QoE-oriented Soft Caching with Content Recommendation for Edge Computing Networks

Chenyang Wang*,††, Yan Chen‡,**, Chuan Sun§, Bosen Jia¶, Xiaofei Wang†,
Tarik Taleb‡, and Victor C. M. Leung‖
*College of Computer Science & Software Engineering, Shenzhen University, Shenzhen, China
†† Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, Guangdong, 518132, China
†College of Intelligence and Computing, Tianjin University, Tianjin, China
‡Faculty of Electrical Engineering and Information Technology, Ruhr University Bochum, Bochum, Germany
§College of Computing and Data Science, Nanyang Technological University, Singapore
¶China Unicom Digital Technology Co., Ltd, Beijing, China
‖University of British Columbia, Vancouver, Canada
E-mail: {chenyangwang@ieee.org, yanchen_edu@outlook.com, chuan.sun@ntu.edu.sg, jiabs10@chinaunicom.cn, xiaofeiwang@tju.edu.cn, tarik.taleb@rub.de, vleung@ieee.org},***Corresponding author*

***Abstract*—Mobile Edge Caching (MEC) can potentially alleviate Internet transmission congestion by delivering content at the network edge. However, current MEC solutions suffer from low resource utilization efficiency and often fail to meet user Quality of Experience (QoE), primarily due to dynamic user requests and obsessive pursuit of direct caching hits. Given the prevalence of recommendation systems, users often lack precise requests when using recommendation-based applications like TikTok and Taobao, insted passively enjoying recommended content. In this paper, we introduce a recommendation-enabled MEC architecture to enhance resource utilization and QoE. We develop a recommendation-enabled soft caching model and formulate the optimization problem as maximizing joint system revenue. To address this, we propose an attention-assisted federated learning deep Q-network algorithm. We conduct the simulations by using the real-world MIND dataset. The results demonstrate that our proposed algorithm outperforms existing baselines, demonstrating its effectiveness in improving resource utilization and QoE.**

I. Introduction

In recent years, the surge in data-intensive applications owns much to AI advancements, communication networks, and mobile tech [1]–[3]. Despite the boosted Internet bandwidth, managing this data flood remains a challenge. Mobile Edge Caching (MEC) tackles this by caching content at the network edge, reducing reliance on distant Cloud Data Centers (CDC) [4]. The cache hit ratio, crucial for MEC efficiency, measures the share of locally cached data in user requests on Edge Servers (ESs) [5]. Deep reinforcement learning (DRL) often complements federated learning (FL) to boost MEC model performance and training efficacy. However, decentralized training demands frequent parameter exchanges between cloud servers and ESs, which strains network resources [4].

Content platforms like TikTok and Netflix have a significant market influence [6], [7]. While users typically interact passively, their continued engagement is crucial for platform viability [8]. To bolster user retention, these platforms deploy recommendation systems that tailor content to individual preferences, boosting engagement [9]. In such applications, recurrent content requests and similar recommendations for users with comparable preferences are common [10], [11]. These platforms integrate MEC with recommendation systems to improve Quality of Service (QoS) and alleviate network congestion, mitigating delays and costs associated with retrieving data from Content Delivery Networks [12]. Nonetheless, optimizing caching policies to balance user requests and preferences poses challenges due to network infrastructure complexity and privacy concerns.

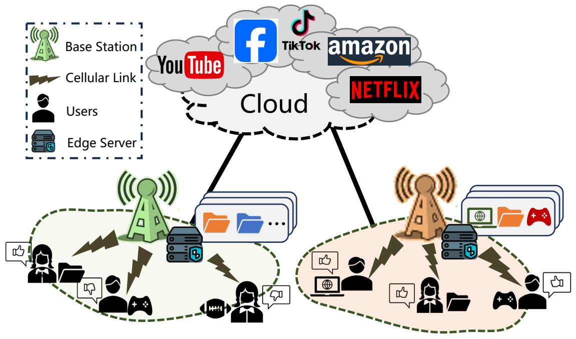


Fig. 1: Recommendation-enabled MEC

Soft edge caching has been developed for content-providing platforms, not only delivers the requested content but also offers alternative content aligned with user preferences when the original request is not available on ESs or when there is no specific content request. Based on these, this study introduces an edge soft caching scheme that integrates with existing recommendation systems to optimize content delivery efficiency at the edge. Additionally, we propose a caching update strategy utilizing a deep Q-network combined with attention-assisted Federated Learning (Att-FLDQN). Our key contributions are summarized as follows.

- We introduce a soft caching model accommodating the recommendation scheme to unveil the impact of user preference rather than merely improving the caching efficiency. Subsequently, the soft caching hit ratio and the content similarity have been further defined by jointly considering the caching and recommendation revenue.

- The optimization problem is decomposed into request processing and caching replacement, which is modelled as a Markov Decision Process (MDP) to maximize joint system revenue. Afterward, we propose an attention-assisted FL deep Q-network (Att-FLDQN) to address the dynamic changes in user-customized preferences.
- We conduct numerous simulations using the Microsoft MIND dataset. The results demonstrate that the proposed Att-FLDQN algorithm outperforms existing baselines.

II. System Model

The proposed recommendation-enabled mobile edge caching (R-MEC) architecture is shown as Fig. 1, where each end user (EU) connects to the Internet by accessing a base station (BS) and requesting content from the local ES directly associated with the BS. Each ES is equipped with limited storage and processing capacities to cache contents and run caching and recommendation algorithms. Additionally, each ES maintains a local caching list. We assume that the CDC has sufficient capabilities for computing and caching and can provide feedback on all the contents requested by EUs.

In the proposed R-MEC architecture, a set of EUs $\mathcal{U} = \{u|1,2,\cdots,U\}$ are randomly distributed in the network, and we denote the $\mathcal{E} = \{e|1,2,\cdots,E\}$ with the storage capacity $\mathbb{C}_e \in \mathbb{R}^+$ as the set of ESs equipped in BSs associated with the CDC. Besides, let $\mathcal{F} = \{f|1,2,\cdots,F\}$ indicate a set of all contents with the size $c_f \in \mathbb{R}^+$ provided by CDC. Assume that the CDC has sufficient storage space to cache $|F|$ contents, and each ES e can cache a subset of contents $\mathcal{F}_e$ from CDC, *i.e.*, $\mathcal{F}_e \subseteq \mathcal{F}$. A binary indicator $x^t_{e,f} = 1$ represents whether the content f is cached on ES e at time t or not, where $x^t_{e,f} = 1$ is cached and 0 is otherwise. Meanwhile, we consider one training process as a set of time slots $\mathcal{T} = \{t|1,2,\cdots,T\}$. Then, we have the capacities of ES are constrained by **C1:** $\sum_{f\in\mathcal{F}_e} x^t_{e,f} c_f \le \mathbb{C}_e, \forall e\in\mathcal{E}, \forall t$.

The mobility of EUs leads to dynamic connections between EUs and ESs. To this end, we use $y^t_{u,e} = 1$ to represent that EU u is directly connected to and requests contents from ES e at time t, $y^t_{u,e} = 0$ otherwise. Thus, $\mathcal{U}^t_e = \{u|y^t_{u,e} = 1, u\in\mathcal{U}\}$ is the set of EUs associated with ES e, and each EU can only connected to one BS, *i.e.*, **C2:** $\sum_{e\in\mathcal{E}} y^t_{u,e} = 1, \forall u\in\mathcal{U}, \forall t$.

In addition, we use a binary indicator $p^t_{u,f} = 1$ to represent the EU u requests a content $f\in\mathcal{F}$ from ES at the beginning of the time slice t, $p^t_{u,f} = 0$ otherwise. We assume that at each time slice, an EU can only request one content, *i.e.*, **C3:** $\sum_{f\in\mathcal{F}} p^t_{u,f} = 1, \forall u\in\mathcal{U}, \forall t$.

The Zipf distribution is widely employed to model the content popularity in time duration, and we use $\mathcal{P} = \{p^t_{u,f}\}_{u\in\mathcal{U}, f\in\mathcal{F}, t\in\mathcal{T}}$ to represent the set of all content requests, and $\mathcal{P} \sim Zipf(\beta, |\mathcal{F}|)$, where β is a parameter. Then, the direct cache hit ratio on an ES e can be defined as,

$$\mu'_e = \frac{\sum_{t\in T}\sum_{u\in\mathcal{U}^t_e}\sum_{f\in\mathcal{F}} p^t_{u,f} y^t_{u,e} x^t_{e,f}}{\sum_{t\in T}\sum_{u\in\mathcal{U}^t_e}\sum_{f\in\mathcal{F}} p^t_{u,f}}, \forall e\in\mathcal{E}. \quad (1)$$

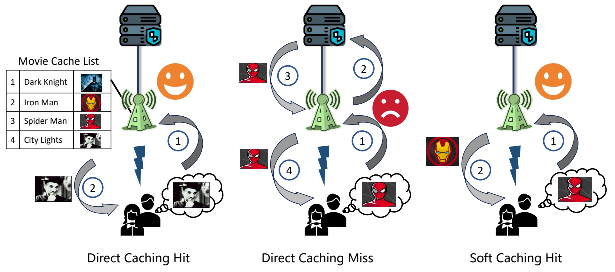


Fig. 2: Direct caching hit *v.s.* soft caching hit.

Suppose content f requested by an EU u is not cached on ES e. In that case, the ES can offer alternative content that aligns with the EU's preferences according to the analysis of the recommendation system. If the content occupies a high position in the EU's preference, we declare that the content hits the request softly, although the requested content is not directly satisfied. In this way, the soft caching hit ratio is defined as:

$$\mu_e = \sum_{t\in T}\sum_{u\in\mathcal{U}}\sum_{f\in F} p^t_{u,f} y^t_{u,e} (1 - \prod_{f'\in F}^{F} (1 - \omega^u_{f,f'} x^t_{e,f'})), \forall e\in\mathcal{E}, \quad (2)$$

where $\omega^u_{f,f'}$ represents the probability that u requests content f but also accepts content f'.

III. Content Recommendation in Edge Caching

A. Soft Caching Hit and Content Recommendation

We posit that there is an inherent potential for preference similarity between any two distinct users. Consequently, it is an attempt to recommend content to users that aligns with the preferences of other users exhibiting similar interests, which also serves as the foundational motivation of the study. Fig. 2 illustrates the difference between direct and soft caching hits. Assume that an ES caches four popular films: Dark Knight, Iron Man, Spider-Man, and City Lights. If "Spider-Man" is requested and not cached, the traditional approach requires fetching the film from a cloud center, potentially causing delays and network congestion. However, with a soft caching mechanism, if an EU requests "Spider-Man" but the ES lacks it yet has similar content like "Iron Man", the server can suggest it instead. This recommendation, based on the user's history of viewing superhero-themed films, increases the likelihood of acceptance and enhances the viewing experience.

Soft caching hit mainly depends on the recommendation aligned to the heterogeneous preferences of EUs. Let $\phi^u_f \in [0,1]$ represent the recommendation index of content f to EU u. Then, we can define the customised content recommendation list of EU u with recommendation indexes as Φ_u since the content provided by the system is finite. The recommendation system can learn from the users' preferences to obtain the recommendation list of each EU for all content (Φ_u). Then, Φ_u is correlated with the EU's preference. We can assume that $\phi^u_{f'} = \omega^u_{f,f'}$ for simplification.

When a request arrives, the ES e prioritizes providing feedback on the requested content f that is stored in the cache. In the absence of cached content, a soft caching hit scheme is initiated to provide feedback on alternative content f'. The determination of the recommendation index ranking between the requested content and other content is facilitated through the utilization of Φ_u. Subsequently, we designate contents positioned within the top 20% of Φ_u as soft caching hit content denoted by $\Phi_u^{20\%}$, and we establish $\omega_{f,f'}^u = 1$ if $f' \in \Phi_u^{20\%}$, and $\omega_{f,f'}^u = 0$ otherwise. The categorization of a request as a soft caching hit is contingent upon whether these designated contents are cached on ES e after the receipt of the request, otherwise, cache misses occur, necessitating the acquisition of content from the CDC.

B. Recommendation based on Collaborative Filtering

In this paper, we employ user-based collaborative filtering (userCF) [13], wherein recommendations are conducted by identifying items of interest to a user based on the preferences of others sharing similar interests. The likelihood of recommending content to a target EU necessitates the construction of a labelled directed graph derived from the analysis of interactions, encompassing positive and negative comments. In this paper, the graph is represented as a co-occurrence matrix $\mathbb{R}$, wherein each element $r_{i,j}$ signifies a score value indicative of the comments originating from user i towards item j. Subsequently, the recommended system seeks to estimate the value of $r_{u,f}$. To accomplish this, the userCF algorithm identifies the top n users with analogous interests to the target EU u. The estimation of comments from user u to specific content is derived by collectively examining the comments from these n identified users.

We employ gradient descent to iteratively optimize and obtain the optimal decomposition matrix, in which the objective function is defined as

$$G = \min_{\boldsymbol{m}^*, \boldsymbol{v}^*} \sum_{(u,f)\in \mathrm{K}} \left(\boldsymbol{r}_{u,f} - \boldsymbol{v}_f^{\mathrm{T}} \boldsymbol{m}_u \right)^2, \tag{3}$$

where $\boldsymbol{m}_u$ is the EU vector, $\boldsymbol{v}_f$ is the content vector, and K is the set of users' comments on content. Then, we use the Pearson correlation coefficient to calculate the similarity between an EU u and others d', *i.e.,*

$$\mathbb{S}_{u,u'} = \left| \frac{\sum (m_u - \overline{m_u})(m_{u'} - \overline{m_{u'}})}{\sqrt{\sum_{i=1}^{|\mathcal{U}|}(m_{u,i} - \overline{m_u})^2}\sqrt{\sum_{i=1}^{|\mathcal{U}|}(m_{u',i} - \overline{m_{u'}})^2}} \right|. \tag{4}$$

Considering user similarities, we can estimate the comments for a target EU u to content f according to relative comments from top-n similar EUs of u. This work employs a weighted average method, *i.e.,*

$$\phi_f^u = r_{u,f} = \frac{\sum_{u'\in\mathcal{U}} (\mathbb{S}_{u,u'} \cdot r_{u',f})}{\sum_{u'\in\mathcal{U}} \mathbb{S}_{u,u'}}, \forall u \in \mathcal{U}, \forall f \in \mathcal{F}. \tag{5}$$

During the system running, the newly arrived users are causing a lack of data to calculate user correlation, as above. We define the public comment $\boldsymbol{r}_{[\cdot,f]}(t)$ of a content f at current time t as the average of comments from all existing users $\hat{\mathcal{U}}$. Then, for a newly joined EU u, $r_{u,f}$ is initialized to be $\boldsymbol{r}_{[\cdot,f]}(t)$. The value of $r_{u,f}$ is calculated until u has interacted with f for more than τ_s times. Then, we have

$$\textbf{C4: } r_{u,f} = \begin{cases} r_{d,f} & \text{if } \sum_t p_{u,f}^t > \tau_s, \\ \boldsymbol{r}_{[\cdot,f]}(t) & \text{otherwise,} \end{cases} \forall u \in \mathcal{U}, \forall f \in \mathcal{F}, \tag{6}$$

where

$$\boldsymbol{r}_{[\cdot,f]}(t) = \frac{\sum_{u'\in\hat{\mathcal{U}}} r_{u',f}}{|\hat{\mathcal{U}}|}. \tag{7}$$

C. Joint Revenue of Caching and Recommendation

1) Recommend revenue: Recall that Φ_u represents the list of EU u's preference indexes on all contents. Thus, providing user content with a higher preference can achieve a higher QoE and higher revenue. As the content recommend index $\phi_{f'}^u$ is correlated with EU's preference, we can define the recommended reward of EU u as $\alpha_u^t(f, f')_{|f\neq f'} = \phi_{f'}^u$ in soft caching condition. Besides, when the requested content f is hit on ES, the reward is 1. Then, the recommendation reward obtained with the requested content f and the set of contents cached on the ES $\mathcal{F}_e^t$ as

$$\alpha_u^t(f, \mathcal{F}_e^t) = \begin{cases} 1 & \text{if } f \in \mathcal{F}_e \\ \max\limits_{f'\in\mathcal{F}_e^t} \alpha_u^t(f, f'), & \text{if } \mathcal{F}_e \cap \Phi_u^{20\%} \neq \varnothing \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

2) Caching revenue: The feedback decision is denoted as $\boldsymbol{z}_u^t = \left\langle z_u^E(t), z_u^S(t), z_u^C(t) \right\rangle$. If this tuple is empty, the requested content is cached on ES ($z_u^E(t) = 1$), provided feedback through the soft caching scheme ($z_u^S(t) = 1$), or obtained from CDC ($z_u^C(t) = 1$). If all three are true, then $z_u^E(t) + z_u^S(t) + z_u^C(t) = 1$. This encompasses scenarios where the requested content is cached on the ES or at least one content belonging to the top-20% preferred contents. We posit that cache hit refers to the scenario where an EU's content request receives a response from the ES, corresponding to situations $z_u^E(t) = 1$ and $z_u^S(t) = 1$. Consequently, we model cache revenue as $\beta_e^t = \mu_e^t \in [0, 1]$, signifying a positive correlation between cache revenue and cache hit ratio.

D. Problem Formulation

The system dynamics encompassing the requests from EU u, user mobility, and the overall requests received by EUs undergo continual fluctuations. Thus, ES e updates their cached content to augment the cache hit ratio. At each time slot, each ES e observes its current state, confining the cached content and requests originating from associated EUs. After this observation, the ES e updates its cached content to optimize revenue generation by catering to the service demands of these EUs. In this paper, we define the system revenue of each ES e as the weighted sum of recommendation and caching revenue, *i.e.,*

$$\mathcal{R}_e^t(\mathcal{S}_e^t, \mathcal{A}_e^t) = \lambda \sum_{u\in\mathcal{U}_e^t} \alpha_u^t(f, \mathcal{F}_e^t) + (1-\lambda)\mu_e^t, \forall e \in \mathcal{E}, \forall t \in \mathcal{T}. \tag{9}$$

Thus, the objective of updating cached content is

$$\max_{\boldsymbol{\pi}} \frac{1}{\mathcal{T}} \sum_{t=0}^{\mathcal{T}} \mathcal{R}_e^t(\mathcal{S}_e^t, \mathcal{A}_e^t) \tag{10}$$

$$s.t. \ \textbf{C1} \sim \textbf{C4}. \tag{11}$$

where $\boldsymbol{\pi} : \mathcal{S}_e^t \rightarrow \mathcal{A}_e^t$ represent the policy that can generate optimal action of updating caching contents $\mathcal{A}_e^t$ based on corresponding state observation $\mathcal{S}_e^t$.

IV. Proposed Approach

A. Problem Analysis

The formulated problem is classified as a classic case of Integer Linear Programming, which is NP-hard [14], [15]. Conventional methods are difficult to solve. Our objective is to develop a policy that optimally updates caching actions for any random state observation. Each ES observes the system state at the beginning of each time slice and updates its cached contents following the caching update action that the policy generates. Notably, the edge caching update process can be modelled as a Markov decision process (MDP).

We define the state observation of each ES at time slot t as the amalgamation of its caching state and the number of requests for content received from all associated users, *i.e.,*

$$\mathcal{S}_e^t = \Big\{\{x_{e,f}^t\}_{f\in\mathcal{F}}, \{\sum_{u\in\mathcal{U}} y_{u,e}^t p_{u,f}^t\}_{f\in\mathcal{F}}\Big\}, \forall e \in \mathcal{E}, \forall t \in \mathcal{T}. \quad (12)$$

Each ES needs to determine how to satisfy requests (*i.e.,* $\boldsymbol{z_e^t}$) and which content should be replaced. Due to limited resources and response time, we set each ES to replace at most one cached content. Let $\boldsymbol{a_e^t} = \{a_{e,f}^t\}_{f\in\{0\}\cup\mathcal{F}}$ denote the content replace action, where $a_{e,f}^t \in \{0,1\}$, and $\sum_{f=0}^{|\mathcal{F}|} a_{e,f}^t = 1$. When $a_{e,f} = 1$, it indicates that content f should be replaced, 0 is otherwise. Thus, the action of each ES is

$$\mathcal{A}_e^t = \{\boldsymbol{z_e^t}, \boldsymbol{a_e^t}\}, \forall e \in \mathcal{E}, \forall t \in \mathcal{T}. \quad (13)$$

Algorithm 1 Policy training based on DQN

```
Input: EC system, initial policy, replay buffer
 1: for $t = 1, 2, 3, \cdots, T$ do
 2:     Receive a request for content $f$ from EU
 3:     if $x_{e,f}^t$ then
 4:         Continue
 5:     else
 6:         Observe state: $\mathcal{S}_e^t$;
 7:         Get $\boldsymbol{z_e^t}$ by checking $\{x_{e,f}^t\}_{f\in\mathcal{F}}$.
 8:         Generate $\boldsymbol{a_e^t}$ by policy with input $\mathcal{S}_e^t$.
 9:         Execute $\mathcal{A}_e^t = \{\boldsymbol{z_e^t}, \boldsymbol{a_e^t}\}$.
10:         Obtain $\mathcal{R}_e^t(\mathcal{S}_e^t, \mathcal{A}_e^t)$ and observer $\mathcal{S}_e^{t+1}$.
11:         Store $< \mathcal{S}_e^t, \mathcal{A}_e^t, \mathcal{R}_e^t, \mathcal{S}_e^{t+1} >$ to replay buffer.
12:         Sample a batch of experience from the replay buffer and
            update Q-networks.
13:         if $t\%t_\gamma$ then
14:             Update target networks.
```

B. Att-FLDQN Caching Update

After receiving requests, the action selection model checks if the requested content exists on ES. When content exists, the ES feedbacks the requested content, and $z_u^E(t) = 1$. Otherwise, the soft caching scheme is triggered to check if there is any of the top-20% preferred content of the EU cached on ES. Then, if there exists any alternative content, $z_u^S(t) = 0$ and return the content with the highest similarity to EU. Otherwise, $z_u^S(t) = 0$, and the ES acquires the required content from the CDC and returns it to the EU. Meanwhile, a content replacement operation is triggered to update the cached content on the ES. During this procedure, the interaction record module stores these interaction experiences in records for further knowledge utilization.

The request processing action (*i.e.,* $\boldsymbol{z_e^t}$) depends on the contents cached on ES. Besides, it is independent of the caching update action. Thus, the request processing and caching update can be executed separately in each time slice. The request processing is executed as described above. Then, DRL in our work addresses the caching replacement action, and the policy training based on DQN is shown as Algorithm 1.

To reduce transmission load and privacy risks associated with accumulating original data, we employ FL to train the caching update policy. The CDC maintains global parameters Θ, and each ES starts updating with the same initial parameters θ_e^t from the global model based on local experience. The CDC then aggregates these updates to refine the global parameters, and the whole process continues until convergence.

The global update is modelled as a weighted accumulation by using the attention mechanism, *i.e.,*

$$\min_{\theta,w}\{F(\Theta)\} \triangleq \sum_{e\in\mathcal{E}} w_e F_e(\theta_e^t), \quad (14)$$

where w_e is the weight of ES e, indicating importance of the model deployed on ES e for global model aggregation. In this paper, we integrate attention mechanisms to calculate the weights for global aggregation based on composite metrics, including hardware and data features.

We define the composite metric of each ES e as a tuple $\mathbb{K}_e =< CA_e, RB_e, DS_e, AR_e >$, where CA_e is the computing ability, RB_e is the replay buffer size that can reflect the memory capacity, DS_e is the size of training data that reflects the data abundance, and AR_e is the average reward that can reflect the quality of the current local model. We select $\mathbb{K}_e$ as the Key and the model parameter as the Value in the attention mechanism. Then, the query is designed as

$$\mathbb{Q} = [\max_e\{CA_e\}, \max_u\{RB_e\}, \max_e\{DS_e\}, \max_e\{AR_e\}]. \quad (15)$$

In this way, the input of the attention mechanism includes $\mathbb{Q}, \mathbb{K}_e$, θ_e^t on CDC. Then, the weight is obtained, *i.e.,*

$$w_e = Attention(\mathbb{Q}, \mathbb{K}_e) = softmax(\frac{\mathbb{Q}\mathbb{K}_e^T}{\sqrt{d}}), \forall e \in \mathcal{E}, \quad (16)$$

where d is the dimension.

The policy training process based on Att-FLDQN is illustrated in Algorithm 2. Initially, each ES uploads the local parameters, and the information including server computing power, experience replay pool size, training data size and average reward information (Lines 1-7). Calculate the weight of each model in the parameter aggregation operation (Line 8). Weighted aggregation is performed to obtain global parameters (Line 9). Distribute global parameters to each ES and update parameters uniformly (Lines 10–11). The computational complexity primarily arises from the DQN training process, denote the number of multiplication operations and the mini-batch in each DQN model as Ψ and Δ, respectively. For T time slots, and considering the storage capacity E, we have the computational complexity of Algorithm 2 as $O(TE\Delta\Psi)$.

Algorithm 2 Policy training based on Att-FLDQN

```
Input: EC system, initial policies, replay buffer
 1: Initialize local model parameters as the global model;
 2: for $t = 1, 2, 3, \cdots, T$ do
 3:     Each ES:
 4:       Receive requests from EUs;
 5:       Update the local model as Algorithm 1;
 6:     if $t\%t_a$ then
 7:        Each ES uploads $\theta_e^t, CA_e, RB_e, DS_e, AR_e$;
 8:        CDC calculate $w_e, \forall e \in \mathcal{E}$ according to (16);
 9:        Update the global model $\Theta^t$ according to (14);
10:        Dispatch global model parameters to ES;
11:        ES updates local model $\theta_e^t \leftarrow \Theta^t, \forall e \in \mathcal{E}$;
```

V. Simulation Results

A. Simulation Setup

We conduct experiments using the MIND dataset [16], a news recommendation dataset that Microsoft released in 2020. We select the top 1000 legitimate news articles that users have clicked on and are still accessible to form the content list. The interaction between content and EU is obtained from the file “behaviors.tsv”, which records the user’s clicks and displays, including ID, user ID, record time, and a list of clicked and viewed news articles.

TABLE I: Settings on Key Parameters

Parameter	Value	Parameter	Value
Replay buffer size	500	ϵ	0.9
Batch size	256	γ	0.95
Hidden layers	2	t_r	100
Hidden size	128	λ	0.5
Optimizer	Adam	Learning rate	0.001

Table I shows the key parameters of the simulations. We set up 4 ESs and 80 EUs, and the storage capacity of each ES is 2048 MB. Each ES provides content for 20 EUs in a training round, which includes 50 time slots. At the beginning of each training round, computing resources are allocated to each ES, randomly selected from [8, 16]. The requests of EUs follow Zipf distribution with $\beta = 1$. We set the content size as the count of words in the news with 1 MB/word. Then, we split the content size into 50 groups, with every 40 MB being a group with [1, 2000] MB. Then, the recommended index for any content to an EU is obtained from Eq. (5). The evaluation from the user to content ($r_{u,f}$) is defined as including click (**1**), view (**0**), and ignore behaviours (**-1**). The interaction threshold of a new EU is 10, *i.e.,* τ_s=10. We compare the proposed approach with FIFO [17], LFU [18], LRU [19], Distributed-DQN [20], and FedAvg-DQN [21].

B. Simulation Results

To explore the effects of recommendation and caching revenues, we consider two cases by varying the parameter λ in the system revenue. We set λ = 1 (only recommendation) and λ = 0 (only caching) to exclusively benefit users and content providers, respectively. As shown in Fig. 3, the system revenue derived from recommendations and caching stabilizes at 0.17 and 0.12, respectively, around the 3000th and 2000th rounds. This highlights that the proposed Att-FLDQN comprehensively accounts for the reciprocal influence of recommendation and caching, achieving a trade-off between them. This is mainly because only caching operations necessitate execution based on the selected replacement action.

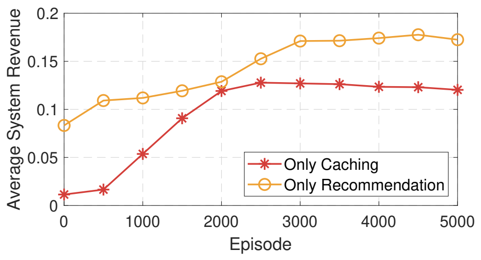
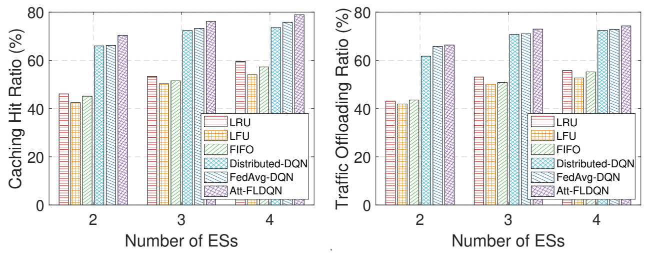


Fig. 3: Effects of Different Revenues.



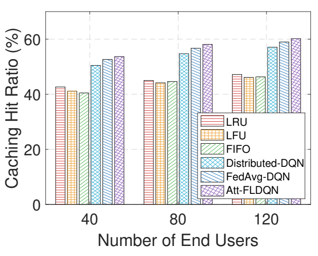
(a) Caching Hit Ratio (b) Traffic Offloading Ratio

Fig. 4: Effects of different numbers of ESs.

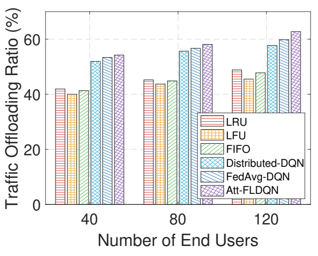
Fig. 4 shows the effects of different numbers of ESs on the caching hit ratio and traffic offloading ratio. The proposed algorithm achieves the best performance compared to the others. As the number of ESs increases, the caching hit ratio and traffic offloading ratio increase because more content is cached at the network edge and more requests can be satisfied. Notably, when the number of ESs is 4, the caching hit ratio of the proposed algorithm reaches nearly 80%.

Fig. 5 shows the effects of different numbers of EUs. It can be seen that Att-FLDQN outperforms the other three rule-based algorithms and slightly outperforms the two DRL-based algorithms in both metrics. This is because the proposed algorithm has better cache replacement action learning capability, and also shares the training results by considering the training effects of different models more reasonably. As the number of EUs grows, the caching hit ratio and traffic offloading ratio improve, and the advantage of the proposed algorithm becomes more pronounced. This is because more EUs bring more differentiated training samples, enhancing the learning abilities of the model.

Fig. 6 shows the effects of cache capacity sizes on the caching hit ratio and traffic offloading ratio, respectively. Both

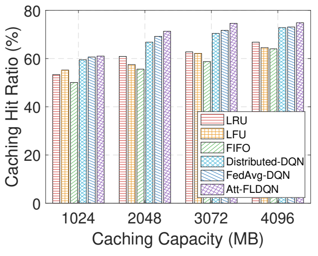


(a) Caching Hit Ratio

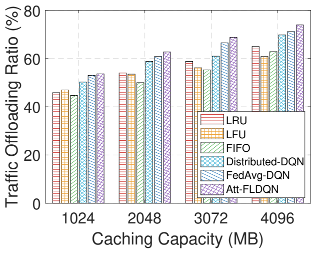


(b) Traffic Offloading Ratio

Fig. 5: Effects of different numbers of end users.



(a) Caching Hit Ratio



(b) Traffic Offloading Ratio

Fig. 6: Effects of different caching capacity sizes.

the caching hit ratio and traffic offloading ratio will increase as the cache capacity of the ES goes up. Additionally, as the cache capacity increases, the difference between the Att-FLDQN algorithm and other benchmark algorithms gradually grows. When the cache capacity is 4096 MB, Att-FLDQN achieves a higher caching hit ratio and a higher traffic offloading ratio of 8.15%, compared to the best-performing rule-based algorithm, LRU. The traffic offloading ratio is 8.5% higher, while the caching hit ratio is 1.8% higher and the traffic offloading ratio is 2.47% higher than FedAvg-DQN.

VI. Conclusion

In this work, we have investigated recommendation-enabled Mobile Edge Caching (MEC) to enhance resource utilization and QoE of users. Then, a soft caching model has been introduced, and we have formulated the optimization problem as maximizing joint system revenue. The model has incorporated both direct and soft caching hit, effectively accommodating a wide range of user requests by catering to their preferences. To address the complex problem, we have decomposed its request processing and caching replacement and proposed an Att-FLDQN algorithm. The algorithm continuously updates the cached content of each ES to meet dynamic user requests. Simulations based on real dataset have verified the effectiveness of the proposed algorithm.

Acknowledgement

This work was supported in part by the National Science Foundation of China under Grant No. 62072332; in part by the Tianjin Xinchuang Haihe Lab under Grant No.Grant 22HHX-CJC00002. This work was also conducted at ICTFICIAL Oy, Finland; and supported in part by the European Union's HE research and innovation program HORIZON-JUSNS-2023 under the 6G-Path project (Grant No. 101139172) and by the AerOS project funded by the European Union's Horizon Europe, the EU's key funding program for research and innovation under Grant No. 101069732. The paper reflects only the authors' views, and the European Commission bears no responsibility for any utilization of the information contained herein.

References

[1] Y. F. *et al.*, "A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6142–6163, 2022.

[2] Y. Zhao and Z. L. *et al.*, "Socialized learning for smart cities: Cognitive paradigm, methodology, and solution," *IEEE Wirel. Commun.*, vol. 28, no. 5, pp. 200–208, 2021.

[3] J. Kang, J. Zhang, H. Yang, D. Ye, and M. S. Hossain, "When metaverses meet vehicle road cooperation: Multi-agent drl-based stackelberg game for vehicular twins migration," *IEEE Internet of Things Journal*, 2024.

[4] C. Wang and R. L. *et al.*, "Heterogeneous edge caching based on actor-critic learning with attention mechanism aiding," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 6, pp. 3409–3420, 2023.

[5] C. S. *et al.*, "Federated deep reinforcement learning for recommendation-enabled edge caching in mobile edge-cloud computing networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 3, pp. 690–705, 2023.

[6] C. A. Gomez-Uribe and N. Hunt, "The netflix recommender system: Algorithms, business value, and innovation," *ACM Trans. Manag. Inf. Syst.*, vol. 6, no. 4, pp. 13:1–13:19, 2016.

[7] Y. Ren, W. Qi, and M. Fan, "The development of tik tok's global market," in *ICEMCI*. Atlantis Press, 2021, pp. 2779–2784.

[8] Y. Wei and X. W. *et al.*, "Graph-refined convolutional network for multimedia recommendation with implicit feedback," in *The 28th ACM ICME, Seattle, WA, USA, October 12-16, 2020*, pp. 3541–3549.

[9] A. Pfadler, H. Zhao, and *et al.*, "Billion-scale recommendation with heterogeneous side information at taobao," in *36th IEEE ICDE, Dallas, TX, USA, April 20-24, 2020*, 2020, pp. 1667–1676.

[10] D. Tsigkari and T. Spyropoulos, "An approximation algorithm for joint caching and recommendations in cache networks," *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 2, pp. 1826–1841, 2022.

[11] J. Liao and W. Z. *et al.*, "Sociallgn: Light graph convolution network for social recommendation," *Inf. Sci.*, vol. 589, pp. 595–607, 2022.

[12] P. Sermpezis and T. G. *et al.*, "Soft cache hits: Improving performance through recommendation and delivery of related content," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1300–1313, 2018.

[13] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 421425, pp. 1–19, 2009.

[14] Y. Chen, Y. Sun, H. Yu, and T. Taleb, "Joint task and computing resource allocation in distributed edge computing systems via multi-agent deep reinforcement learning," *IEEE Transactions on Network Science and Engineering*, vol. 11, no. 4, pp. 3479–3494, 2024.

[15] K. Poularakis and J. L. *et al.*, "Joint service placement and request routing in multi-cell mobile edge computing networks," in *IEEE INFOCOM, Paris, France, April 29 - May 2, 2019*. IEEE, 2019, pp. 10–18.

[16] F. Wu and *et al.*, "MIND: A large-scale dataset for news recommendation," in *Pro. of the 58th AMACL*, Jul. 2020, pp. 3597–3606.

[17] D. Morse and G. Richardson, "The lifo/fifo decision," *J. acnt. resch.*, pp. 106–127, 1983.

[18] C. Li and Y. Z. *et al.*, "Collaborative caching strategy based on optimization of latency and energy consumption in MEC," *Knowl. Based Syst.*, vol. 233, p. 107523, 2021.

[19] P. Yuan and S. S. *et al.*, "Caching hit ratio maximization in mobile edge computing with node cooperation," *Comput. Networks*, vol. 200, p. 108507, 2021.

[20] X. Wang and C. W. *et al.*, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9441–9455, 2020.

[21] X. Li, K. Huang, and *et al.*, "On the convergence of fedavg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.