*, vol. 6, pp. 7214–7223, 2018.

[27] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.

[28] H. ElSawy, A. Sultan-Salem, M.-S. Alouini, and M. Z. Win, "Modeling and analysis of cellular networks using stochastic geometry: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 167–203, 2016.

[29] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7-9, pp. 1180–1190, 2008.

[30] R. Li, Y. Zhao, C. Wang, X. Wang, V. C. Leung, X. Li, and T. Taleb, "Edge caching replacement optimization for d2d wireless networks via weighted distributed dqn," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2020, pp. 1–6.

[31] J. Dilley and M. Arlitt, "Improving proxy cache performance: Analysis of three replacement policies," *IEEE Internet Computing*, vol. 3, no. 6, pp. 44–50, 1999.

[32] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 5668–5675.

[33] 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.

Chenyang Wang (Member, IEEE) received the B.S. and M.S. degrees in computer science and technology from Henan Normal University, Xinxiang, China, in 2013 and 2017, respectively. He is currently pursuing a Ph.D. degree from the School of Computer Science and Technology, College of Intelligence and Computing, Tianjin University, Tianjin, China. He is also a visiting Ph.D. student under the support of the China Scholarship Council (CSC) at the School of Electrical Engineering, Aalto University, Espoo, Finland since 15 May 2021. His current research interests include edge computing, big data analytics, reinforcement learning, and deep learning. He received the Best Student Paper Award of the 24th International Conference on Parallel and Distributed Systems from the IEEE Computer Society in 2018. He also received the Best Paper Award from the IEEE International Conference on Communications in 2021. In 2022, he received the "IEEE ComSoc Asia-Pacific Outstanding Paper Award".

Ruibin Li (Student Member, IEEE) received his B.S. and master degree from the School of Computer Science and Technology, College of Intelligence and Computing, Tianjin University, Tianjin, China. Now he is pursuing his Ph.D. degree at The Hong Kong Polytechnic University, Hong Kong, China. His current research interests include diffusion model, reinforcement learning, edge intelligence, distributed federated learning optimization, and deep learning.

Xiaofei Wang (Senior Member, IEEE) received the B.S. degree from Huazhong University of Science and Technology, China, and received M.S. and Ph.D. degrees from Seoul National University, Seoul, South Korea. He was a Postdoctoral Fellow with The University of British Columbia, Vancouver, Canada, from 2014 to 2016. He is currently a Professor at the College of Intelligence and Computing, Tianjin University, Tianjin, China. Focusing on the research of edge computing, edge intelligence, and edge systems, he has published more than 160 technical papers in IEEE JSAC, TCC, ToN, TWC, IoTJ, COMST, TMM, INFOCOM, ICDCS and so on. He has received the best paper awards of IEEE ICC, ICPADS, and in 2017, he was the recipient of the "IEEE ComSoc Fred W. Ellersick Prize", and in 2022, he received the "IEEE ComSoc Asia-Pacific Outstanding Paper Award".

Tarik Taleb (Senior Member, IEEE) received the B.E. degree (with distinction) in information engineering and the M.Sc. and Ph.D. degrees in information sciences from Tohoku University, Sendai, Japan, in 2001, 2003, and 2005, respectively. He is currently a Professor at the Center of Wireless Communications, at the University of Oulu, Finland. He is the founder and the Director of the MOSA!C Lab, Espoo, Finland. He was an Assistant Professor with the Graduate School of Information Sciences, Tohoku University, in a laboratory fully funded by KDDI until 2009. He was a Senior Researcher and a 3GPP Standards Expert with NEC Europe Ltd., Heidelberg, Germany. He was then leading the NEC Europe Labs Team, involved with research and development projects on carrier cloud platforms, an important vision of 5G systems. From 2005 to 2006, he was a Research Fellow with the Intelligent Cosmos Research Institute, Sendai. He has also been directly engaged in the development and standardization of the Evolved Packet System as a member of the 3GPP System Architecture Working Group. His current research interests include architectural enhancements to mobile core networks (particularly 3GPP's), network softwarization and slicing, mobile cloud networking, network function virtualization, software-defined networking, mobile multimedia streaming, intervehicular communications, and social media networking.

Song Guo (Fellow, IEEE) received the bachelor's degree in computer software from the Huazhong University of Science and Technology, the master's degree in computer engineering from the Beijing University of Posts and Telecommunications, and the Ph.D. degree in computer science from the University of Ottawa. He is currently a Full Professor and an Associate Head with the Department of Computing, The Hong Kong Polytechnic University. Before joining PolyU, he was a Professor with the University of Aizu, Japan. His research interests are mainly in the areas of big data, cloud computing, mobile computing, and distributed systems. Dr. Guo is an IEEE Fellow (Computer Society) and the Editor-in-Chief of the IEEE OPEN JOURNAL OF THE COMPUTER SOCIETY. He was a recipient of the 2019 IEEE TCBD Best Conference Paper Award, 2018 IEEE TCGCC Best Magazine Paper Award, 2019 and 2017 IEEE Systems Journal Annual Best Paper Award, and other six Best Paper Awards from IEEE/ACM conferences. His work was also recognized by the 2016 Annual Best of Computing: Notable Books and Articles in Computing in ACM Computing Reviews. Some of his Transactions papers were selected as Featured or Spotlight papers. He was a Distinguished Lecturer of IEEE Communications Society (ComSoc) and served in the IEEE ComSoc Board of Governors. He has been named on editorial board of a number of prestigious international journals like the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON CLOUD COMPUTING, and IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING. He has also served as chairs of organizing and technical committees of many international conferences.

Yuxia Sun (Member, IEEE) received the B.S. degree from Department of Computer Science, Huazhong University of Science and Technology, Wuhan, China, and the Ph.D. degree from the Department of Computer Science, Sun Yat-sen University, Guangzhou, China. She is currently an Associate Professor with Department of Computer Science, Jinan University, Guangzhou. She was a Research Associate with The Hong Kong Polytechnic University, Hong Kong SAR., and The University of Hong Kong, Hong Kong SAR., and was a Research Scholar with College of Computing, Georgia Institute of Technology, Atlanta, GA, USA. Her current research interests include machine learning, software safety, system safety, and software engineering.

Victor C. M. Leung (Life Fellow, IEEE) is a Distinguished Professor of Computer Science and Software Engineering at Shenzhen University. He is also an Emeritus Professor of Electrical and Computer Engineering and the Director of the Laboratory for Wireless Networks and Mobile Systems at the University of British Columbia (UBC). His research is in the broad areas of wireless networks and mobile systems. He has co-authored more than 1300 journal/conference papers and book chapters. Dr. Leung is serving on the editorial boards of IEEE Transactions on Green Communications and Networking, IEEE Transactions on Cloud Computing, IEEE Access, and several other journals. He received the IEEE Vancouver Section Centennial Award, 2011 UBC Killam Research Prize, 2017 Canadian Award for Telecommunications Research, and 2018 IEEE TCGCC Distinguished Technical Achievement Recognition Award. He co-authored papers that won the 2017 IEEE ComSoc Fred W. Ellersick Prize, 2017 IEEE Systems Journal Best Paper Award, 2018 IEEE CSIM Best Journal Paper Award, and 2019 IEEE TCGCC Best Journal Paper Award. He is a Life Fellow of IEEE, and a Fellow of the Royal Society of Canada, Canadian Academy of Engineering, and Engineering Institute of Canada. He is named in the current Clarivate Analytics list of Highly Cited Researchers.