Federated Distributed Deep Reinforcement Learning for Recommendation-enabled Edge Caching

Huan Zhou, *Senior Member, IEEE,* Hao Wang, Zhiwen Yu, *Senior Member, IEEE,*
Guo Bin, *Senior Member, IEEE,* Mingjun Xiao, *Member, IEEE,* and Jie Wu, *Fellow, IEEE*

***Abstract*—Recently, in response to the low efficiency and high transmission latency of traditional centralized content delivery networks, especially in congested scenarios, edge caching has emerged as a promising method to bring content caching closer to the edge of the network. However, traditional content delivery methods might still lead to low utilization of cache resources. To tackle this challenge, this paper investigates a content recommendation-based edge caching method in multi-tier edge-cloud networks while considering content delivery and cache replacement decisions as well as bandwidth allocation strategies. Firstly, we consider a multi-tier edge caching-enabled content delivery network architecture combined with a content recommendation system and formulate the optimization problem with the objective of minimizing long-term content delivery delay and maximizing cache hit rate. Secondly, considering time-varying system environments and uncertain content demands, we approximate the optimization process of content delivery and cache replacement for each agent as a Partially Observable Markov Decision Process (POMDP) and propose a single-agent Deep Deterministic Policy Gradient (DDPG)-based method. Subsequently, we extend the POMDP to a multi-agent scenario. To address the issue of agents converging to local optima and establish more personalized models, we propose a Federated Distributed DDPG-based method (FD3PG) to solve the corresponding problem in a multi-agent system. Finally, simulation results demonstrate that the proposed FD3PG achieves lower delivery delay and higher cache hit rate compared with other baselines in various scenarios. Specifically, compared with FADE, MADRL, and DDPG, FD3PG achieves a significant decrease in average delivery delay, approximately 10%, 11%, and 35% on the Synthetic dataset, and 12%, 14%, and 48% on the MovieLens Latest Small dataset, respectively.**

***Index Terms*—Edge caching, content recommendation, distributed training, federated learning, deep reinforcement learning.**

I. Introduction

IN the current digital era, with the exponential growth of online content and a surge in demand for rapid and customized content delivery, network caching systems are facing unprecedented challenges [1]–[4]. Traditional Content Delivery Networks (CDNs) typically employ a centralized architecture, relying on cloud servers or other centralized servers for content caching and distribution. While this method has proven effective over the years, the explosive growth of information and the constantly evolving demand for real-time and personalized contents from User Devices (UDs) have put enormous pressure on this method [5], [6]. The drawback of centralized CDNs lies in the fact that servers are often deployed far from UDs, resulting in long-distance transmission that may significantly increase transmission delay. The expectations of UDs for instant access to information and highly personalized content pose challenges for centralized CDNs to meet the growing demands for real-time and personalized content. Simultaneously, the fixed cache location and limited cache capacity make it increasingly challenging to accurately predict and meet the various content requirements of UDs [7], [10], [11]. As a result, the search for a more flexible and efficient content delivery method has become an urgent priority.

Recently, edge caching has emerged as a promising method [12]. In contrast to traditional CDNs heavily relying on centralized data centers, edge caching deploys cache-enabled Edge Servers (ESs) at the network edge, such as Base Stations (BSs), bringing contents closer to UDs. This decentralized architecture offers several advantages. Firstly, ESs are deployed closer to UDs than cloud servers, significantly reducing content delivery delay. Secondly, edge caching enhances scalability and adaptability to constantly changing content demands [16]. Additionally, this architecture helps alleviate network congestion by distributing content delivery across multiple ESs. Despite these potential benefits, edge caching faces challenges [17]. Due to the limited cache space of ESs deployed on BSs, only a subset of contents can be cached locally on these servers. When UDs request content not available in ESs, they still need to retrieve it from cloud servers or other centralized servers, diminishing the advantages of edge caching, especially in cases of numerous content requests [18]. Therefore, finding a solution to alleviate this limitation is crucial for improving the utilization of limited cache space and the overall performance of edge caching.

The Content Recommendation System (CRS) can analyze the interests and personalities of UDs to recommend specific contents to them. By integrating edge caching with CRS, edge

H. Zhou is with the School of Computer Science, Northwestern Polytechnical University, Xi'an, 710129, China, and also with the College of Computer and Information Technology, China Three Gorges University, Yichang, 443002, China (e-mail: zhouhuan117@gmail.com).

H. Wang is with the Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, the College of Computer and Information Technology, China Three Gorges University, Yichang, 443002, China (e-mail: wanghhhaoo@gmail.com).

Z. Yu is with the School of Computer Science, Northwestern Polytechnical University, Xi'an, 710129, China, and also with Harbin Engineering University, Ha'erbin 150001, China (e-mail: zhiwenyu@nwpu.edu.cn).

G. Bin is with the School of Computer Science, Northwestern Polytechnical University, Xi'an, 710129, China (e-mail: guob@nwpu.edu.cn).

M. Xiao is with the School of Data Science, University of Science and Technology of China, Anhui, 230026, China (e-mail: xiaomj@ustc.edu.cn).

J. Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA (e-mail: jiewu@temple.edu).

servers can provide alternative solutions closely related to the content requested by UDs. This makes content requests easier to fulfill, even when the requested contents are not readily available in the ES's cache space [13]–[15]. The combination of edge caching and CRS not only optimizes the utilization of limited cache space but also reduces reliance on centralized servers. This enhancement benefits both the Quality of Experience (QoE) for UDs and the Quality of Service (QoS) for ESs. Inspired by this, this paper investigates recommendation-enabled edge caching, combining caching strategy and CRS in multi-tier edge-cloud networks. Initially, a multi-tier edge caching-enabled CDN architecture is considered. CRS is combined into the CDN to recommend similar contents to UDs when the requested contents are unavailable at Small Base Stations (SBSs). Cooperative caching among multiple SBSs and a Macro Base Station (MBS) is utilized to reduce content delivery delay and backhaul traffic in the CDN.

Furthermore, adopting appropriate caching strategies by SBSs is crucial. With the development of 5G networks, there has been an exponential growth in content diversity and data complexity [8], [9]. Traditional optimization methods, relying on fixed, static rules or manual configuration, evidently cannot adapt flexibly to environmental changes. In contrast, Artificial Intelligence (AI)-based methods, such as Deep Reinforcement Learning (DRL), can utilize data-driven intelligent decision-making and optimization techniques to automatically learn and adjust caching strategies to adapt to evolving data access patterns and network conditions. This makes them more feasible and effective for practical applications. However, traditional multi-agent DRL methods often aim to train a unified model for all agents, neglecting the need for personalization required by agents deployed in different environments. Therefore, improving cache hit rate and reducing content delivery delay while ensuring agent personalization for each agent is a pressing issue that needs to be addressed.

To tackle these challenges, the decision-making process of each SBS is approximated as a Partially Observable Markov Decision Process (POMDP) in this paper. Then, a single-agent Deep Deterministic Policy Gradient (DDPG)-based method is proposed, in which each agent interacts with the environment and learns the optimal policy based on the feedback (rewards) from the environment. It is worth noting that we embrace a cooperative model wherein multiple agents coexist simultaneously. However, single-agent DRL-based methods frequently face challenges in multi-agent scenarios and may converge to local optimal solutions. Many existing multi-agent DRL-based methods often follow a training strategy of centralized learning and distributed execution. This strategy necessitates cross-communication between agents to share their intelligence and achieve the optimal global solution, which could increase the communication load. Federated Learning (FL), as a novel distributed machine learning approach, conducts model training across multiple devices or servers rather than concentrating on a single central location [51], [52]. Motivated by this technique, we propose a **F**ederated **D**istributed **D**eep **D**eterministic **P**olicy **G**radient-based method (FD3PG), which allows all agents to share their intelligence and carry out federated updating after several episodes. Furthermore, unlike traditional federated averaging algorithms that prioritize model generalization at the expense of personalization, FD3PG utilizes an Earth Mover's Distance (EMD)-based method for model selection to ensure personalized training while optimizing overall performance.

The main contributions of this paper can be summarized as follows:

- A multi-tier edge caching-enabled CDN architecture, combined with a CRS, is contemplated. The optimization problem is formulated with the objective of minimizing long-term content delivery delay and maximizing cache hit rate.
- To minimize content delivery delay and maximize cache hit rate, the optimization process of content delivery and cache replacement for each SBS is approximated as a POMDP. A single-agent DDPG-based method is proposed to solve the POMDP.
- The POMDP is then extended to a multi-agent scenario. To address the issue of agents converging to local optima and establish more personalized models, a more effective method is further designed, referred to as FD3PG. It combines FL principles into the single-agent DDPG.
- Extensive simulations are conducted to rigorously evaluate the performance of FD3PG under various scenarios. Simulation results show that, compared with FADE, MADRL, and DDPG, FD3PG achieves a significant decrease in average delivery delay, approximately 10%, 11%, and 35% on the Synthetic dataset, and 12%, 14%, and 48% on the MovieLens Latest Small dataset, respectively.

The rest of this paper is organized as follows: We summarize the related work in Section II and describe the system model as well as the problem formulation in Section III. Then, we approximate the decision-making process of each SBS as a POMDP and propose a single-agent DDPG-based method in Section IV. In Section V, we extend the POMDP to a multi-agent scenario, and further propose FD3PG to solve the optimization problem. We analyze the simulation in Section VI. Finally, we summarize the conclusion in Section VII.

II. Related Work

Currently, various works have explored the realms of edge caching and content recommendation. In this section, we provide a brief review of existing works from these four perspectives.

A. Edge Caching

In recent years, edge caching technology has become integral to CDNs. Lyu *et al.* proposed a strategy for deploying edge caches at a large scale in WiFi networks in [19], and designing a traffic-weighted greedy-based algorithm to maximize long-term caching gain. Tian *et al.* in [20] introduced a FL-based cooperative caching framework to address computational complexity and communication cost issues in mobile edge networks. In [21], Li *et al.* exploited a joint optimization scheme for FL, considering participant selection and resource allocation to minimize energy consumption and

training delay. Qiao *et al.* in [22] proposed an adaptive FL-based active content caching algorithm to meet low-delay requirements for content access. Saputra *et al.* in [23] designed both a centralized caching-delivering algorithm to minimize content delivery delay and then extend it to a large-scale network. Kharbutli *et al.* in [24] introduced a last-level cache-based cache replacement algorithm that is cost-sensitive and locality-aware.

While these studies have successfully showcased the effectiveness of edge caching technology, their focus remains predominantly on scenarios where content requests must be fully fulfilled. Unfortunately, this singular focus may overlook the inherent challenges posed by the limited cache capacity of ESs, potentially undermining the advantages of edge caching in certain cache hit methods. Moreover, many of the cache replacement strategies proposed in the aforementioned studies may not perform optimally in highly dynamic and large-scale scenarios, introducing a need for more robust solutions in these complex environments.

B. Content Recommendation

Recommendation algorithms play a crucial role in assisting users with information tasks. Content recommendation serves as a method to identify contextually relevant information for these tasks. For instance, Xu *et al.* in [26] introduced MetaCAR, utilizing a dual conditional variational autoencoder to generate user ratings and design mutually exclusive tasks in recommendation scenarios. Liu *et al.* combined a graph-based goal planning module with a goal-guided responding module in [27], presenting a two-stage multi-goal-driven conversation generation framework.

The integration of edge caching and CRS enhances ESs' ability to meet users' demands by providing relevant alternatives closely aligned with the requested content. While this area of research is relatively underexplored, several works have made notable contributions. Sheng *et al.* in [28] investigated cooperative content caching with recommendations to maximize caching gains. Sun *et al.* in [29] explored a mobile computing network combining edge caching and CRS and employed a soft hit method to reduce content delivery delay. Song *et al.* in [30] presented a method that combines user-side recommendation and D2D-assisted caching, considering personalized preferences and relative locations of users through D2D links. Yu *et al.* in [31] proposed a system to reduce the cost of content service centers by jointly considering content caching and recommendation through opportunistic mobile networks. Li *et al.* in [32] introduced a recommendation-aided edge caching approach using a group interest aggregation algorithm to determine content caching strategies for edge nodes. Fu *et al.* in [33] investigated a cache hit rate maximization problem and proposed a framework to achieve the optimal solution regarding joint cache placement and recommendation decisions. However, when considering CRS, the above studies ignore the impact of the heterogeneity of user references on the accuracy of recommendation algorithms.

C. Multi-agent DRL-based Methods

AI-based methods, such as DRL, have been applied in many works due to their ability to autonomously learn and adjust caching strategies. Ndikumana *et al.* in [34] proposed a deep learning model to predict the contents cached in self-driving cars and multi-access edge computing servers attached to roadside units. To reduce the overall backhaul congestion and access delay, Lekharu *et al.* in [35] proposed a deep learning-based caching model according to the content popularity at different time slots of the day. Yu *et al.* in [36] proposed a mobility-aware proactive edge caching scheme based on FL to adapt to the changing popularity of content. This method employs Context-aware Adversarial AutoEncoder to predict the highly dynamic content popularity. Bakr *et al.* in [37] proposed an end-to-end Deep Learning framework for proactive content caching that considers the dynamic interaction between users and content items, particularly their features. Pang *et al.* in [38] developed a deep-learning-based solution, DeepCache, to facilitate smart caching beyond simple frequency- and time-based replace strategies and cooperation among BSs. Although the above works have proven their effectiveness, they aim to train a uniform model for all agents leading to suboptimal solutions when agents are deployed in diverse environments.

D. Federated DRL-based Methods

To further enhance training efficiency and reduce the complexity of agent interactions, federated DRL is gradually becoming a significant research direction. Zhao *et al.* in [39] proposed a unified federated deep Q-learning caching scheme for collaborative edge networks, aiming to minimize the long-term average system cost under the uncertainties of heterogeneous user demands and dynamic content popularity. Li *et al.* in [40] utilized federated DRL to optimize content caching and distribution strategies, enabling adaptive collaboration between different UAVs while optimizing overall network performance. Wu *et al.* in [41] proposed a multi-agent federated DRL-based cooperative caching strategy for vehicular edge networks. By utilizing a recurrent neural network and a multi-head attention-based popularity prediction model, this strategy effectively enhances cache hit rate and content delivery efficiency. Jiang *et al.* in [42] combined asynchronous FL and DRL to design a mobility-aware edge caching strategy, which reduces backhaul pressure and delay in the Internet of Vehicles. Lei *et al.* in [43] proposed a federated DRL-based partial cooperative edge caching scheme. This scheme balances user-specific local characteristics and overall global characteristics by switching data training between two models maintained at each fog access point.

Compared to the aforementioned studies, the FD3PG proposed in this paper considers personalization of users and agents, and optimizes edge caching strategies from a distributed perspective. Additionally, to address the issue of local optima, we allow agents to engage in a phase of parameter sharing and aggregation after a fixed number of training rounds, which can not only improve model accuracy but also reduce communication load.

TABLE I
SYMBOLS AND DEFINITIONS

Symbol	Definition
$\mathcal{F}$	The content library
c_f	The size of content f
$\mathcal{I}$	The set of BSs
C_i	The cache capacity of SBS i
B_i	The bandwidth of SBS i
$\mathcal{T}$	The set of time slots
$\mathcal{CS}_i^t$	The cache state of SBS i in time slot t
$\mathcal{M}$	The set of UDs
$\mathcal{M}_i$	The set of UDs within the scope of SBS i
$pos_{m,i}$	The position of UD m
$q_{m,i}^{f,t}$	The content request of UD m
$sim_m^{f,g}$	The similarity score between contents f and g for UD m
λ_m^f	The UD m's rating score on content f
$\mathcal{X}_m$	The similarity matrix for UD m
$w_{m,i}^{f,t}$	The willingness of UD m to accept recommendations in time slot t
β_i^t	The cache replacement decision of SBS i in time slot t
$L_{m,i}^{f,t}$	The delivery delay of content f via the UD-SBS link
$L_{i,j}^{f,t}$	The delivery delay of content f among SBSs
$L_{i,0}^{f,t}$	The delivery delay of content f via the SBS-MBS link
L_m^t	The delivery delay of content f requested by UD m in time slot t
$r_{m,i}^t$	The wireless downlink transmission rate between UD m and SBS i
$r_{i,j}$	The wired transmission rate between SBS i and SBS j
$r_{i,0}$	The the wired transmission rate between SBS i and the MBS
$b_{m,i}^t$	The proportion of bandwidth allocated by SBS i to UD m in time slot t
P_i	The transmission power of SBS i
$G_{m,i}^t$	The channel gains between SBS i and UD m
σ^2	The noise power
$\alpha_i^t(m)$	The content delivery decision of SBS i for the content request of UD m in time slot t
h_i^t	The cache hit rate of SBS i in time slot t

III. System Model and Problem Formulation

In this section, we first introduce the system model of a cooperative edge caching-enabled CDN architecture, and then present the problem formulation in detail. The symbols employed throughout this paper are summarized in Table I.

A. System Model

We consider a multi-tier edge caching-enabled CDN architecture, comprising a MBS and multiple SBSs equipped with ESs, as illustrated in Fig. 1. Let $\mathcal{F} = \{1, 2, \ldots, F\}$ denote the finite set of the content library, where the size of content f is denoted as c_f. Each content is assumed to be indivisible in this paper, and content popularity follows the Mandelbrot-Zipf distribution [44], [45]. The MBS is connected to the Content Service Provider (CSP) to provide all content services. The set of BSs is denoted as $\mathcal{I} = \{0, 1, 2, \ldots, I\}$, where $\{0\}$ represents the set of MBS and $\{1, 2, \ldots, I\}$ represents the set of SBSs. The state of SBS i is represented as $\{C_i, B_i\}$, where C_i and B_i denote the cache capacity and bandwidth of SBS i, respectively. Due to their limited memory size, SBSs can only cache a subset of contents from the content library.

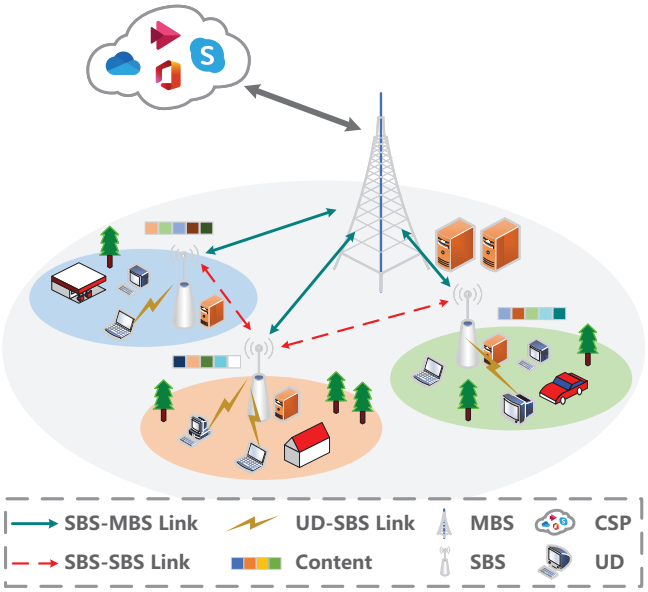

Fig. 1. Overview of the system model.

The system operates in fixed-length time slots $t \in \mathcal{T} = \{1, 2, \ldots, T\}$ with routine updates of caching strategies for all SBSs. The cache state of SBS i in time slot t is denoted by $\mathcal{CS}_i^t = \{cs_i^{1,t}, cs_i^{2,t}, \ldots, cs_i^{F,t}\}$, where $cs_i^{f,t} \in \{0, 1\}$. Here, $cs_i^{f,t} = 1$ signifies that SBS i has cached content f, while $cs_i^{f,t} = 0$ indicates the opposite. The UDs are represented by the set $\mathcal{M} = \{1, 2, \ldots, M\}$ and are randomly distributed within the areas covered by SBSs. They send content requests to their associated SBSs. The position of UD m is described by $pos_{m,i} \in \{0, 1\}$, where $pos_{m,i} = 1$ indicates that UD m is located within the coverage area of SBS i, and SBS i is referred to as the local SBS for UD m. Without considering the range overlap between the areas covered by adjacent SBSs, multiple SBSs cannot simultaneously serve a UD, i.e., $\sum_{i \in \mathcal{I}} pos_{m,i} = 1$. We denote the set of UDs within the coverage area of SBS i as $\mathcal{M}_i = \{m \in \mathcal{M} | pos_{m,i} = 1\}$, where $\mathcal{M}_i \in \mathcal{M}$.

Within the area covered by MBS, each SBS is able to cache certain contents to fulfill content requests from UDs. Specifically, let $q_{m,i}^{f,t} \in \{0, 1\}$ denote the content request of UD m. If $q_{m,i}^{f,t} = 1$, it indicates that UD m is requesting content f from its local SBS in time slot t. Otherwise, $q_{m,i}^{f,t} = 0$. Upon receiving a content request for content f from UD m, SBS i searches for the content in its cache space. If content f is cached in SBS i's cache space, it directly delivers content f to UD m. Otherwise, it proceeds to content f from other SBSs or the MBS.

B. Content Recommendation Model

This paper investigates a content similarity-based recommendation system where various contents exhibit distinct degrees of similarity. Specifically, when UD m requests content f from SBS i and content f is not available in its cache space, SBS i can recommend other cached contents that bear a high degree of similarity.

According to the collaborative filtering algorithm [29], [46], the similarity score between two different contents can be calculated as

$$sim^{f,g} = \frac{\left|\mathcal{M}^f \bigcap \mathcal{M}^g\right|}{\sqrt{|\mathcal{M}^f|}\sqrt{|\mathcal{M}^g|}}, \tag{1}$$

where $|\mathcal{M}^f|$ and $|\mathcal{M}^g|$ represent the number of UDs who desire contents f and g, as well as $|\mathcal{M}^f \bigcap \mathcal{M}^g|$ denotes the number of UDs who desire both. However, variations exist in the behavior and preferences of UDs in different areas. For instance, some UDs may favor a larger variety of contents, while others prefer a more limited selection. These high-activity UDs may overestimate the similarity score for low-activity UDs. Consequently, the same similarity score between two different contents may not be suitable for different UDs. To address this and to personalize the content similarity, we introduce the rating score of content, denoted as λ_m^f, representing UD m's individual rating score for content f. By combining it with $sim^{f,g}$, the new similarity score can be expressed as

$$sim_m^{f,g} = \lambda_m^f sim^{f,g}. \tag{2}$$

Remarkably, the similarity score of content f with itself is 1, i.e., $sim_m^{f,f} = 1$. Subsequently, we build a similarity matrix $\mathcal{X}_m$ for UD m, comprising the set of similarity scores.

It is essential to highlight that not all UDs are inclined to accept recommendations from SBSs. To denote this, we use $w_{m,i}^{f,t} \in \{0,1\}$ to signify whether UD m is willing to accept the recommendation in time slot t. Specifically, $w_{m,i}^{f,t} = 1$ indicates that UD m is open to accepting the recommendation from SBS i or can obtain the requested content f from SBS i, while $w_{m,i}^{f,t} = 0$ signifies that UD m insists on requesting its preferred content.

C. Cache Replacement Model

Given the dynamic content popularity in different periods, the implementation of intelligent content replacement strategies becomes a crucial prerequisite for effectively managing limited cache space [47]–[49]. During the cache replacement phase, each SBS is able to update its cache state. This process enhances overall cache utilization and efficiency, reducing the long-term system overhead associated with content delivery. When an SBS requests contents from the MBS within a specific time slot, it must make decisions regarding whether these contents need to be cached. Furthermore, considering the limited cache space, the SBS must determine whether cached contents need to be replaced and, if so, which of them should be prioritized for replacement.

Let $\beta_i^t = \{\beta_i^{1,t}, \beta_i^{2,t}, \ldots, \beta_i^{F,t}\}$ denote the cache replacement decision of SBS i in time slot t, where $\beta_i^{f,t} \in \{0,1\}$. When $\beta_i^{f,t} = 1$, it indicates that SBS i will replace content f with the requested content from MBS. SBS i replaces nothing when $\sum_{f\in\mathcal{F}} \beta_i^{f,t} = 0$.

D. Communication Model

In this paper, we assume that the UD-SBS link is wireless, while the SBS-SBS and SBS-MBS links are wired. The delivery delay of content f via the UD-SBS link is denoted by $L_{m,i}^{f,t}$, which can be given by

$$L_{m,i}^{f,t} = \frac{c_f}{r_{m,i}^t}, \tag{3}$$

where $r_{m,i}^t$ is the wireless downlink transmission rate between UD m and SBS i in time slot t, which can be expressed as

$$r_{m,i}^t = b_{m,i}^t B_i log_2\left(1 + \frac{P_i|G_{m,i}^t|^2}{\sigma^2 + \sum_{m'\in\mathcal{M}_i} P_i|G_{m',i}^t|^2}\right), \tag{4}$$

where P_i denotes the transmission power of SBS i. $G_{m,i}^t$ and $G_{m',i}^t$ indicate the channel gains between SBS i and UD m and between SBS i and other UDs within the scope of SBS i, respectively. σ^2 represents the noise power. $b_{m,i}^t$ denotes the proportion of bandwidth allocated by SBS i to UD m in time slot t, and $b_{m,i}^t \in [0,1]$.

The delivery delay of content f among SBSs can be calculated by

$$L_{i,j}^{f,t} = \frac{c_f}{r_{i,j}}, \tag{5}$$

where $r_{i,j}$ denotes the wired transmission rate between SBS i and SBS j. The delivery delay of content f via the SBS-MBS link is represented by $L_{i,0}^{f,t}$, which can be expressed as

$$L_{i,0}^{f,t} = \frac{c_f}{r_{i,0}}, \tag{6}$$

where $r_{i,0}$ denotes the wired transmission rate between SBS i and the MBS. Because the data size of a content request is much smaller than the content itself, the uplink link delivery delay is ignored in this paper.

The content delivery decision of SBS i for the content request of UD m in time slot t can be represented as $\alpha_i^t(m) = \{\alpha_{i,i}^t, \alpha_{i,j}^t, \alpha_{i,0}^t\}$, where $\alpha_{i,i}^t$, $\alpha_{i,j}^t$, and $\alpha_{i,0}^t \in \{0,1\}$. Since contents are indivisible, it is assumed that the content request of each UD can only be fulfilled in one way, i.e., $\alpha_{i,i}^t + \alpha_{i,j}^t + \alpha_{i,0}^t = 1$. Then, the different delivery decisions can be analyzed in detail as follows:

- $\alpha_{i,i}^t = 1$ indicates that SBS i has cached the content requested by UD m or other contents with high similarity in the local cache space during time slot t. If SBS i possesses the requested content, it transmits the content to UD m via a wireless link. In the absence of the requested content but with another content bearing high similarity, SBS i can recommend those alternatives to UD m to fulfill the content request. The content delivery delay in this case is denoted as $L_{m,i}^{f,t}$.
- $\alpha_{i,j}^t = 1$ signifies that SBS i cannot fulfill the content request of UD m during time slot t and must seek the content from other SBSs. To simplify the search process, SBS i only requests content from its nearest neighbor, termed the adjacent SBS. The adjacent SBS j delivers the content to UD m via SBS i if it has the requested content or similar alternatives. The total delay for content delivery in this scenario is $L_{m,i}^{f,t} + L_{i,j}^{f,t}$.
- $\alpha_{i,0}^t = 1$ indicates that SBS i is unable to fulfill the content request of UD m during time slot t. Consequently, SBS i must request the content from the MBS. Since all

contents are available in the MBS, content recommendation is not considered when making requests to the MBS. The total delay for content delivery in this case is $L_{m,i}^{f,t} + L_{i,0}^{f,t}$.

Therefore, the delivery delay of content f requested by UD m in time slot t can be given by

$$\begin{aligned} L_m^t = \sum_{f\in\mathcal{F}} q_{m,i}^{f,t} \big[& \alpha_{i,i}^t L_{m,i}^{f,t} + \alpha_{i,j}^t (L_{m,i}^{f,t} + L_{i,j}^{f,t}) \\ & + \alpha_{i,0}^t (L_{m,i}^{f,t} + L_{i,0}^{f,t}) \big]. \end{aligned} \tag{7}$$

In our model, we refer to the proportion of content requests fulfilled by SBSs as the cache hit rate, i.e., $\alpha_{i,i}^t = 1$ or $\alpha_{i,j}^t = 1$. Then, the cache hit rate can be expressed as

$$h_i^t = \frac{\sum_{m\in\mathcal{M}_i} \sum_{f\in\mathcal{F}} q_{m,i}^{f,t} w_{m,i}^{f,t} \left(\alpha_{i,i}^t + \alpha_{i,j}^t\right)}{\sum_{m\in\mathcal{M}_i} \sum_{f\in\mathcal{F}} q_{m,i}^{f,t}}. \tag{8}$$

E. Problem Formulation

In this paper, our dual objective is to minimize content delivery delay while simultaneously maximizing the cache hit rate. Solving these two optimization goals concurrently poses a significant challenge. In an ideal scenario, where SBSs can fulfill all content requests locally, the content delivery delay is minimized, resulting in a high cache hit rate, and vice versa. Consequently, optimizing for reducing overall content delivery delay inherently leads to an increased cache hit rate. The optimization problem can be reformulated as minimizing content delivery delay across all SBSs while satisfying certain constraints, which can be described as

$$\begin{aligned} \min_{\substack{\alpha_i^t,\beta_i^t,\\ b_{m,i}^t}} \ & \lim_{T\to\infty} \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{m\in\mathcal{M}_i} L_m^t \\ s.t. \ & c01: \alpha_{i,i}^t, \alpha_{i,j}^t, \alpha_{i,0}^t, \in \{0,1\}, \forall i, \forall j, \forall t, \\ & c02: \beta_i^{f,t} \in \{0,1\}, \forall i, \forall f, \forall t, \\ & c03: b_{m,i}^t \in [0,1], \forall i, \forall m, \forall t, \\ & c04: \alpha_{i,i}^t + \alpha_{i,j}^t + \alpha_{i,0}^t = 1, \forall i, \forall j, \forall t, \\ & c05: \sum_{f\in\mathcal{F}} \beta_i^{f,t} \le 1, \forall i, \forall t, \\ & c06: \sum_{m\in\mathcal{M}_i} b_{m,i}^t \le 1, \forall i, \forall t, \\ & c07: \sum_{f\in\mathcal{F}} cs_i^{f,t} c_f \le C_i, \forall i, \forall t, \\ & c08: sim_m^{f,g} \ge st^{min}, \forall m, \forall f, \forall g, \\ & c09: h_i^t \ge h^{min}, \forall i, \forall t, \end{aligned} \tag{9}$$

where $c01$ and $c04$ correspond to the content delivery decisions of SBS i, where each UD's content request can only be fulfilled by a single decision per time slot. $c02$ and $c05$ represent cache replacement decisions, allowing SBS i to replace one content per time slot. $c03$ and $c06$ indicate the bandwidth allocation strategy for UD m, with the sum of allocated bandwidths constrained not to exceed the upper limit B_i in each time slot. $c07$ ensures that the total size of cached contents does not surpass the cache capacity C_i. $c08$ sets a limit on the similarity between the requested content f and the recommended content g, ensuring it is not lower than the similarity tolerance st^{min}. $c09$ mandates that the cache hit rate of SBS i should surpass the minimum threshold h^{min}.

F. Problem Analysis

To address the optimization problem described above, it is necessary to find the optimal decision of $\alpha_i^t(m)$, β_i^t, and $b_{m,i}^t$ in each time slot. However, these variables are highly dynamic and discrete, and the complexity of the problem grows exponentially as the number of contents, or SBSs, increases. Meanwhile, $\alpha_i^t(m)$ and β_i^t are integer variables, while $b_{m,i}^t$ is a continuous variable. It can be determined that the objective function is a Mixed Integer NonLinear Programming (MINLP) issue, which is NP-hard [23]. Furthermore, in a more realistic scenario, the prior information about content request patterns over time is not known in advance. Traditional methods struggle to adapt to dynamic environments and make intelligent caching decisions. Thus, a DRL-based method is adopted here to solve the optimal decision problem.

IV. Single-agent DRL-based Method

In this section, we analyze the optimization problem mentioned above and propose a single-agent DRL-based method to solve the corresponding problem. Specifically, we first discuss how to approximate the decision-making process of agents as a POMDP, and then provide detailed explanations of parameter updates for agents during the training process.

A. Problem Transformation

Given the decentralized CDN environment, which comprises multiple decision-makers without a central controller, each SBS lacks direct access to the complete state of the environment. Instead, these entities rely on observation sequences to glean insights into the system. Moreover, to mitigate additional bandwidth and time costs, we operate under the assumption that there is no information exchange between different agents during the training stage [50]. In other words, each SBS remains unaware of decisions made by others. Consequently, we re-model the decision-making process for each SBS as a POMDP and design a single-agent DRL-based method to address the POMDP, where each SBS acts as an agent to learn the optimal policy. Each critical element of POMDP is defined detailedly as follows:

1) State Space: Let $\mathcal{S} = s_1^t, s_2^t, \ldots, s_I^t$ denote a specific condition or situation of the system, where s_i^t represents the set of possible observations that agent i can observe. In this paper, the state information observed by agent i in time slot t can be denoted as

$$s_i^t = \left\{\mathcal{CS}_i^t, \{q_{m,i}^{f,t}, w_{m,i}^{f,t} | m \in \mathcal{M}_i\}\right\}, \tag{10}$$

where $\{q_{m,i}^{f,t}, w_{m,i}^{f,t} | m \in \mathcal{M}_i\}$ denotes the set of content requests and decisions on whether UDs accept recommendations.

2) Action Space: The set of actions available for each agent is denoted as a_i^t. In this paper, each agent is tasked not only with making decisions regarding content delivery and cache replacement but also with determining the allocation of bandwidth to UDs. Therefore, the action space of agent i can be expressed as

$$a_i^t = \left\{\{\alpha_i^t(m), b_{m,i}^t | m \in \mathcal{M}_i\}, \beta_i^t\right\}, \tag{11}$$

where $\{\alpha_i^t(m), b_{m,i}^t | m \in \mathcal{M}_i\}$ denotes the set of content delivery decisions and bandwidth allocation strategies.

3) Reward Function: r_i^t defines the immediate reward or cost associated with executing an action in the given state, which can be used to measure the quality of actions. As our goal is to minimize the content delivery delay for all SBSs, the reward function can be set as follows:

$$r_i^t(s_i^t, a_i^t) = -\sum_{m \in \mathcal{M}_i} L_m^t - Pe, \tag{12}$$

where Pe is the penalty when agent i violates constraints, which can reflect the undesirable consequences or costs of performing an action or reaching a certain state.

In our proposed model, each agent learns adaptively and makes decisions through interactions with the environment in each time slot. For instance, agent i observes its observation s_i^t and selects an action a_i^t based on its policy π_i in time slot t. Subsequently, after all agents take their respective actions, the current state $\mathcal{S}^t$ transitions to the next state $\mathcal{S}^{t+1}$. The probability of transitioning from $\mathcal{S}^t$ to $\mathcal{S}^{t+1}$ can be given by

$$\mathcal{P}_{ss'} = \mathbb{P}[\mathcal{S}^{t+1} = s' | \mathcal{S}^t = s]. \tag{13}$$

Let U^t represent the reward in time slot t, which can be calculated by

$$U^t = r^{t+1} + \gamma r^{t+2} + \cdots = \sum_{\tau=0}^{\mathcal{T}} (\gamma)^\tau r^{t+\tau+1}, \tag{14}$$

where $\gamma \in [0, 1]$ denotes the discount factor, indicating the importance agents attach to future rewards. A small value of γ tends to make agents more inclined to adopt short-term strategies, implying a focus on maximizing immediate rewards. Conversely, when γ is large, agents may also lean towards adopting long-term strategies to maximize future rewards. The state-value function is used to estimate the state, which is given as

$$V_\pi(s) = \mathbb{E}[U^t | \mathcal{S}^t = s, \pi]. \tag{15}$$

Then, according to the Bellman Equation, we can transform the state-value function into the following equation by combining Eqs. (14) and (15), which can be expressed as

$$\begin{aligned} V_\pi(s) &= \mathbb{E}\left[U^t | \mathcal{S}^t = s, \pi\right] \\ &= \mathbb{E}\left[r^{t+1} + \gamma r^{t+2} + (\gamma)^2 r^{t+3} + \ldots | \mathcal{S}^t = s, \pi\right] \\ &= \mathbb{E}\left[r^{t+1} + \gamma (r^{t+2} + \gamma r^{t+3} + \ldots) | \mathcal{S}^t = s, \pi\right] \\ &= \mathbb{E}\left[r^{t+1} + \gamma U^{t+1} | \mathcal{S}^t = s, \pi\right] \\ &= \mathbb{E}\left[r^{t+1} + \gamma V_\pi(s') | \mathcal{S}^t = s, \mathcal{S}^{t+1} = s', \pi\right]. \end{aligned} \tag{16}$$

According to Eq. (13), we can further transform Eq. (16) into the following equation:

$$V_\pi(s) = r(s) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'} V_\pi(s'). \tag{17}$$

Thus, our objective is to find an optimal policy π^* to maximize the cumulative discount reward, which can be given by

$$V_{\pi^*}(s) = \max_\pi V_\pi(s). \tag{18}$$

B. DDPG-based Method

Given the challenges posed by the large and dynamic network environment, solving Eq. (21) becomes a formidable task. To address this, we employ a DDPG-based method. The fundamental concept of DDPG involves optimizing the policy to enhance the decision-making capabilities of each agent. This policy optimization is facilitated through the use of two neural network models: Critic and Actor networks. The Critic network, denoted as $Q(\cdot|\theta^Q)$, evaluates the actions selected by the Actor network and provides guidance for policy updates. Simultaneously, the Actor network, represented as $\mu(\cdot|\theta^\mu)$, learns a policy function π that maps states to corresponding actions, aiming to maximize cumulative rewards. DDPG introduces the target Actor network $\mu'(\cdot|\theta^{\mu'})$ and the target Critic network $Q'(\cdot|\theta^{Q'})$ to enhance the stability of the learning process.

Similar to Eq. (17), we can derive the Q-value function, which represents the reward $Q(\mathcal{S}, \mathcal{A})$ obtained after taking action $\mathcal{A}$ in state $\mathcal{S}$,

$$Q_\pi(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \sum_{a' \in \mathcal{A}} \pi(a'|s') Q_\pi(s', a'). \tag{19}$$

After that, the relationship between the state-value function and Q-value function can be gotten as

$$V_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) Q_\pi(s, a). \tag{20}$$

In POMDP, the current state is only related to the previous state. Therefore, the action that maximizes the value function in the current state can maximize the cumulative reward in the current state. Thus, the optimization problem in Eq. (19) can be transformed into

$$V_{\pi^*}(s) = \max_{a \in \mathcal{A}} Q_{\pi^*}(s, a). \tag{21}$$

The traditional Q-learning method employs a Q-table to store the value of the Bellman Equation and then iterates continuously to update the Q-values. However, as the state space and action space expand, maintaining the Q-table becomes increasingly challenging. In our method, DDPG uses the Critic network to approximately estimate the Q-value, which is given as

$$Q\left(s, \mu(s|\theta^\mu)|\theta^Q\right) \approx Q_\pi(s, a), \tag{22}$$

where θ^Q and θ^μ are parameters of the Critic network and Actor network, respectively. $\mu(\cdot|\theta^\mu)$ is the Actor network. In order to obtain a more accurate Q-value, the Critic network

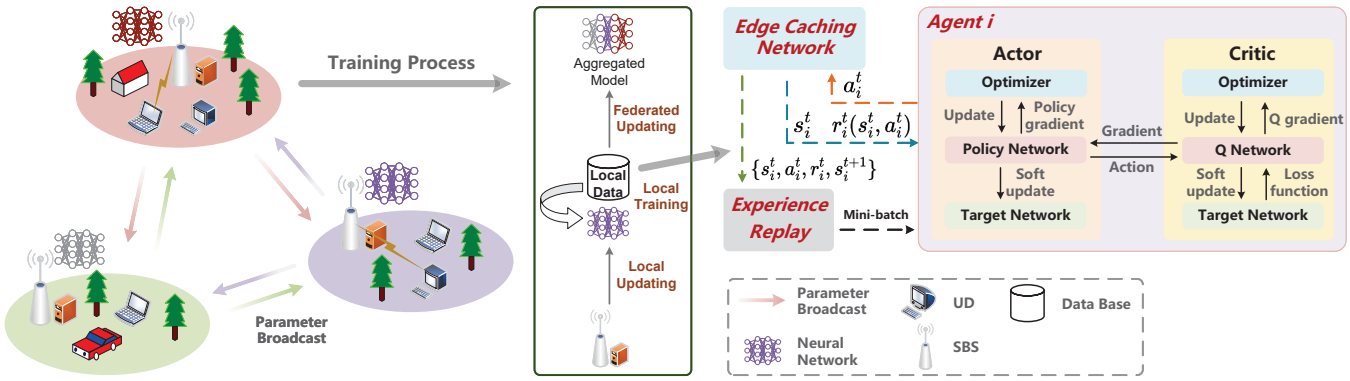

Fig. 2. Overview of the proposed FD3PG.

Algorithm 1: Experience Replay Procedure

foreach agent i **do**
  if *the number of transitions* $\leq |\mathcal{E}_i|$ **then**
    Randomly sample $\{s_i^t, a_i^t, r_i^t, s_i^{t+1}\}$ from $\mathcal{E}_i$;
    Calculate the target Q-value via Eq. (23);
    Update the Critic network by minimizing the loss function using Eq. (24);
    Update the Actor network by maximizing the evaluation according to Eq. (25);
    Update the target networks using Eq. (26).
  end
end

aims to minimize the difference between estimated value $Q(s, a|\theta^Q)$ and target value, which can be calculated by

$$y = r + \gamma Q'\left(s', \mu'(\cdot|\theta^{\mu'})|\theta^{Q'}\right), \tag{23}$$

where $\theta^{Q'}$ and $\theta^{\mu'}$ are parameters of the target Critic network and target Actor network. $\theta^{\mu'}$ is the target Actor network. The loss function of the Critic network can be expressed as

$$\mathcal{L}(\theta^Q) = \frac{1}{T}\sum_{t=0}^{T}\left(y^t - Q(s^t, a^t|\theta^Q)\right)^2. \tag{24}$$

On the other hand, the Actor network aims to choose the optimal action. Therefore, the update goal of the network is to maximize the Q-value by gradient backpropagation, and its gradient is expressed as follows:

$$\nabla_{\theta^\mu}\mathcal{J} \approx \frac{1}{T}\sum_{t=0}^{T}\underbrace{\nabla_a Q(s, a|\theta^Q|_{s=s^t, a=\mu(s|\theta^\mu)})}_{\partial Q/\partial\mu}\underbrace{\nabla_{\theta^\mu}\mu(s|\theta^\mu)}_{\partial\mu/\partial\theta^\mu}. \tag{25}$$

Finally, the target networks of Critic and Actor networks can be updated, respectively, as follows:

$$\begin{aligned}\theta^Q &\leftarrow \varsigma\theta^Q + (1-\varsigma)\theta^{Q'},\\ \theta^\mu &\leftarrow \varsigma\theta^\mu + (1-\varsigma)\theta^{\mu'},\end{aligned} \tag{26}$$

where ς is the update weight.

Additionally, experience replay technology is incorporated to enhance the utilization rate of data samples and training stability. Each agent observes the state s^t, selects an action a^t based on the current policy π, and receives an immediate reward r^t. The current state s^t transitions to the next state s^{t+1} after the agent interacts with it. These experiences are then encapsulated into a tuple $\{s^t, a^t, r^t, s^{t+1}\}$ and stored in the replay buffer $\mathcal{E}$. During training, each agent can randomly sample experiences from its own buffer. The experience replay procedure is summarized in Algorithm 1.

V. Federated Distributed DDPG-based Method

As mentioned above, we embrace a cooperative model wherein multiple agents coexist simultaneously. However, single-agent DRL-based methods frequently face challenges in multi-agent scenarios and may converge to local optimal solutions. Many existing multi-agent DRL-based methods often follow a training strategy of centralized learning and distributed execution. This strategy necessitates cross-communication between agents to share their intelligence and achieve the optimal global solution, which could increase the communication load.

To address this challenge, motivated by FL principles [51], [52], we propose FD3PG, which allows all agents to share their intelligence and carry out federated updating after each Δ episodes, as shown in Fig. 2. Because the data size of model parameters is smaller than that of models, in order to reduce the communication cost, we use the model parameters as the intelligence of each agent in this paper. According to the proposed updating rule, each agent maintains the parameters with weight ω and mixes the parameters of others, which can be expressed as:

$$\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t \cdot \Omega, \tag{27}$$

where $\boldsymbol{\theta}^t = [\theta_1^t, \theta_2^t, \ldots, \theta_I^t]$ denotes the vector of network parameters of all agents at the t-th episode. Ω denotes the weight matrix, which will be given in the following.

However, the distribution of UDs in different areas varies significantly, resulting in considerable heterogeneity between the data samples owned by each UD. To illustrate, consider a real-life scenario: UDs located in a school may request content that is completely different from those located in a park. This diversity leads to significant differences in the decisions made by the SBS deployed near the school and the SBS deployed near the park in each time slot. Consequently, these two SBSs

Algorithm 2: FD3PG: **F**ederated **D**istributed **D**eep **D**eterministic **P**olicy **G**radient

Input: state space, action space, replay buffer size, mini-batch size.
Output: optimal action for content delivery, cache replacement, and bandwidth allocation.

Initialize network parameters θ_i^Q and θ_i^μ;
Initialize target network parameters $\theta_i^{Q'} \leftarrow \theta_i^Q$ and $\theta_i^{\mu'} \leftarrow \theta_i^\mu$;
Instantiate replay buffer $\mathcal{E}_i$ for each agent i;
Build similarity matrix $\mathcal{X}_m$;
foreach *episode* **do**
  Initialize the action exploration ϵ;
  Recevie initial the state $\mathcal{S}$;
  Stage 1: Model Training
  foreach time slot *t* **do**
    foreach agent *i* **do**
      if $exp < \epsilon$ **then**
        Randomly select action a_i^t;
      end
      else
        Observes state s_i^t;
        Select action $a_i^t = \mu(s_i^t|\theta_i^\mu)$;
      end
      Execute action a_i^t to obtain an immediate reward r_i^t and a new state s_i^{t+1};
      Store transition $\{s_i^t, a_i^t, r_i^t, s_i^{t+1}\}$ into $\mathcal{E}_i$;
      if *the number of transitions* $\geq |\mathcal{E}_i|$ **then**
        Replace the oldest experience;
      end
    end
    Run replay procedure in Algorithm 1;
  end
  Stage 2: Federated Updating
  if *episode* mod Δ == 1 **then**
    foreach agent *i* **do**
      Broadcast model parameters and calculate the variation values between agent i's model and other models according to Eq. (31);
      Calculate the aggregation weight between agent i's model and other models using Eq. (32);
      Run federated updating using Eq. (27).
    end
  end
end

need to train different decision-making models. However, the training strategy adopted by traditional DRL algorithms results in each agent having the same decision-making model. When deployed in different environments, this uniformity can lead to suboptimal decision-making effects for each agent.

To tackle this problem, inspired by [53], [54], we design an EMD-based method to measure the variations between different models and calculate the update weights of different model parameters. EMD is a metric used to measure the distance between two probability distributions. It describes the minimum cost required to transform one distribution into another. This cost is typically interpreted as the minimum total cost of moving the mass of one distribution from one position to another.

Specifically, we take a set of sample data as input and use models X and Y to obtain their action outputs. Then, we extract eigenvectors from the output action vectors and define the signature of sample data $d \in \mathcal{D}$ on model X as

$$X_d = \{(x_1, l_{x_1}), (x_2, l_{x_2}), \ldots, (x_N, l_{x_N})\}, \tag{28}$$

where x_n represents the n-th eigenvalue, and l_{x_n} represents the corresponding label of the n-th eigenvalue. $\mathcal{D}$ denotes the dataset. The signature of model Y is the same as above, which can be expressed as

$$Y_d = \{(y_1, l_{y_1}), (y_2, l_{y_2}), \ldots, (y_N, l_{y_N})\}. \tag{29}$$

In this paper, we use Euclidean distance as a metric for the EMD value to measure the variation between two feature values. Then, the EMD value between models X and Y can be calculated as

$$EMD_{x,y} = \sqrt{x^2 - y^2}, x \in X_d, y \in Y_d. \tag{30}$$

Therefore, the variation value between models X and Y is defined as

$$Var(X, Y) = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} \sum_{n=1}^{N} EMD_{x,y}, \tag{31}$$

where $|\mathcal{D}|$ indicates the size of $\mathcal{D}$. After receiving the model parameters broadcast by other agents, agent i calculates the variation values between its own model and others, selectively aggregating a subset of these models. Specifically, let $\mathbf{u}_i^t = \{u_{i,1}^t, \ldots, u_{i,i-1}^t, u_{i,i+1}^t, \ldots, u_{i,I}^t\}$ represent the set of variation values between agent i's model and the other models. Then, models with variation values less than the threshold k are chosen for weighted aggregation. The threshold is dynamically determined based on the number of agents participating in training. In scenarios with fewer agents in the network, ensuring the model's performance necessitates reducing the threshold for model selection appropriately. This adjustment allows each agent to aggregate more parameters, thereby compensating for the limited dataset size. Conversely, in situations with a larger number of agents, concerns about insufficient sample aggregation diminish. The aggregation weight between agents i and j is represented as $\varphi_{i,j}$, which can be expressed as follows:

$$\varphi_{i,j} = \begin{cases} \frac{1-\omega}{Var(X,Y)}, & \text{if } u_{i,j} \leq k \\ 0. & \text{else} \end{cases} \tag{32}$$

After that, the weight matrix of agent i can be given by

$$\Omega_i = \begin{bmatrix} \omega & \varphi_{1,2} & \cdot & \cdot & \cdot & \varphi_{1,I} \\ \varphi_{2,1} & \omega & \cdot & \cdot & \cdot & \varphi_{2,I} \\ \cdot & \cdot & \cdot & & & \cdot \\ \cdot & \cdot & & \cdot & & \cdot \\ \cdot & \cdot & & & \cdot & \cdot \\ \varphi_{I,1} & \varphi_{I,2} & \cdot & \cdot & \cdot & \omega \end{bmatrix}. \tag{33}$$

TABLE II
PAREMETERS SETTINGS

Paremeter	Defintion	Value
I	The number of SBS	$[5, 10]$
c_f	The size of content	$[5, 10]$ MB
C_i	The cache capacity of SBS	100 MB
B_i	The bandwidth of SBS	20 MHz
P_i	The wireless transmission power	38 dBm
σ^2	The noise power	-95 dBm
$r_{i,0}$	The transmission rate of SBS-MBS link	50 Mbps

The details of the proposed FD3PG are outlined in Algorithm 2. The main steps of this method are as follows: In lines 1 to 4, the Critic and Actor networks, along with the corresponding target networks, are randomly initialized. Simultaneously, a similarity matrix is built for each UD, and the experience replay buffer is instantiated. In lines 10 to 17, agent i observes the state information s_i^t and selects the action a_i^t based on the policy π_i in time slot t. If the exploration value exp is less than the threshold ϵ, agent i randomly selects actions. In lines 18 to 22, agent i obtains the reward r_i^t after executing the action, and the state transitions to the next state s_i^{t+1}. These experience tuples $\{s_i^t, a_i^t, r_i^t, s_i^{t+1}\}$ are stored in the experience replay buffer $\mathcal{E}_i$. The Critic and Actor networks are trained using Algorithm 1 in line 24. In lines 27 to 33, each agent broadcasts its model parameters to other agents and calculates variation values between two different models using Eq. (31) after each Δ episodes. Agents sort these models based on these variation values and perform federated updating using Eq. (27).

VI. PERFORMANCE EVALUATION

In this section, we conduct extensive simulations to verify the performance of the proposed FD3PG.

A. Simulation Settings

We validate the effectiveness and convergence performance of the proposed method using both a Synthetic dataset and the MovieLens Latest Small (MLS) dataset. For the Synthetic dataset, the network comprises 100 UDs, and the content library contains 200 to 400 contents, depending on the simulation setting. On the other hand, the MLS dataset consists of 600 UDs and 6000 to 9000 contents, again depending on the simulation setting. We model the wireless channel as a block Rayleigh fading channel following the Okumura-Hata path loss model. The channel gain is calculated as $G_{m,i}^t = 30.6 + 36.7log_{10}$ dBm. In the proposed method, we set the size of the experience replay buffer to 1000, the size of the mini-batch to 64, and the discount factor $\gamma = 0.99$. The simulation parameters are summarized in Table II.

For comparing the performance of the proposed FD3PG, we introduce the following benchmark baselines:

- **FADE:** This method combines the DDQN algorithm and FL to solve the problem of convergence to local optima by parameter aggregation [58].
- **MADRL:** In this method, each agent trains a Critic with global information and an Actor with local information. The environment interacts with the actions of all agents simultaneously, generating distinct rewards for them in each time slot [56].
- **DDPG:** This method utilizes a DRL algorithm based on the Actor-Critic framework.
- **Deep Q-Network (DQN):** This method employs a DRL algorithm that combines deep neural networks with the Q-learning algorithm. Each agent makes decisions based on the DQN method in each time slot. Each agent makes decisions based on the DDPG method in each time slot.
- **Least Recently Used (LRU):** The underlying principle of this method is that the least recently used content is unlikely to be used in the future. Each agent employs the LRU method as the cache replacement strategy, replacing the least recently used content in each time slot [57].
- **Least Frequently Used (LFU):** The underlying principle of this method is that frequently accessed data is likely to continue being accessed in the future. Each agent adopts the LFU method as the cache replacement strategy, replacing the least frequently used content in each time slot [59].

Besides, there are two metrics used to estimate the performance quantitatively:

a) average delivery delay: The average delivery delay for all requests in each time slot;

b) cache hit rate: The definition is given by Eq. (8).

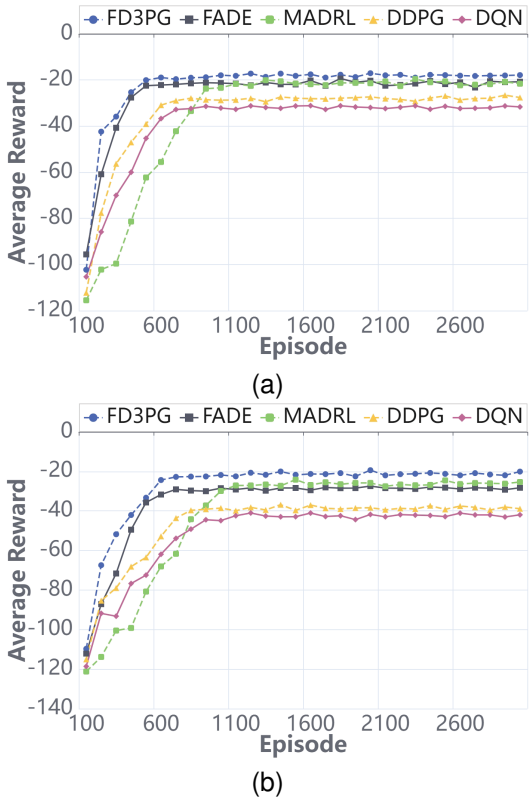

Fig. 3. Convergence curves of methods based on FADE, MADRL, DDPG, DQN, and the proposed FD3PG on (a) the Synthetic dataset and (b) the MLS dataset.

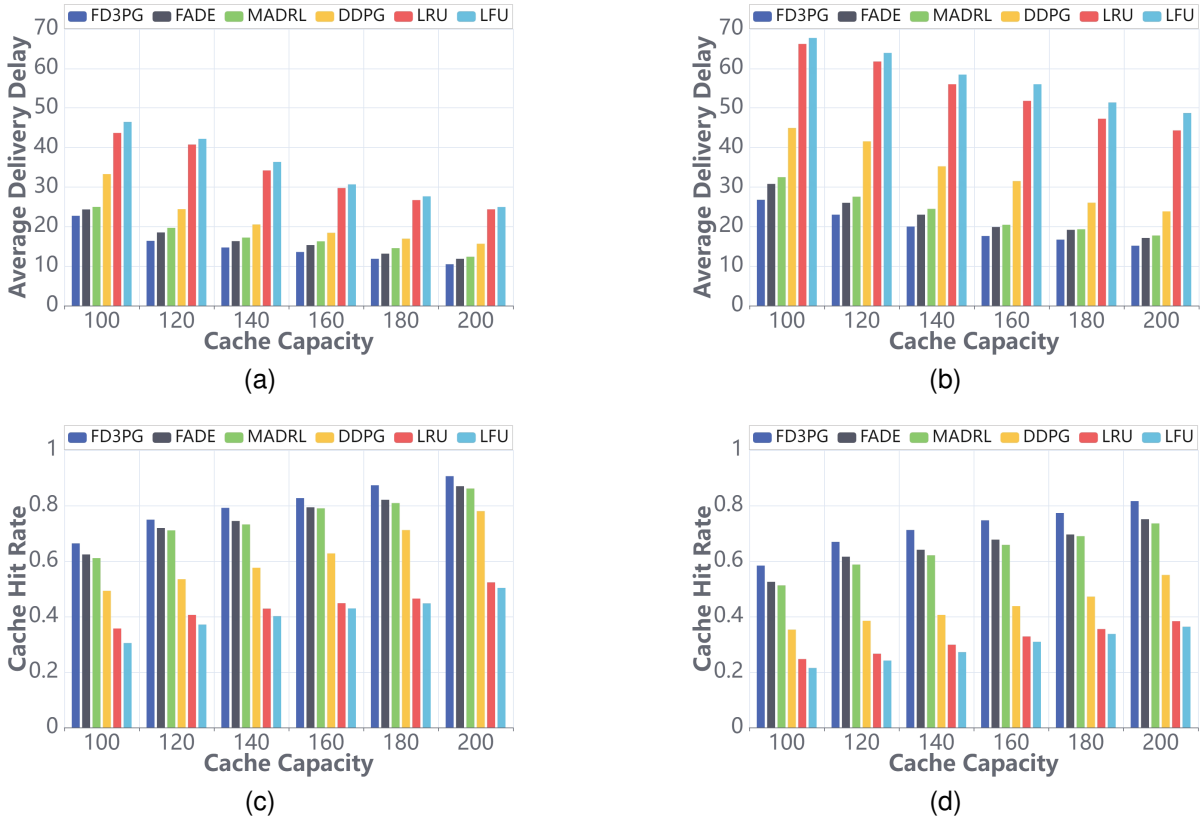

Fig. 4. Average delivery delay versus different cache capacities of SBS on (a) the Synthetic dataset and (b) the MLS dataset. Cache hit rate versus different cache capacities of SBS on (c) the Synthetic dataset and (d) the MLS dataset.

B. Performance Comparison

1) The convergence performance: As shown in Fig. 3, we present the convergence curves of methods based on DDPG, DQN, MADRL, FADE, and the proposed FD3PG. Fig. 3(a) displays the convergence curves of these methods on the Synthetic dataset, with a content library size of 200 and a cache capacity of 100 MB set for each SBS. It is noticeable that the initial rewards of these methods are relatively low, but rapidly increase as the number of episodes grows. The proposed FD3PG starts converging to a relatively stable value after approximately 500 episodes, outperforming other methods in terms of runtime and rewards.

Specifically, single-agent DRL-based methods, such as DDPG and DQN, can only observe local state information in multi-agent systems and utilize limited information for model training. Due to the absence of comprehensive state information, the decision-making model of each agent can easily converge to local optima, leading to suboptimal decisions that fail to maximize the utility of the entire system. Consequently, when multiple such agents exist in the system, their decisions collectively diminish the overall effectiveness of the system. In contrast, MADRL, which employs centralized training, achieves better model performance. However, centralized training necessitates agents employing a larger and more complex neural network to capture relationships between numerous global input states and the local policies of each agent. Consequently, this method significantly increases the scale and complexity of neural networks, resulting in slower training speeds. FADE combines DRL with FL, using parameter aggregation to address the issue of convergence on local optima and achieves faster training speeds. However, it does not consider the impact of agent personalization on model performance. Although parameter aggregation can improve model performance, the reduction in agent personalization still negatively affects model performance, especially when agents are in different environments. In the proposed FD3PG, agents engage in distributed training based on their local observations and periodically broadcast their model parameters to others for aggregation, without relying on a centralized server. Additionally, we fully consider the impact of agent personalization on model performance and utilize the EMD-based model selection method, further enhancing the effectiveness of parameter aggregation. Continuing to Fig. 3(b), we observe that all five methods converge much more slowly on the MLS dataset compared to the Synthetic dataset. It is worth noticing that, even in this scenario, the proposed FD3PG consistently outperforms other methods.

2) The impact of the cache capacity of SBS: The impact of varying cache capacities of SBS on the performance of different methods is explored, with each SBS's cache capacity ranging from 100 to 200 MB, as depicted in Fig. 4. Fig. 4(a) illustrates the effect of different cache capacities of SBS on the average delivery delay for the Synthetic dataset. As the cache capacity of SBS increases, FD3PG consistently outperforms all

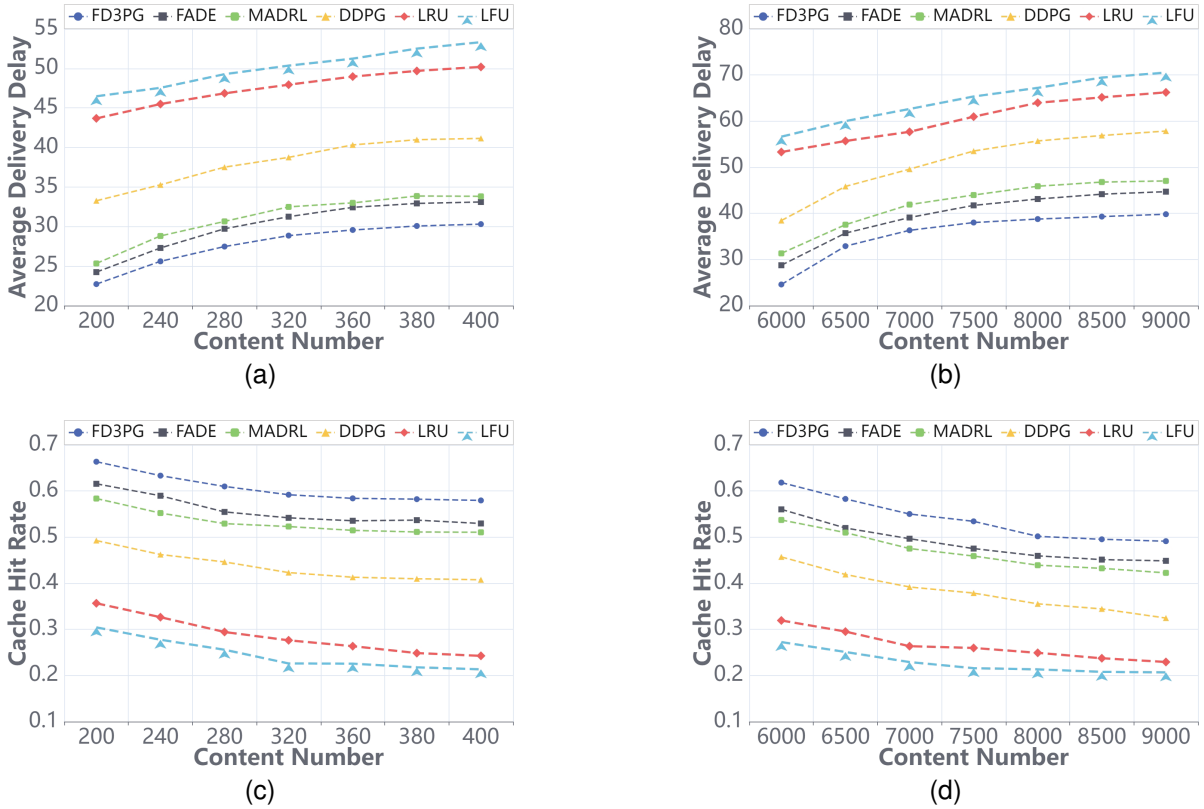

Fig. 5. Average delivery delay versus different numbers of content on (a) the Synthetic dataset and (b) the MLS dataset. Cache hit rate versus different numbers of content on (c) the Synthetic dataset and (d) the MLS dataset.

other methods in terms of average delivery delay. In the initial stage, FD3PG achieves the lowest average delay, reducing it by approximately 10%, 11% and 35% compared to FADE, MADRL and DDPG, not to mention the simple rule-based methods, LFU and LRU. Moving on to Fig. 4(b), we observe the impact of different cache capacities of SBS on the average delivery delay for the MLS dataset. The average delivery delay of all methods on this dataset is higher than that on the Synthetic dataset. Nevertheless, our proposed FD3PG still reduces the average delivery delay by approximately 12%, 14%, 36%, 53%, and 62% compared to FADE, MADRL, DDPG, LFU, and LRU, respectively.

Furthermore, an increased cache capacity has a positive impact on the cache hit rate across different datasets, as illustrated in Figs. 4(c) and 4(d). All methods, particularly DRL-based ones, exhibit an upward trend in cache hit rate with an increase in the cache capacity of each SBS. This trend is attributed to the ability of SBSs to cache more content from the library as their cache capacity grows, leading to a higher satisfaction of content requests. Considering cooperative caching among SBSs, the surplus cache capacity allows them to assist other SBSs in fulfilling more content requests, reducing duplicate content delivery and minimizing the necessity to fetch content from the MBS. Consequently, it is logical that all methods achieve lower average delivery delay and a higher cache hit rate with an increase in cache capacity.

It is noteworthy that agents in FD3PG, FADE, and MADRL can learn from their own environment while also accounting for the impact of decisions made by other agents. This ability allows them to strike a balance between sacrificing some local cache hit rate and sharing cache capacity to process content requests from adjacent SBSs. In contrast, agents in single-agent-based methods, like DDPG, can only observe limited local state information and make decisions based on that. Furthermore, as mentioned earlier, there is no information exchange between agents in single-agent methods. Therefore, they can only focus on the impact of their own decisions on the environment, making them prone to falling into local optimal solutions. Their efficiency of complex datasets is unstable, leading to unsatisfactory results. Additionally, simple rule-based methods, such as LFU and LRU, struggle to fulfill content requests in dynamic and complex environments.

3) The impact of the number of content: Fig. 5 illustrates the performance of various methods with different content numbers. The number of contents ranges from 200 to 400 on the Synthetic dataset and from 6000 to 9000 on the MLS dataset, while the initial cache capacity of each SBS is set to 100 MB. As shown in Figs. 5(a) and 5(b), the average delivery delay for all methods experiences a slight increase with the rise of the content number. Additionally, all methods perform worse on the MLS dataset than on the Synthetic dataset. This trend indicates that the diversity of content requests increases as the number of contents in the library grows. However, SBS cannot guarantee that the cached content can fulfill various content requests in each time slot due to limited local cache space. This reduces the

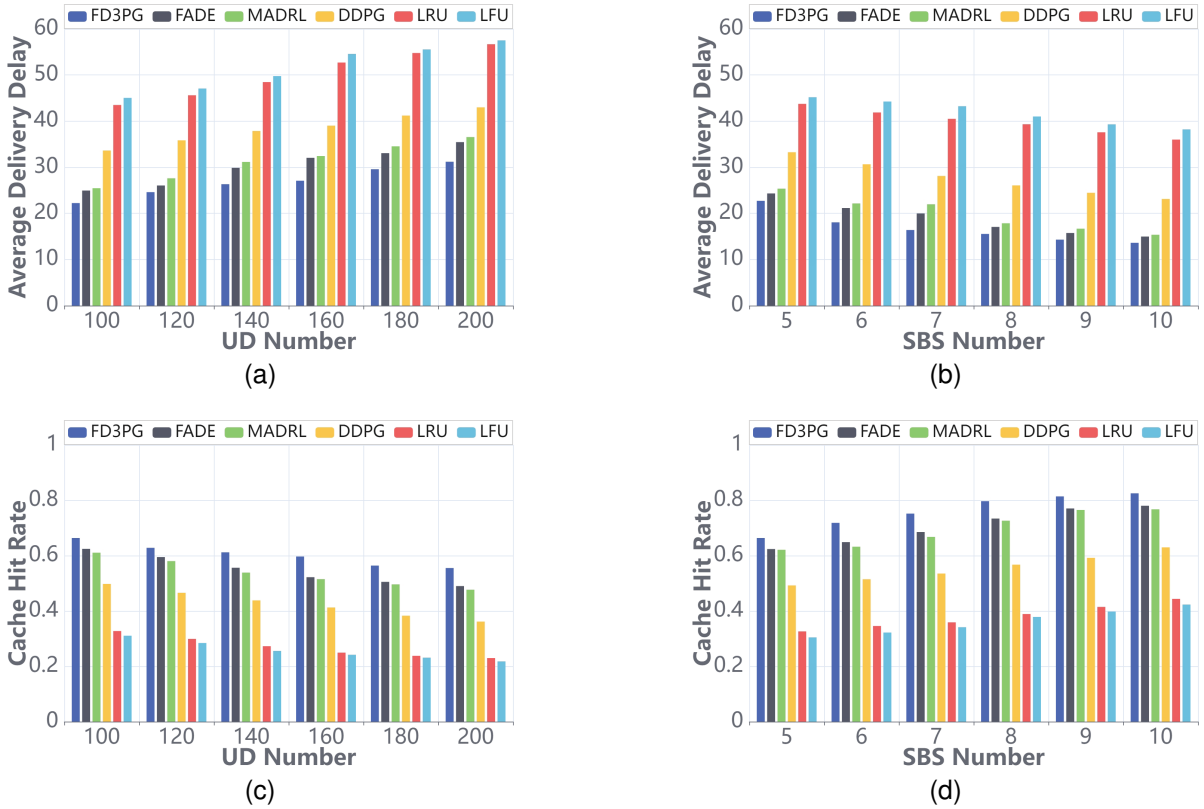

Fig. 6. (a) Average delivery delay and (c) cache hit rate versus different numbers of UD. (b) Average delivery delay and (d) cache hit rate versus different numbers of SBS.

cache hit rate of SBS and increases the probability of SBS requesting content from other SBSs or MBS. Consequently, this increases the complexity and frequency of SBS performing cache replacements to fulfill time-varying content requests. Therefore, each SBS needs to perform cache replacement more frequently to fulfill time-varying content requests. On the MLS dataset, the complexity of content requests further increases, leading to worse performance for all methods.

Furthermore, the cache hit rate of all methods decreases with an increasing content number. As depicted in Fig. Fig. 5(c), the proposed FD3PG achieves a cache hit rate of 57% when the number of contents reaches 400, while LFU and LRU perform the worst at 24% and 21%, respectively. Similarly, all methods perform worse on the MLS dataset than on the Synthetic dataset, as shown in 5(d). Specifically, due to their simple rule-based nature, LFU and LRU struggle to learn optimal policies based on different state information in dynamic and complex environments. In contrast, DRL-based methods use neural networks to make decisions, replacing simple rules. They not only leverage historical experience to learn optimal strategies but also adapt to new requests by continually interacting with the environment and adjusting their strategies based on feedback. Moreover, DRL-based methods prioritize obtaining higher rewards rather than merely satisfying content requests. As anticipated, FD3PG outperforms other methods across varying content quantities.

4) The impact of the number of UD and SBS: Fig. 6 illustrates the impact of the number of UD and the number of SBS on both the average delivery delay and cache hit rate. The range of the number of UD varies from 100 to 200, while the number of SBS increases from 5 to 10. Figs. 6(b) and 6(d) depict the impact of the number of SBS on the average delivery delay and cache hit rate, respectively. With the increase of the number of SBSs, the average delivery delay decreases across all methods. This is attributed to the collaborative caching among SBSs, which allows UDs to request more contents through SBSs as their number increases, resulting in higher cache hit rate and lower average delivery delay. It is worth noting that the impact of the number of SBS is more significant for the proposed FD3PG and MADRL compared to other methods. These two methods consider the interaction among agents, making the collaborative caching among SBSs more efficient. In contrast, DDPG, LRU, and LFU, while also reducing delivery delay and improving cache hit rate, often focus only on the impact of the action of agents on themselves, neglecting the actions of other agents. Therefore, the increase in the number of SBSs has less pronounced impact on them. It can be observed that the number of UD is positively correlated with average delivery delay and negatively correlated with cache hit ratio. Specifically, with a constant number of SBS, increasing the number of UD leads to greater diversity in content requests, which implies that UDs will request more contents that is not cached in SBSs.

5) The impact of the model selection: The impact of the EMD-based model selection is illustrated in Fig. 7. As shown in Fig. 7, when model selection is taken into account, the agent

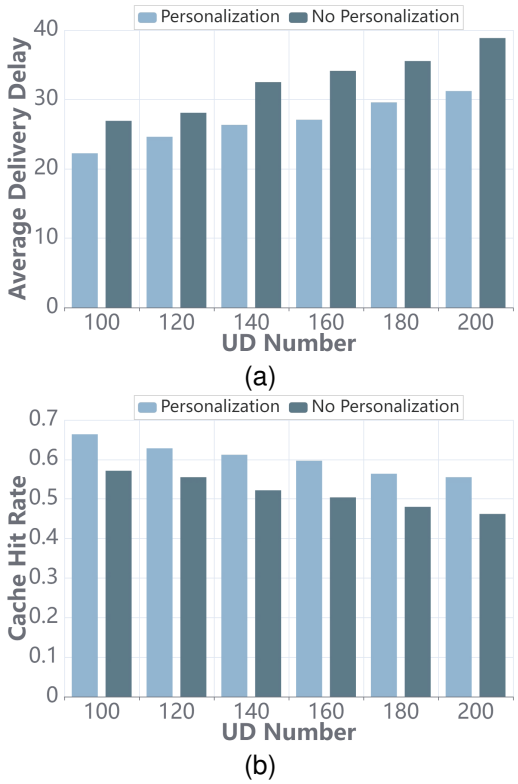

Fig. 7. (a) Average delivery delay and (b) cache hit rate versus different numbers of UD.

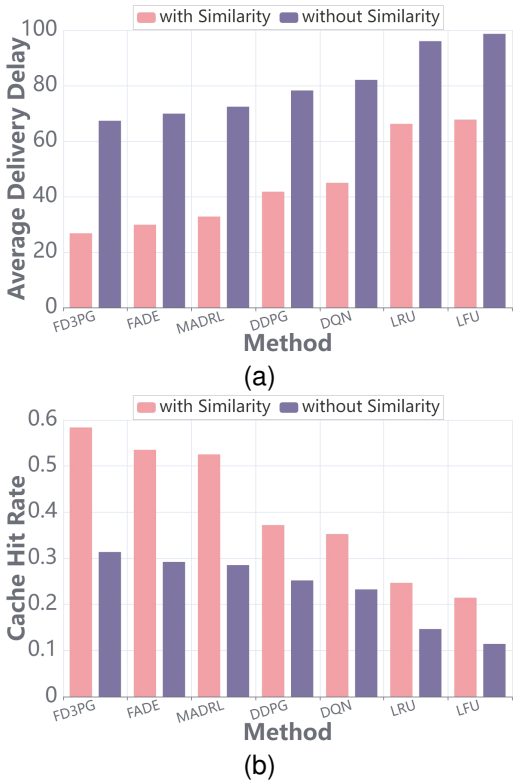

Fig. 8. (a) Average delivery delay and (b) Cache hit rate versus different methods.

performs better than when model selection is not taken into account. As noted in [60], models trained with the participation of diverse UDs exhibit a certain degree of personalization. In this study, we assume that there are significant differences in the average preferences of UDs served by different agents, which can lead to changes in the required optimal caching decision. If parameter aggregation is conducted among these agents, the personalization of their respective models may diminish. The decisions made by these models might not be optimally tailored to their specific environments, which results in a decrease in the QoS for the UDs they serve. Therefore, by employing EMD-based model selection, we aim to maintain a high degree of personalization and ensure that each agent can make decisions that are most suitable for their unique operational contexts, ultimately enhancing the overall experience of UDs.

6) The impact of the content recommendation: Finally, we present a comparison of different methods on the MLS dataset with and without content recommendation, as depicted in Fig. 8. From Fig. 8(a), it is obvious that all methods utilizing content recommendation achieve lower delivery delay. Fig. 8(b) illustrates the comparison of cache hit rate between methods with and without content recommendation. Traditional edge caching typically employs the direct hit method, satisfying content requests but leading to a higher probability of SBSs requesting content from other BSs, thereby increasing visits and content delivery delay. Additionally, not all cached contents in SBS are popular, resulting in inefficient use of cache space resources. CRS addresses these issues by analyzing content similarities and recommending similar content to UDs. This approach enhances cache resource utilization, mitigating the need for SBS to retrieve content from other BSs due to cache limitations. Furthermore, we consider collaboration between SBSs to further enhance edge caching resource utilization. Overall, methods incorporating content recommendation consistently achieve lower delivery delay and higher cache hit rate. This indicates that the integration of content recommendation and edge caching significantly improves the QoE for UDs and the QoS for SBSs.

In summary, it can be concluded that the proposed FD3PG can not only reduce content delivery delay and achieve significantly higher cache hit rate but also exhibit efficiency in more complex scenarios. Therefore, we demonstrate the effectiveness of the proposed method across various scenarios.

VII. Conclusion

This paper has investigated a content recommendation-based edge caching method in which each SBS recommends similar content to UDs instead of requesting content from the MBS or other SBSs when the requested content is not available in the cache space. First, we considered a multi-tier edge caching-enabled CDN architecture, considering the heterogeneity of content requests in different areas and cooperative caching among SBSs. Second, we formulated the optimization problem with the goal of minimizing long-term content delivery delay, approximated the optimization process as a POMDP, and proposed a DDPG-based method to solve it. Finally, we extended the POMDP to the case of multi-agent systems and proposed the FD3PG to address the corresponding

problem in the multi-agent case. Extensive simulations are conducted to rigorously evaluate the performance of FD3PG, and the results demonstrate its effectiveness across various scenarios compared with other existing methods.

VIII. Acknowledgments

This paper is supported in part by the National Natural Science Foundation of China (No. 61960206008), National Science Fund for Distinguished Young Scholars (No. 62025205), NSFC under grant 62172255, and Outstanding Youth Program of Hubei Natural Science Foundation under grant 2022CFA080. The corresponding author is Hao Wang.

References

[1] L. Li, G. Zhao and R. S. Blum, "A Survey of Caching Techniques in Cellular Networks: Research Issues and Challenges in Content Placement and Delivery Strategies," *IEEE Commun. Surv. Tutorials*, vol. 20, no. 3, pp. 1710-1732, 2018.

[2] T. Taleb, A. Ksentini, M. Chen *et al.*, "Coping With Emerging Mobile Social Media Applications Through Dynamic Service Function Chaining," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2859-2871, 2016.

[3] L. Pu, X. Chen, G. Mao, Q. Xie *et al.*, "Chimera: An Energy-Efficient and Deadline-Aware Hybrid Edge Computing Framework for Vehicular Crowdsensing Applications," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 84-99, 2019.

[4] H. Zhou, Z. Zhang, Y. Wu *et al.*, "Energy Efficient Joint Computation Offloading and Service Caching for Mobile Edge Computing: A Deep Reinforcement Learning Approach," *IEEE Trans. Green Commun. Networking*, vol. 7, no. 2, pp. 950-961, 2023.

[5] J. Sung, M. Kim, K. Lim *et al.*, "Efficient Cache Placement Strategy in Two-Tier Wireless Content Delivery Network," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 1163-1174, 2016.

[6] X. Wang, Y. Han, V. C. M. Leung *et al.*, "Convergence of Edge Computing and Deep Learning: A Comprehensive Survey," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 869-904, 2020.

[7] C. Papagianni, A. Leivadeas and S. Papavassiliou, "A Cloud-Oriented Content Delivery Network Paradigm: Modeling and Assessment," *IEEE Trans. Dependable Secure Comput.*, vol. 10, no. 5, pp. 287-300, 2013.

[8] T. Taleb, P. A. Frangoudis, I. Benkacem "CDN Slicing over a Multi-Domain Edge Cloud," *IEEE Trans. Mob. Comput.*, vol. 19, no. 9, pp. 2010-2027, 2020.

[9] H. Hu, Y. Wen, T. -S. Chua, "Joint Content Replication and Request Routing for Social Video Distribution Over Cloud CDN: A Community Clustering Method," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 7, pp. 1320-1333, 2016.

[10] H. Zhou, M. Li, N. Wang *et al.*, "Accelerating Deep Learning Inference via Model Parallelism and Partial Computation Offloading," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 2, pp. 475-488, 2023.

[11] H. Zhou, Z. Wang, H. Zheng *et al.*, "Cost Minimization-Oriented Computation Offloading and Service Caching in Mobile Cloud-Edge Computing: An A3C-based Approach," *IEEE Trans. Network Sci. Eng.*, vol. 10, no. 3, pp. 1326-1338, 2023.

[12] H. Zhou, K. Jiang, X. Liu *et al.*, "Deep Reinforcement Learning for Energy-Efficient Computation Offloading in Mobile-Edge Computing," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1517-1530, 2022.

[13] B. Jedari, G. Premsankar, G. Illahi *et al.*, "Video Caching, Analytics, and Delivery at the Wireless Edge: A Survey and Future Directions," *IEEE Commun. Surv. Tutorials*, vol. 23, no. 1, pp. 431-471, 2021.

[14] Q. Yuan, X. Wei, X. Xiong *et al.*, "A Hybrid Neural Collaborative Filtering Model for Drug Repositioning," *Proc. IEEE BIBM*, 2020, pp. 515-518.

[15] B. Sun, H. Tan, D. Yang *et al.*, "The System of Personalized Learning Resource Recommendation and Experimental Teaching Based on Collaborative Filtering," *Proc. IEEE ISCAS*, 2022, pp. 862-866.

[16] H. Feng, S. Guo, C. Chen *et al.*, "Distributed Caching Control Strategy in Mobile Edge Computing: A Mean Field Game Approach," *Proc. IEEE SECON*, 2022, pp. 443-451.

[17] S. Liu, C. Zheng, Y. Huang *et al.*, "Distributed Reinforcement Learning for Privacy-Preserving Dynamic Edge Caching," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 749-760, 2022.

[18] L. Chhangte, N. Karamchandani, D. Manjunath *et al.*, "Towards a Distributed Caching Service at the WiFi Edge Using Wi-Cache," *IEEE Trans. Netw. Serv. Manage.*, vol. 18, no. 4, pp. 4489-4502, 2021.

[19] F. Lyu, J. Ren, N. Cheng *et al.*, "LeaD: Large-Scale Edge Cache Deployment Based on Spatio-Temporal WiFi Traffic Statistics," *IEEE Trans. Mob. Comput.*, vol. 20, no. 8, pp. 2607-2623, 2021.

[20] A. Tian, B. Feng, H. Zhou *et al.*, "Efficient Federated DRL-Based Cooperative Caching for Mobile Edge Networks," *IEEE Trans. Netw. Serv. Manage.*, vol. 20, no. 1, pp. 246-260, 2023.

[21] C. Li, Y. Zhang and Y. Luo, "A Federated Learning-Based Edge Caching Approach for Mobile Edge Computing-Enabled Intelligent Connected Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 3, pp. 3360-3369, 2023.

[22] D. Qiao, S. Guo, D. Liu *et al.*, "Adaptive Federated Deep Reinforcement Learning for Proactive Content Caching in Edge Computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4767-4782, 2022.

[23] Y. M. Saputra, D. T. Hoang, D. N. Nguyen *et al.*, "A Novel Mobile Edge Network Architecture with Joint Caching-Delivering and Horizontal Cooperation," *IEEE Trans. Mob. Comput.*, vol. 20, no. 1, pp. 19-31, 2021.

[24] M. Kharbutli and R. Sheikh, "LACS: A Locality-Aware Cost-Sensitive Cache Replacement Algorithm," *IEEE Trans. Comput.*, vol. 63, no. 8, pp. 1975-1987, 2014.

[25] Z. Zhou, W. Wang, M. Guo *et al.*, "A Design Space for Surfacing Content Recommendations in Visual Analytic Platforms," *IEEE Trans. Visual Comput. Graphics*, vol. 29, no. 1, pp. 84-94, 2023.

[26] H. Xu, C. Li, Y. Zhang, L. Duan *et al.*, "MetaCAR: Cross-Domain Meta-Augmentation for Content-Aware Recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 8199-8212, 2023.

[27] Z. Liu, D. Zhou, H. Liu *et al.*, "Graph-Grounded Goal Planning for Conversational Recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4923-4939, 2023.

[28] M. Sheng, W. Teng, X. Chu *et al.*, "Cooperative Content Replacement and Recommendation in Small Cell Networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 2049-2063, 2021.

[29] C. Sun, X. Li, J. Wen *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.

[30] M. Song, H. Shan, Y. Fu *et al.*, "Joint User-Side Recommendation and D2D-Assisted Offloading for Cache-Enabled Cellular Networks with Mobility Consideration," *IEEE Trans. Wireless Commun.*, vol. 22, no. 11, pp. 2023.

[31] D. Yu, T. Wu, C. Liu *et al.*, "Joint Content Caching and Recommendation in Opportunistic Mobile Networks Through Deep Reinforcement Learning and Broad Learning," *IEEE Trans. Serv. Comput.*, vol. 16, no. 4, pp. 2727-2741, 2023.

[32] Z. Li, X. Gao, Q. Li *et al.*, "Edge Caching Enhancement for Industrial Internet: A Recommendation-Aided Approach," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 16941-16952, 2022.

[33] Y. Fu, Q. Yu, A. K. Y. Wong *et al.*, "Exploiting Coding and Recommendation to Improve Cache Efficiency of Reliability-Aware Wireless Edge Caching Networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7243-7256, 2021.

[34] A. Ndikumana, N. H. Tran, D. H. Kim *et al.*, "Deep Learning Based Caching for Self-Driving Cars in Multi-Access Edge Computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 5, pp. 2862-2877, 2021.

[35] A. Lekharu, M. Jain, A. Sur *et al.*, "Deep Learning Model for Content Aware Caching at MEC Servers," *IEEE Trans. Netw. Serv. Manage.*, vol. 19, no. 2, pp. 1413-1425, 2022.

[36] Z. Yu, J. Hu, G. Min *et al.*, "Mobility-Aware Proactive Edge Caching for Connected Vehicles Using Federated Learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5341-5351, 2021.

[37] E. M. Bakr, H. Ben-Ammar, H. M. Eraqi *et al.*, "End-to-End Deep Learning Proactive Content Caching Framework," *Proc. IEEE GLOBECOM*, 2022, pp. 1043-1048.

[38] H. Pang, J. Liu, X. Fan *et al.*, "Toward Smart and Cooperative Edge Caching for 5G Networks: A Deep Learning Based Approach," *Proc. IEEE/ACM IWQoS*, 2018, pp. 1-6.

[39] M. Zhao and M. R. Nakhai, "A Unified Federated Deep Q Learning Caching Scheme for Scalable Collaborative Edge Networks," *IEEE Trans. Mob. Comput.*, doi: 10.1109/TMC.2024.3382824.

[40] X. Li, J. Liu, X. Chen *et al.*, "Caching on the Sky: A Multi-Agent Federated Reinforcement Learning Approach for UAV-Assisted Edge Caching," *IEEE Internet Things J.*, doi: 10.1109/JIOT.2024.3401219.

[41] H. Wu, B. Wang, H. Ma, *et al.*, "Multi-Agent Federated Deep Reinforcement Learning Based Collaborative Caching Strategy for Vehicular Edge Networks," *IEEE Internet Things J.*, doi: 10.1109/JIOT.2024.3392329.

[42] K. Jiang, Y. Cao, Y. Song *et al.*, "Asynchronous Federated and Reinforcement Learning for Mobility-Aware Edge Caching in IoV," *IEEE Internet Things J.*, vol. 11, no. 9, pp. 15334-15347, 2024.

[43] M. Lei, Q. Li, X. Ge *et al.*, "Partially Collaborative Edge Caching Based on Federated Deep Reinforcement Learning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 1, pp. 1389-1394, 2023.

[44] S. He, Y. Lu, Q. Tang *et al.*, "Edge Caching and Computation Management for Real-Time Internet of Vehicles: An Online and Distributed Approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2183-2197, 2021.

[45] X. Li, X. Wang, P. -J. Wan *et al.*, "Hierarchical Edge Caching in Device-to-Device Aided Mobile Networks: Modeling, Optimization, and Design," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1768-1785, 2018.

[46] D. Yu, T. Wu, C. Liu *et al.*, "Joint Content Caching and Recommendation in Opportunistic Mobile Networks through Deep Reinforcement Learning and Broad Learning," *IEEE Trans. Serv. Comput.*, vol. 16, no. 4, pp. 2727-2741, 2023.

[47] H. Zhou, K. Jiang, G. Min *et al.*, "Distributed Multi-Agent Reinforcement Learning for Cooperative Edge Caching in Internet of Vehicles," *IEEE Trans. Wireless Commun.*, vol. 22, no. 4, pp. 9595-9609, 2023.

[48] S. Baek, H. G. Lee, C. Nicopoulos *et al.*, "Size-Aware Cache Management for Compressed Cache Architectures," *IEEE Trans. Comput.*, vol. 64, no. 8, pp. 2337-2352, 2015.

[49] T. Xie, N. Nambiar, T. He *et al.*, "Attack Resilience of Cache Replacement Policies: A Study Based on TTL Approximation," *IEEE/ACM Trans. Networking*, vol. 30, no. 6, pp. 2433-2447, 2022.

[50] Z. Zhu, S. Wan, P. Fan *et al.*, "Federated Multiagent Actor-Critic Learning for Age Sensitive Mobile-Edge Computing," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1053-1067, 2022.

[51] H. Zhou, M. Li, P. Sun *et al.*, "Accelerating Federated Learning via Parameter Selection and Pre-synchronization in Mobile Edge-Cloud Networks," *IEEE Trans. Mob. Comput.*, vol. pp, no. 99, pp. 1-1, 2024.

[52] D. Yang, X. He, J. Wang *et al.*, "Federated Causality Learning with Explainable Adaptive Optimization," *Proc. AAAI*, Vol. 38, No. 15, pp. 16308-16315, 2024.

[53] Y. Zhang, D. Liu, M. Duan *et al.*, "FedMDS: An Efficient Model Discrepancy-Aware Semi-Asynchronous Clustered Federated Learning Framework," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 3, pp. 1007-1019, 2023.

[54] J. Huang, R. Zhang, R. Buyya *et al.*, "Heads-Join: Efficient Earth Mover's Distance Similarity Joins on Hadoop," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 6, pp. 1660-1673, 2016.

[55] S. He, Y. Lu, Q. Tang *et al.*, "Blockchain-Based P2P Content Delivery With Monetary Incentivization and Fairness Guarantee," *IEEE Trans. Parallel Distrib. Syst.*, vol. 34, no. 2, pp. 746-765, 2023.

[56] X. Cao, N. Chen, X. Yuan *et al.*, "A Cooperative Edge Caching Approach Based on Multi-Agent Deep Reinforcement Learning," *Proc. CSCWD*, 2023, pp. 1772-1777.

[57] G. Quan, J. Tan and A. Eryilmaz, "Counterintuitive Characteristics of Optimal Distributed LRU Caching Over Unreliable Channels," *IEEE/ACM Trans. Networking*, vol. 28, no. 6, pp. 2461-2474, 2020.

[58] X. Wang, C. Wang, X. Li *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.

[59] U. Akhtar and S. Lee, "Adaptive Cache Replacement in Efficiently Querying Semantic Big Data," *Proc. IEEE ICWS*, 2018, pp. 367-370.

[60] Long, G., Xie, M., Shen, T. *et al.*, "Multi-center federated learning: clients clustering for better personalization," *World Wide Web*, vol. 26, pp. 481-500, 2023.

Huan Zhou (Senior Member, IEEE) received the Ph.D. degree from the Department of Control Science and Engineering, Zhejiang University. He was a visiting scholar with the Temple University from November 2012 to May, 2013, and a CSC supported postdoc fellow with the University of British Columbia from November 2016 to November 2017. Currently, he is a full professor with the College of Computer and Information Technology, China Three Gorges University. He was a lead guest editor of the *Pervasive and Mobile Computing*, and Special Session chair of the 3rd International Conference on Internet of Vehicles (IoV 2016), and TPC member of IEEE WCSP'13'14, CCNC'14'15, ICNC'14'15, ANT'15'16, IEEE Globecom'17'18, ICC'18'19, etc. He has published more than 50 research papers in some international journals and conferences, including the *IEEE Journal on Selected Areas in Communications*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Vehicular Technology* and so on. His research interests include mobile social networks, vehicular ad hoc networks, opportunistic mobile networks, and mobile data offloading. He received the Best Paper Award of I-SPAN 2014 and I-SPAN 2018, and is currently serving as an associate editor of the *IEEE Access* and *EURASIP Journal on Wireless Communications and Networking*.

Hao Wang received the B.E. degree from Hubei Normal University, Huangshi, China, in 2021, and the M.S. degree from China Three Gorges University, Yichang, China, in 2024. Currently, he is pursuing the Ph.D. degree from Beijing University of Posts and Telecommunications. His main research interests include edge computing, computation offloading, content caching, and federated learning.

Zhiwen Yu (Senior Member, IEEE) received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2005. He is currently a professor with the School of Computer Science, Northwestern Polytechnical University. He was an Alexander Von Humboldt fellow with Mannheim University, Germany, and a research fellow with Kyoto University, Kyoto, Japan. His research interests include ubiquitous computing, HCI, and mobile sensing and computing.

Bin Guo (Senior Member, IEEE) received the Ph.D. degree in computer science from Keio University, Japan, in 2009, and then was a postdoc researcher with Institut Telecom SudParis, France. He is a professor with Northwestern Polytechnical University, China. His research interests include ubiquitous computing, mobile crowd sensing, and HCI. He has served as an associate editor of IEEE Communications Magazine and IEEE Transactions on Human-Machine-Systems, the guest editor of ACM Transactions on Intelligent Systems and Technology and IEEE Internet of Things Journal, the general co-chair of IEEE UIC'15, and the program chair of IEEE CPSCom'16, ANT'14, and UIC'13.

Mingjun Xiao (Senior Member, IEEE) received the Ph.D. degree from USTC, in 2004. He is a professor with the School of Computer Science and Technology, University of Science and Technology of China (USTC). His research interests include crowdsourcing, mobile social networks, mobile cloud computing, blockchain, data security and privacy. He has published more than 80 papers in referred journals and conferences, including *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Computers*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Services Computing*, INFOCOM, ICNP, etc. He served as the TPC member of INFOCOM'20, DASFAA'20, INFOCOM'19, ICDCS'19, DASFAA'19, etc. He is on the reviewer board of several top journals such as *IEEE Transactions on Mobile Computing*, *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Vehicular Technology*, *IEEE Transactions on Cloud Computing*, etc.

Jie Wu (Fellow, IEEE) is the director of the Center for Networked Computing and Laura H. Carnell professor with Temple University. He also serves as the director of International Affairs at College of Science and Technology. He served as chair of Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and associate vice provost for International Affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director with the National Science Foundation and was a distinguished professor with Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including the *IEEE Transactions on Services Computing* and the *Journal of Parallel and Distributed Computing*. He was general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, IEEE ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society distinguished visitor, ACM distinguished speaker, and chair for IEEE Technical Committee on Distributed Processing (TCDP). He is a CCF distinguished speaker. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.