# Resilient Collaborative Caching for Multi-edge Systems with Robust Federated Deep Learning

Zheyi Chen, *Member, IEEE*, Jie Liang, Zhengxin Yu, *Member, IEEE*, Hongju Cheng, *Senior Member, IEEE*, Geyong Min, *Member, IEEE*, and Jie Li, *Fellow, IEEE*

***Abstract*—As a key technique for future networks, the performance of emerging multi-edge caching is often limited by inefficient collaboration among edge nodes and improper resource configuration. Meanwhile, achieving optimal cache hit rates poses substantive challenges without effectively capturing the potential relations between discrete user features and diverse content libraries. These challenges become further sophisticated when caching schemes are exposed to adversarial attacks that seriously impair cache performance. To address these challenges, we introduce *RoCoCache*, a resilient collaborative caching framework that uniquely integrates robust federated deep learning with proactive caching strategies, enhancing performance under adversarial conditions. First, we design a novel partitioning mechanism for multi-dimensional cache space, enabling precise content recommendations in user classification intervals. Next, we develop a new Discrete-Categorical Variational Auto-Encoder (DC-VAE) to accurately predict content popularity by overcoming posterior collapse. Finally, we create an original training mode and proactive cache replacement strategy based on robust federated deep learning. Notably, the residual-based detection for adversarial model updates and similarity-based federated aggregation are integrated to avoid the model destruction caused by adversarial updates, which enables the proactive cache replacement adapting to optimized cache resources and thus enhances cache performance. Using the real-world testbed and datasets, extensive experiments verify that the *RoCoCache* achieves higher cache hit rates and efficiency than state-of-the-art methods while ensuring better robustness. Moreover, we validate the effectiveness of the components designed in *RoCoCache* for improving cache performance via ablation studies.**

***Index Terms*—multi-edge collaborative caching, robust federated deep learning, cache space partitioning, content popularity prediction, proactive cache replacement.**

## I. Introduction

With the significant development of 5G techniques, increasingly sophisticated applications are being deployed across industrial manufacturing, digital economy, vehicle networking, and smart cities [1]. In the case of cloud computing, the tasks and data generated by the applications are uploaded to the remote cloud for processing, causing serious network congestion and service delay [2]. To alleviate this problem, Mobile Edge Computing (MEC), as an essential technique for future networks, deploys computing and storage resources at the network edge in closer proximity to end devices [3], provisioning more reliable real-time computing and data storage for end-device applications. Thus, edge nodes can perform various management operations such as signal processing, distributed caching, and wireless resource collaboration. Among these operations, distributed caching enables caching user-centric content within edge nodes, which reduces access delay and data duplication storage, thus enhancing user experience and cutting system costs. Whereas, cache performance is commonly limited by the size of edge cache space and overheads. Therefore, how to efficiently utilize the cache space and improve cache performance has garnered extensive attention from both academia and industry [4]. Generally, the cache performance is constrained by many factors such as cache size, content relevance, cache partitioning, and cache replacement. Previous studies commonly considered proactive content caching based on user preferences in constrained cache space [5], [6]. To optimize the configuration of cache resources, the cache space needs to be further partitioned according to user features, activities, and request patterns, enabling more precise content recommendations for similar users across multiple edges. Therefore, exploring the potential relationships between users and content characteristics in multi-dimensional space will enhance the hit rate of user-accessed resources, but it remains an open challenge.

Multi-edge collaborative caching has emerged as a feasible manner that optimizes cache resource configuration and reduces service delay [7]. Users can determine their requested contents from other edge nodes that perform collaborative caching if their connected edge nodes do not match their requests. Most of the existing studies [5], [6], [8]–[11] employed Deep Reinforcement Learning (DRL) or game theory. However, with the increasing number of end devices and heightened sensitivity of user privacy, the existing solutions cannot effectively handle the security issues and cache resource configuration. As a distributed training framework, Federated Learning (FL) is regarded as a promising solution to handle the above issues [12]. Following its core principle, edge nodes collaborate to train a global model by uploading model parameters without revealing raw data [13]–[17]. FL tends to

This work was partly supported by the National Natural Science Foundation of China under Grant No. 62202103, the Central Funds Guiding the Local Science and Technology Development under Grant No. 2022L3004, the Fujian Province Technology and Economy Integration Service Platform under Grant No. 2023XRH001, and the Fuzhou-Xiamen-Quanzhou National Independent Innovation Demonstration Zone Collaborative Innovation Platform under Grant No. 2022FX5. (Corresponding authors: Zhengxin Yu and Geyong Min)

Zheyi Chen, Jie Liang, and Hongju Cheng are with the College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China, the Engineering Research Center of Big Data Intelligence, Ministry of Education, Fuzhou 350002, China, and also with the Fujian Key Laboratory of Network Computing and Intelligent Information Processing (Fuzhou University), Fuzhou 350116, China. (e-mail: z.chen@fzu.edu.cn, steveliangjie@163.com, cscheng@fzu.edu.cn)

Zhengxin Yu is with the School of Computing and Communications, Lancaster University, Lancaster LA1 4YW, United Kingdom. (e-mail: z.yu8@lancaster.ac.uk)

Geyong Min is with the Department of Computer Science, Faculty of Environment, Science and Economy, University of Exeter, Exeter EX4 4QF, United Kingdom. (e-mail: g.min@exeter.ac.uk)

Jie Li is with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China. (e-mail: li-jiecs@sjtu.edu.cn)

meet the heterogeneous user demands across multiple edges by the well-trained global model while ensuring privacy security and the scalability of large-scale networks [18]. However, in complex MEC environments, unintentional model corruption or adversarial model interference with malicious intention may cause the inability to perform model training and the degraded quality of model updates [19]. For example, unintentional model corruption may occur due to noisy training labels, insufficient data samples, and uploaded models with low quality. Moreover, malicious edge nodes may deliberately launch adversarial attacks to tamper models such as Byzantine [20] and Backdoor attacks [21]. However, the detection of adversarial attacks is commonly computation-intensive since it requires substantial data collection and analysis to identify misbehaviors [22]. These issues seriously impact cache performance and increase system overheads. Specifically, the key challenges when dealing with the problem of multi-edge collaborative caching are summarized below.

- ***Discrete user features and diverse content requests.*** Different users may have various content preferences due to their discrete features. Therefore, it is challenging to find the potential relationships between the discrete user feature distribution and diverse content requests.
- ***Model scalability.*** With an increasing number of end devices, there will be more data that exhibit discrete distributions, resulting in higher computation and communication overheads for the caching scenario with a single edge node. However, classic centralized training frameworks encounter limited model scalability.
- ***Model robustness.*** Edge nodes might unintentionally upload low-quality models or be subjected to adversarial attacks by malicious nodes, leading to seriously degraded robustness during model updates.

To address these important challenges, we propose *RoCoCache*, a novel resilient collaborative caching framework for multi-edge systems with robust federated deep learning. The main contributions of this paper are summarized as follows.

- We propose a new partitioning mechanism for multi-dimensional edge cache space, consisting of multi-dimensional user partitioning and cache space partitioning. This mechanism considers user features, activities, and the divergence of memory access interval to perceptually optimize cache resources, promising that users receive accurate content recommendations in their classification interval.
- We investigate the potential posterior collapse in the classic Variational Auto-Encoder (VAE). To this end, we develop a Discrete-Categorical Variational Auto-Encoder (DC-VAE) for more precise adaptation to discrete distributions in the user request matrix. Specifically, the DC-VAE first learns the implicit category space consisting of discrete vectors. Next, it employs the nearest neighbor to find discrete implicit vectors, assisting the decoder in generating the user request matrix. Thus, we solve the posterior collapsing issue and enhance the prediction accuracy of content popularity.
- We design a novel training mode based on robust federated deep learning (RFDL). By training with the user request data stored on each edge node, local models are aggregated to generate a globally-shared model. Specifically, a residual-based detection method is proposed to accurately capture adversarial model updates. Meanwhile, a similarity-based FL aggregation method is designed to avoid the destruction of the globally-shared model caused by adversarial updates. Notably, we theoretically verify the effectiveness of both methods in detail.
- Using the real-world testbed and datasets, we conduct extensive experiments to validate the superiority of the proposed *RoCoCache*. The results show that the *RoCoCache* achieves higher cache hit rates and improved efficiency than state-of-the-art methods while guaranteeing better robustness. Moreover, the effectiveness of the components designed in *RoCoCache* is also demonstrated via ablation studies.

The rest of this paper is organized as follows. Section II reviews the related work. Section III describes the system model and formulates the optimization problem. Section IV presents the proposed *RoCoCache* in detail. Section V evaluates the *RoCoCache*. Finally, Section VI concludes this paper.

## II. Related Work

Classic caching strategies, such as First In First Out (FIFO), Least Frequently Used (LFU), and Least Recently Used (LRU) [23], follow static rules to update cached contents. However, they are not adapted to complex multi-edge environments since they cannot analyze changing content popularity. This issue has been significantly improved by recent advancements in learning-based methods [24]. In multi-edge caching, several key factors should be also considered including user features, user privacy, model scalability, and model robustness [25]. This section analyzes the related studies from the perspectives of collaborative caching and FL-based edge caching.

### A. Collaborative Caching

Zhou $et\ al.$ [5] developed a Multi-Agent Reinforcement Learning (MARL) enabled cooperative caching strategy, which formulated the joint cooperative caching and recommendation problem as a Multi-Agent Multi-Armed Bandit (MAMAB) problem with the aim of minimizing the average download latency. Lin $et\ al.$ [6] proposed a preference-aware content caching and migration scheme for video content delivery, with the consideration of users' request preferences and long-term content migration costs. Zong $et\ al.$ [8] designed an ensemble learning-based edge caching strategy, which equipped classic LFU/LRU with LSTM-based time-series analysis, aiming to optimize the cache configurations for heterogeneous edge nodes. Chen $et\ al.$ [9] formulated the collaborative caching as a Partially Observable Markov Decision Process (POMDP) and designed a multi-agent reinforcement learning based method to optimize the overheads of multi-edge cache systems. Hui $et\ al.$ [10] proposed a collaborative caching scheme based on a double-auction game, which motivated cellular base stations to cooperate with roadside units for multicast-assisted content delivery. Lin $et\ al.$ [11] investigated

a collaborative service caching scheme among multiple Service Providers (SPs) by using game theory, where SPs were modeled as profit-oriented players that formed coalitions to share edge resources and costs. Zhang *et al.* [26] designed a two-step approach, which consisted of a content popularity prediction model with temporal convolution networks and a dynamic programming algorithm to address the problem of cooperative content caching. Jin *et al.* [27] developed a cooperative caching strategy with optimization theory, aiming to deal with the dynamic changes in vehicular edge networks and minimize average transmission delay. Peng *et al.* [28] designed a cooperative caching system that adopted Automatic Content Congregating (ACC) to improve cache hit rate and Mutual Assistance Group (MAG) to balance workloads by guiding requests, prefetching content, and redistributing requests among edge nodes.

In general, the existing studies on collaborative caching have exhibited good performance in simple and static caching scenarios. However, they cannot effectively manage discrete user features and diverse content requests. This inadequacy undermines their ability to provide precise content recommendations based on user features, activities, and request patterns, seriously affecting cache hit rate and increasing system overheads. Furthermore, due to inefficient multi-edge collaboration and irrational cache resource configuration, they struggled to cope with the complex multi-edge caching scenarios characterized by the dramatically increasing number of end devices and high privacy sensitivity. Therefore, it is challenging for them to learn the optimal caching strategy and achieve high efficiency of collaborative caching.

### B. FL-based Edge Caching

Qiao *et al.* [13] optimized edge caching based on Federated Reinforcement Learning (FRL) that can adjust FL participants automatically, solving the problems of high resource overheads and low training efficiency. Krishnendu *et al.* [14] proposed an FL-based caching strategy that considered the decision span and neighbor correlation, providing high-probability generalization guarantees for improving cache performance. Yu *et al.* [15] designed an FL-based hierarchical cooperative caching method, which employed VAE to capture different content popularity and then combined horizontal and vertical cooperation to implement content caching with different content popularity. Li *et al.* [16] developed an FL-based caching scheme among heterogeneous edge nodes, where an information aggregation protocol was designed to reduce the disparity in content popularity between local user feedback and the central server. Wu *et al.* [17] proposed a cooperative caching scheme in vehicular edge networks based on asynchronous federated deep reinforcement learning, aiming to predict popular contents and then obtain the proper cooperative caching locations for the predicted popular contents. However, in complex edge environments, security concerns exacerbate the challenges of maintaining high cache performance. The classic FL-based caching strategies are vulnerable to unintended model corruption or intentional adversarial model interference [20], [21], resulting in the degraded quality of the global model. This degradation adversely impacts content popularity prediction and cache performance.

Recently, there have been some studies on the security and robustness of edge caching. Cui *et al.* [29] proposed a blockchain-based FL edge cache system with model compression, which adopted smart contracts for decentralized entities to record and verify the transactions, improving cache performance while ensuring system security. Feng *et al.* [30] designed a proactive caching method with FL-based prediction, incorporating a privacy-preserving framework with local differential privacy and an attention mechanism for adversarial attack detection. Xie *et al.* [31] proposed a cache replacement scheme that can perceived and adapted to the attack levels, which evaluated a suite of representative replacement policies within the Time To Live (TTL) approximation framework to assess their effectiveness in preserving hit ratios for legitimate users. Manzoor *et al.* [32] proposed a robust FL-based content caching approach that utilized a generative adversarial network to differentiate noisy and actual federated weights, thereby mitigating the interference from adversarial model updates.

Generally, the existing studies on FL-based edge caching have proved its promising capability in enhancing privacy protection, model scalability, and robustness. They also performed well in model training speed and solving the issue of data island. The blockchain-based solutions are not lightweight, and they might cause excessive overheads. The other solutions may fail to accurately identify and mitigate malicious attacks, which cannot promise good model security and cache performance. There remains an ongoing challenge in optimizing cache performance while considering both model scalability and model robustness.

## III. System Model and Problem Formulation

The proposed multi-edge collaborative caching system, depicted in Fig. 1, consists of $M$ edge nodes, each containing a MEC server and a base station, denoted by the set $E = \{e_1, e_2, ..., e_m, ..., e_M\}$, and $N$ users, denoted by the set $U = \{u_1, u_2, ..., u_n, ..., u_N\}$. The caching space of edge nodes is defined as the set $C = \{C_1, C_2, ..., C_m, ..., C_M\}$. Each user is connected to an edge node and communicates through the wireless link provided by the associated base station. Furthermore, the communications among edge nodes, as well as between edge nodes and the cloud data center are conducted via the backhaul link. The caching space status of each edge node is periodically updated according to the proactive caching replacement strategy and then broadcast to other edge nodes within the proposed system. The content library of the cloud data center is denoted as $F = \{f_1, f_2, ..., f_i, ..., f_I\}$, where $I$ indicates the number of accessible contents. Moreover, users are discretely distributed in the service zone of each edge node. When the user $u_n$ sends a request for the content $f_i$ to its connected edge node, the workflow is given as follows.

**Step 1:** The current edge node checks whether it has cached $f_i$. If $f_i$ is cached, the edge node will send it to $u_n$ directly. Otherwise, it goes to **Step 2**.

**Step 2:** The current edge node searches for whether there exists a collaborative edge node that caches $f_i$. If there exists,

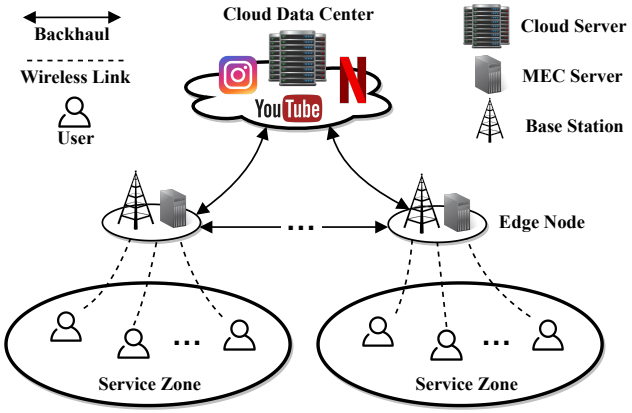



Fig. 1. The proposed multi-edge collaborative caching system.

the collaborative edge node will forward $f_i$ to the current edge node via the backhaul link, and then $f_i$ will be sent to $u_n$. Otherwise, it goes to **Step 3**.

**Step 3:** If no collaborative edge node caches $f_i$, the content library in the cloud data center will provide $f_i$ and forward it to the current edge node through the backhaul link, and then $f_i$ will be sent to $u_n$.

In the scenario of multi-edge collaborative caching, the content requests from users are dynamic, revealing spatio-temporal dependencies. Therefore, enhancing the cache hit rate heavily relies on precise content popularity prediction, followed by caching the user-interest contents into the cache space of edge nodes. Specifically, the popularity of $f_i$ on the edge node $e_m$ is defined as

$$P_{i,m} = \frac{req_{i,m}}{req_m}, \tag{1}$$

where $req_{i,m}$ is the number of requests for $f_i$ received by $e_m$, and $req_m$ is the total number of requests received by $e_m$.

The proposed *RoCoCache* (details are given in Section IV) enables precise prediction of content popularity. To evaluate the prediction accuracy, the global loss function is defined as

$$J(w^{(r)}) = \sum_{m=1}^{M} \frac{req_m}{req} MSE_m(w_m^{(r)}), \tag{2}$$

where $r$ indicates the FL communication round, $w^{(r)}$ is the parameter of the global prediction model, $req$ is the total number of requests received by all edge nodes, $w_m^{(r)}$ is the parameter of a local prediction model, and Mean-Square Error (MSE) is defined as

$$MSE_m(w_m^{(r)}) = \frac{1}{I} \sum_{i=1}^{I} \left(y_{i,m}(w_m^{(r)}) - P_{i,m}(r)\right)^2, \tag{3}$$

where $y_{i,m}(w_m^{(r)})$ is the predicted popularity value of $f_i$ on $e_m$ and $P_{i,m}(r)$ is the actual value.

Moreover, the cache hit rate is defined as

$$\mathrm{H} = \frac{\sum_{m=1}^{M} \sum_{i=1}^{N} \theta_m(f_i)}{\sum_{m=1}^{M} req_m} \times 100\%, \tag{4}$$

where $\theta_m(f_i)$ indicates whether $e_m$ caches the content requested by users or not, and it is defined as

$$\theta_m(f_i) = \begin{cases} 1, & \text{if } f_i \text{ is in } C_m \\ 0, & \text{otherwise} \end{cases}. \tag{5}$$

The cache performance is affected by various factors including cache resource configuration, content popularity, model robustness, and cache replacement strategy. By comprehensively considering these factors, the proposed *RoCoCache* is able to effectively improve the cache performance within a multi-edge collaborative caching system.

## IV. The Proposed RoCoCache

Based on the proposed system model and problem formulation, we propose *RoCoCache*, a novel resilient collaborative caching framework for multi-edge systems with RFDL. First, we optimize the cache space of edge nodes via a new multi-dimensional cache space partitioning and determine the proper size of cache space for interval users. Next, we design a new DC-VAE to learn the implicit category space comprising discrete vectors. In DC-VAE, the decoder employs the nearest neighbor to find discrete hidden vectors and then generates the calibrated user request matrix, thereby improving the accuracy of content popularity prediction. Then, we design a novel training mode based on RFDL to improve model scalability and robustness. This involves a residual-based detection method to capture adversarial model updates, coupled with a similarity-based FL aggregation method to avoid the damage of the globally-shared model caused by adversarial updating. Finally, we design a proactive cache replacement strategy based on RFDL to better fit the optimized cache resource configuration and improve the performance of multi-edge collaborative caching.

### A. Multi-dimensional Cache Space Partitioning

The main steps of the proposed multi-dimensional cache space partitioning are outlined in Algorithm 1, comprising two key components: **multi-dimensional user partitioning** and **cache space partitioning**. In multi-dimensional user partitioning, we classify and segment feature groups with various numbers of users, where user-interest contents are cached individually for different groups. In cache space partitioning, leveraging the established classification, the cache space is perceptually optimized based on user features, user activities, and the divergence of memory access interval.

*(1) Multi-Dimensional User Partitioning*

To a certain extent, user features reflect their preferences for cache contents [33]. As shown in Fig. 2, to accurately predict user preferences, we propose a user partitioning method based on their multi-dimensional features including gender, age, and occupation, which are consecutively coded as coordinate axes, denoted by the set $\Gamma = \{\gamma_1, \gamma_2, ..., \gamma_t, ..., \gamma_T\}$. Thus, this enables the construction of a feature hypercube, where users are grouped to form user intervals. As the partitioning progresses, the user intervals are denoted as the set $H = \{h_1, h_2, ..., h_s, ..., h_S\}$, where $S$ is the number of user intervals and $Grade(h_s)$ indicates the user partitioning grade of user

**Algorithm 1:** The proposed multi-dimensional cache space partitioning

**# Multi-dimensional user partitioning.**
**Initialize:** the user intervals $H = \{h_0\}$, user features $\Theta$, and hyper-parameter $\alpha$.
**while** $|h_s| > \zeta(Grade(h_s))$ **do**
  **for** $l_s^t \in \Theta(h_s)$ **do**
    $l_s^t = l_s^t/2$;
  **end**
  Generate $2^T$ new user intervals;
  Update $H$;
**end**
**# Cache space partitioning.**
**Initialize:** the number of requests $req_m$ received by $e_m$, number of accessible contents $I$, and size of cache space $cache_m$ on $e_m$.
**Input:** $H$ after partitioning.
**for** $h_s \in H$ **do**
  Calculate the user activity $active(h_s)$ and the divergence of memory access interval $diverge(h_s)$ by Eq. (6);
  Realign the cache space of $h_s$ by Eq. (7);
**end**

interval $h_s$. Moreover, the side length of each hypercube is denoted as the set $\Theta(h_s) = \{l_s^1, l_s^2, ..., l_s^t, ..., l_s^T\}$.

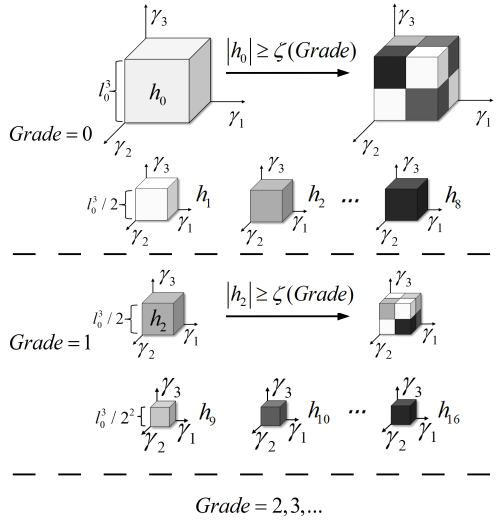



Fig. 2. The proposed multi-dimensional user partitioning.

At the initial stage ($Grade(h_0) = 0$), all users with diverse features are placed within the same user interval $h_0$. If the number of users in $h_0$, represented by $|h_0|$, exceeds the threshold $\zeta(Grade(h_0))$, it will be equally divided into $2^T$ user intervals along each dimension, where the length of each divided dimension will be halved ($l_s^t = l_s^t/2$). $\zeta(Grade(h_s))$ determines the number of users in a user interval. When $\zeta(Grade(h_s))$ is larger, it results in higher user density within each interval, compromising the ability to accurately capture diverse user preferences. Conversely, smaller values of $\zeta(Grade(h_s))$ correspond to reduced user density, risking inaccurate cache predictions. To achieve adaptive partitioning of user intervals and capture the potential relationships between interval users and their preferred contents, we set $\zeta(Grade(h_s)) = \alpha 2^{Grade(h_s)}$ [34], where $\alpha$ is a hyper-parameter. The partitioning in each user interval may continue and go to the following stages (e.g., $Grade = 1, 2, ...$) according to performance requirements.

*(2) Cache Space Partitioning*

Allocating proper cache space for user intervals is important to improve cache performance. Various factors need to be considered during the allocation of cache space, including the number of users, user activities, and the divergence of memory access interval. For example, younger users prefer richer types of contents and show higher activities, while older users may focus on limited contents and exhibit lower activities.

Specifically, in the partitioned user interval $h_s$, the number of users is denoted as $|h_s|$. The user activity and the divergence of memory access interval are defined as

$$active(h_s) = \frac{req_s}{req_m},\ diverge(h_s) = \frac{|\mathbb{C}(h_s)|}{I}, \quad (6)$$

where $req_s$ is the number of user requests and $\mathbb{C}(h_s)$ is the memory access interval of $h_s$.

Considering the above factors, the size of cache space allocated to $h_s$ is defined as

$$cache_s = \frac{\varphi(|h_s|, active(h_s), diverge(h_s)) \times cache_m}{\sum_{s=1}^{S} \varphi(|h_s|, active(h_s), diverge(h_s))}, \quad (7)$$

where $\varphi(\cdot)$ indicates the cumulative product function and $cache_m$ indicates the size of cache space on the edge node connected to $h_s$.

### B. DC-VAE-based Content Popularity Prediction

In real-world scenarios, the content popularity usually follows Zipf's law (*i.e.,* a typical power law), which has been confirmed by analyzing video popularity on mainstream platforms (*e.g.,* MovieLens, iQIYI, Youku, YouTube, and Netflix) [35]. According to Zipf's law, the frequency of content occurrence is inversely proportional to its rank in the frequency table, which indicates that a small fraction of content dominates in popularity while the majority of the content receives less attention. Meanwhile, the less popular content may tend to have similar popularity compared to the more popular ones, making it difficult to distinguish them. Although classic Recurrent Neural Networks (RNN) and its improved variants are good at analyzing sequential data, they struggle to differentiate the content with similar popularity patterns and levels. This limitation complicates the caching strategies based on user preferences, making it hard to establish clear caching boundaries [36].

As a prominent learning algorithm in unsupervised learning, the Variational Auto-Encoder (VAE) employs continuous variables in its hidden layers to efficiently reconstruct compressed input data, facilitating data clustering within the latent space [37]. Specifically, the VAE consists of three main components: encoder, latent space, and decoder. With

an input data $x$, the encoder outputs the parameters $\mu$ and $\log(\sigma^2)$ of the Gaussian distribution $q_\phi(z \mid x) = \mathcal{N}\left(\mu, \sigma^2\right)$, where $z$ is the latent representation. Next, the VAE samples $z$ from $q_\phi(z \mid x) = \mathcal{N}\left(\mu, \sigma^2\right)$ though the reparameterization, where $z = \mu + \sigma \cdot \epsilon$. The decoder is denoted as $p_\theta(x \mid z)$ that reconstructs the compressed input data $x$. During this process, the Evidence Lower Bound (ELBO) is employed to minimize the reconstruction error and regularize the distribution of the latent space, which is defined as

$$ELBO = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathrm{KL}(q_\phi(z|x)||p(z)), \quad (8)$$

where the first term is for reconstruction, and $\mathrm{KL}(\cdot)$ indicates the Kullback-Leibler (KL) divergence that is for regularization. These two terms are used to achieve a balance when fitting a posterior and maintaining the distribution of the latent space.

However, when dealing with the continuous variables in hidden layers, the VAE is susceptible to strong noises or weak signals, leading to the posterior collapse [38]. This issue seriously affects the learning and reconstruction of the original data distribution, thereby causing inaccurate popularity prediction. Specifically, given the substantial influence of significant noises on the latent space representations in VAE, the estimated values for $\mu$ and $\log(\sigma^2)$ may become unstable, thereby rendering an elusive latent variable $z$. Under this situation, the regularization term in Eq. (8) loses its efficacy, as it fails to enforce adherence to the prior distribution. Therefore, the decoder abandons utilizing the latent space, leading to a universal output $x$, which is commonly deemed as a characteristic manifestation of posterior collapse.

When confronted with weak signals, the approximated posterior distribution $q_\phi(z \mid x)$ tends to approach the prior distribution $q_\phi(z)$, denoted by $q_\phi(z \mid x) \simeq q_\phi(z)$, causing the ineffectiveness of KL divergence. As it happens, $ELBO \approx \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$, resembling a conventional Auto-Encoder (AE) that exhibits limited robustness in reconstructing the data with complex distribution. Due to the constrained prior distribution of the latent variable $z$, it fails to fully capture the intricate complexities in the posterior distribution, thereby causing posterior collapse.

To address this issue and prevent the VAE from getting trapped in the solution space of the local optimum, we design a new Discrete-Categorical Variational Auto-Encoder (DC-VAE), which adopts learnable discrete vectors to form the implicit category space, replacing the hidden layers in the classic VAE. When predicting the content popularity, the DC-VAE aims to find the vector in the implicit category space with the closest distance to the output encoding of the encoder network, and then it reconstructs the mapped vector via the decoder network. Fig. 3 illustrates the DC-VAE based content popularity prediction. Specifically, the DC-VAE learns the implicit distribution in the user request matrix $X$, aiming to obtain future user requests in the reconstructed matrix output by the decoder. The user request matrix $X$ contains historical information of user-requested contents on edge nodes, which is defined as

$$X = [x_1, ..., x_n, ..., x_N]^T \in \mathbb{R}^{N \times I}, \quad (9)$$

where $1 \leq n \leq N$, and $n$ indicates the number of users connected to an edge node. $x_n = [x_n^1, ..., x_n^i, ..., x_n^I]^T$ indicates the content request record of the user $n$, where $1 \leq i \leq I$, and $i$ is the index of the content library. $x_n^i = 1$ indicates the successful content request. $x_n^i = 0$ indicates either a failed content request or the content that is not of interest, but these two cases are hard to be distinguished, leading to inaccurate prediction. To solve this issue, we supplement and calibrate the matrix $X$.

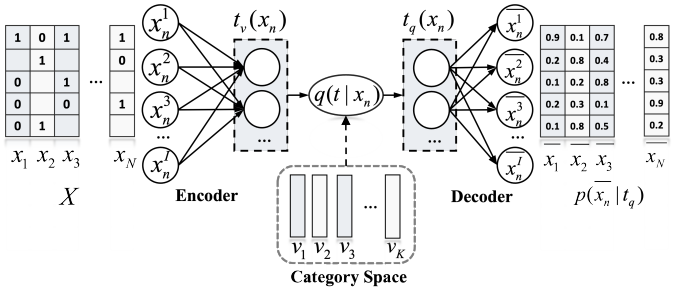



Fig. 3. The proposed DC-VAE based content popularity prediction.

In DC-VAE, the implicit category space is defined as $v \in \mathbb{R}^{K \times D}$, where $K$ is the space size and $D$ is the dimension of the category vector. Thus, there are $K$ category vectors $v_k \in \mathbb{R}^D$ $(k \in 1, 2, ..., K)$. As shown in Fig. 3, the DC-VAE inputs $x_n$ and outputs $t_v(x_n)$ via the encoder network. Next, discrete hidden variable $t$ is calculated by the nearest neighbor algorithm, and the posterior probability distribution $q(t|x_n)$ is one-hot encoding, which is defined as

$$q(t = k|x_n) = \begin{cases} 1, & \text{for } k = \mathrm{argmin}_j \|t_v(x_n) - v_j\|_2 \\ 0, & \text{otherwise} \end{cases}. \quad (10)$$

The input of the decoder is defined as

$$t_q(x_n) = v_k, \quad (11)$$

where $k$ is the index of the decoder input, and it is defined as

$$k = \mathrm{argmin}_j \|t_v(x_n) - v_j\|_2. \quad (12)$$

To address the problem of gradient collapse caused by introducing the implicit category space, we replicate the gradient $\nabla_z L$ from the decoder network to the encoder network during the back-propagation. When training the DC-VAE, the loss function is defined as

$$\begin{aligned} L = &\log p(x_n|t_q(x_n)) + \|sg[t_v(x_n)] - v\|_2^2 \\ &+ \lambda \|t_v(x_n) - sg[v]\|_2^2, \end{aligned} \quad (13)$$

where $\log p(x_n|t_q(x_n))$ is the reconstruction loss, aiming to optimize the encoder and decoder networks. Since the back-propagation gradient is directly replicated to the encoder network, the loss $\log p(t|t_q(x_n))$ is not considered. In $\|sg[t_v(x_n)] - v\|_2^2$, L2 error is used to drive $v_k$ towards $t_v(x_n)$, aiming to optimize the implicit category space. $\lambda \|t_v(x_n) - sg[v]\|_2^2$ is to prevent the encoder output from exceeding the scope of the implicit category space, where $\lambda$ depends on the reconstruction loss, and $sg$ is the stop-gradient

operator that is constant with the partial derivative of 0 during the forward propagation.

Next, the log-likelihood function is defined as

$$\log p(x_n) \approx \log p(x_n|t_q(x_n))p(t_q(x_n)). \tag{14}$$

According to Jensen's Inequality, Eq. (13) is rewritten as

$$\log p(x_n) \geq \log p(x_n|t_q(x_n))p(t_q(x_n)). \tag{15}$$

### C. Robust Federated Deep Learning (RFDL)

There are two key components in the proposed RFDL: **residual-based detection** and **similarity-based federated aggregation**. The residual-based detection is to detect adversarial model updates by parameter ranking. The similarity-based federated aggregation is to avoid the destruction of the globally-shared model by adversarial updating and generates a robust and accurate content popularity prediction model in complex multi-edge environments.

*(1) Residual-based Detection*

For classic FL training, some adversarial model updates may happen, severely affecting model robustness. To address this issue, we design a parameter ranking matrix $\widetilde{R}$ to detect the adversarial updating. Typically, adversarial updates reveal some distinctive features in the ranking domain such as unusual mean and variance [39]. As shown in Fig. 4, we first combine the model parameters from all edge nodes into a matrix $R \in \mathbb{R}^{\theta \times M}$, which is defined as

$$R_{:,m} = \frac{\partial L(w^{(*)}; d_m)}{\partial w^{(*)}}, \tag{16}$$

where the globally-shared mode is parameterized by $w^{(*)}$, and $d_m$ indicates the local training data.

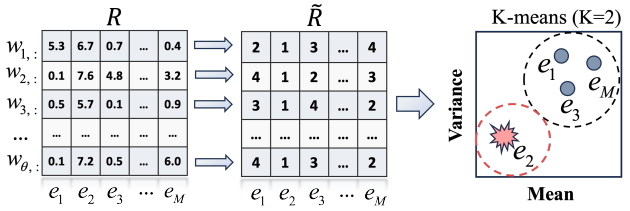



Fig. 4. The proposed residual-based detection for adversarial model updates.

Next, we arrange the elements of each column in $R$ in descending order, retain their sorted positions, and transform them into $\widetilde{R}$. For example, $R(5.3, 6.7, 0.7, 0.4) \rightarrow \widetilde{R}(2, 1, 3, 4)$. Specifically, the mean and variance (var) of $\widetilde{R}$ are defined as

$$\text{mean}_m = \frac{1}{\theta}\sum_{\vartheta=1}^{\theta} \widetilde{R}_{\vartheta,m}, \tag{17}$$

$$\text{var}_m = \frac{1}{\theta}\sum\nolimits_{\vartheta=1}^{\theta} \left(\widetilde{R}_{\vartheta,m} - \text{mean}_m\right)^2. \tag{18}$$

Following the mean and var, we can divide the normal and adversarial model updates into two clusters through the K-means, where the adversarial model updates can be easily identified by the proposed residual-based detection.

**Lemma 1:** *(Detect adversarial attacks with residual-based detection).* The residual-based detection can effectively detect adversarial attacks by leveraging $\text{mean}_m$ and $\text{var}_m$ when the majority of nodes are normal.

For common Byzantine attacks such as Gaussian Noise Attack (GNA) and Sign Flipping Attack (SFA), we assume that the parameter ranks of normal and adversarial nodes comply with the following distributions.

$$B \sim \mathcal{N}(\bar{\mu}_b, \bar{s}_b^2),\ G \sim \mathcal{N}(\bar{\mu}_a, \bar{s}_a^2), \tag{19}$$

where both $B$ and $G$ are $\theta$-dimensional.

In the case of GNA, there are

$$\bar{\mu}_b = \bar{\mu}_a = \frac{M+1}{2}, \tag{20}$$

$$\bar{s}_b^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{b\vartheta},\ \bar{s}_a^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{a\vartheta}, \tag{21}$$

where $Z_{b\vartheta}$ and $Z_{a\vartheta}$ are the functions that count normal and adversarial nodes, respectively.

When it comes to SFA, there are

$$\bar{\mu}_b = \rho \cdot \frac{|E_b|+1}{2} + (1-\rho)\cdot\frac{M+|E_b|+1}{2}, \tag{22}$$

$$\bar{\mu}_a = \rho \cdot \frac{M+|E_b|+1}{2} + (1-\rho)\cdot\frac{|E_a|+1}{2}, \tag{23}$$

$$\bar{s}_b^2 = \rho \cdot O^2_{[1,|E_b|]} + (1-\rho)\cdot O^2_{[|E_a|+1,M]} - \bar{\mu}_b^2), \tag{24}$$

$$\bar{s}_a^2 = \rho \cdot O^2_{[1,|E_a|]} + (1-\rho)\cdot O^2_{[|E_b|+1,M]} - \bar{\mu}_a^2), \tag{25}$$

where $E_b$ and $E_a$ indicate the sets of normal and adversarial nodes, respectively. $|E_b|$ and $|E_a|$ indicate their counts, respectively. $\rho = \lim_{\theta\to\infty}\sum_{\vartheta=1}^{\theta}\mathbb{M}(\mu_\vartheta{>}0)/\theta$ and $O^2_{[o,o']} = (1/(o'-o+1))\sum_{x=o}^{o'} x^2$.

**Proof:** Details are given in Appendix A.

Therefore, by using the proposed residual-based detection method, the edge nodes that offer normal model updates can be filtered, denoted by $E_b = \{e_{b,1}, e_{b,2}, ..., e_{b,M'}\}$.

*(2) Similarity-based Federated Aggregation*

By the aforementioned residual-based detection, we can effectively identify adversarial model updates and proactively mitigate their impact before the FL aggregation. It is noted that a limited number of adversarial nodes may go undetected due to the inadequate sample size available for analysis.

To further avoid the model destruction caused by adversarial model updates, we design a similarity-based federated aggregation method. Specifically, we adopt the Canonical Correlation Analysis (CCA) to measure the similarity between the model updates of each edge node and the average one [40], which determines the weights of different model updates when performing federated aggregation. This process is described as

$$w^{r+1} = \sum_{m=1}^{M'} \frac{|d_m|}{|d|} * \frac{\kappa_m}{\sum \kappa_m} * w_m^r, \tag{26}$$

where $\kappa_m$ indicates the similarity score.

**Lemma 2:** *(Mitigate negative impact with CCA-based federated aggregation).* The CCA-based federated aggregation

can effectively mitigate the negative impact of updates from malicious nodes by leveraging the CCA similarity.

For each filtered node, the CCA similarity to the average one is defined as

$$\kappa_m = \frac{Q^T Cov(V, A)W}{\sqrt{Q^T Cov(V, V)Q}\sqrt{W^T Cov(A, A)W}}, \tag{27}$$

$$\kappa_m = \begin{cases} \kappa_m, & \kappa_m \geq 0 \\ 0, & \kappa_m < 0 \end{cases}, \tag{28}$$

$$\begin{cases} C_{VA}W - \lambda_1 C_{VV}Q = 0, \\ C_{AV}Q - \lambda_2 C_{AA}W = 0, \end{cases} \tag{29}$$

where $V$ is the parameter matrix of the globally-shared model and $A$ is the model parameter matrix of $e_m$. $Q$ and $W$ are the eigenvectors derived from complex functions that involve the variances between $V$ and $A$.

When facing SFA, there is a negative value of $\kappa_m$. As for GNA, there is a reduction in the value of $\kappa_m$.

**Proof:** Details are given in Appendix B.

By integrating the residual-based detection with similarity-based federated aggregation, we propose a novel RFDL, whose key steps are given in Algorithm 2.

**Algorithm 2:** The proposed RFDL

1 **# Update in cloud data center.**
2 **Initialize:** the FL communication round $r_{\max}$ and global prediction model of content popularity $w^{(r)}$.
3 **for** round $r = 1, 2, ..., r_{\max}$ **do**
4  **for** $e_m \in E$ **in parallel do**
5  ${w_m}^{(r+1)} \leftarrow$edge node updates$(w^{(r)}, m)$;
6  **end**
7  Construct $R$ by Eq. (16) and convert it to $\widetilde{R}$;
8  Calculate mean and var of $\widetilde{R}$ by Eqs. (17) and (18);
9  Classify model updates by the K-means;
10  Obtain $E_b$ that provides normal model updates;
11  Generate the globally-shared model by Eq. (26) and distribute it to edge nodes;
12 **end**
13 **# Update in each edge node.**
14 **Initialize:** the training epoch $c_{\max}$, mini-batch $B$, and learning rate $\eta$.
15 **Input:** the globally-shared model $w^{(r)}$.
16 **for** epoch $c = 1, 2, ..., c_{\max}$ **do**
17  **for** batch $b \in B$ **do**
18  Update DC-VAE parameters: $w_m^{(r+1)} \leftarrow w^{(r)} - \eta \nabla L(w^{(r)}; b)$;
19  **end**
20 **end**
21 Upload $w_m^{(r+1)}$ to the cloud data center.

- *Update in cloud data center.* First, we set the total number of FL communication round $r_{\max}$ and initialize the global content popularity prediction model $w^{(r)}$ (Line 2). For every FL communication round, edge nodes update their local models in parallel (Lines 4∼6). Next, the residual-based detection is to identify adversarial model updates and obtain the edge nodes $E_b$ that provide normal model updates (Lines 7∼10). Finally, the globally-shared model is generated by the similarity-based federated aggregation and distributed to edge nodes (Line 11).
- *Update in each edge node.* First, we initialize the training epoch $c_{\max}$, mini-batch $B$, and learning rate $\eta$ (Line 14). With the input of the globally-shared model $w^{(r)}$, each edge node starts its local training (Line 15). For every epoch, the DC-VAE adopts the mini-batch to train and update the local model with the Adam optimizer (Lines 17∼19). After local training, each edge node uploads its latest local model to the cloud data center (Line 21).

### D. Proactive Cache Replacement with RFDL

Based on the proposed RFDL, we design a proactive cache replacement strategy with multi-edge collaboration. The key steps are given in Algorithm 3. For each edge node, we initialize the cache space $cache_{temp}$ and set of user intervals $H$ through the multi-dimensional cache space partitioning (Line 2). While $cache_{temp} \geq 0$, Algorithm 2 is called to predict and sort the content popularity, and the user-interest contents will be placed into the temporary cache library $C_{temp}$ (Line 4). To avoid the cache redundancy caused by overlapping user-interest contents in different intervals, we replace $C_{temp}$ by $C_s$ that selects $cache_h$ most popular contents in the current user interval $h_s$ from $C_{temp}$ (Lines 5∼7). Next, we remove the duplicates in the cache library $C_m$ on each edge node and update the available cache space (Lines 8∼9). The above steps will be repeated until the cache space is fully occupied.

**Algorithm 3:** The proposed RFDL-based proactive cache replacement

1 **for** $e_m \in E$ **in parallel do**
2  **Initialize:** the cache space $cache_{temp} = cache_{all}$ and set of user interval $H$.
3  **while** $cache_{temp} \geq 0$ **do**
4  $C_{temp} \leftarrow$ Call **Algorithm 2** to predict and sort the content popularity;
5  **for** $h = 1, 2, ..., S$ **do**
6  $C_s \leftarrow$ Select $cache_h$ most popular contents in the current user interval $h_s$ from $C_{temp}$;
7  **end**
8  Remove duplicates: $C_m \leftarrow unique(\sum_{h=1}^{S} C_s)$;
9  Update the available cache space: $cache_{temp} = cache_{temp} - cache_h$;
10  **end**
11 **end**

## V. Performance Evaluation

In this section, we introduce the real-world experiment setup and evaluate the proposed *RoCoCache* through extensive comparative experiments.

### A. Experiment Setup

**Real-world Testbed.** We construct a real-world testbed comprising a workstation and 20 Jetson TX2s, as shown in Fig. 5. The workstation acts as the cloud data center, equipped with two NVIDIA GeForce GTX 3090 GPUs, one Intel (R) Xeon (R) CPU Silver 4208 @ 2.10GHz, and 32GB of RAM. The Jetson TX2s act as edge nodes, each equipped with an NVIDIA Pascal GPU with 256 CUDA capable cores, and a CPU processor consisting of a 2-core Denver2 and a 4-core ARM CortexA57. The workstation and Jetson TX2s are deployed on the same network. Based on the FLASK web framework, we built a backend to serve the communication among the workstation and Jetson TX2s. Moreover, the above testbed devices adopt Ubuntu 18.04 OS with CUDA v10.0 and cuDNN v7.5.0.

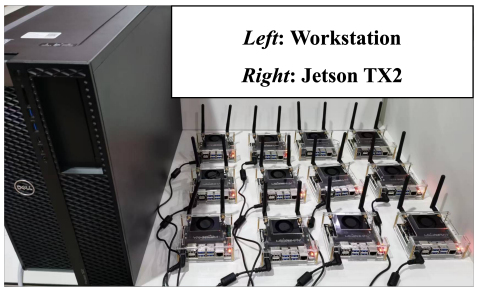



Fig. 5. Real-world testbed for *RoCoCache*.

**Datasets.** As shown in Table I, three real-world datasets are used including MovieLens 100K and 1M [41] and iQIYI [42], which approximately consist of 100,000, 1,000,000, and 10,000,000 request records, respectively. Specifically, the datasets of MovieLens 100K and 1M were collected by GroupLens Research. MovieLens 100K contains approximately 10,000 ratings from 943 anonymous users for 1,682 movie items, and MovieLens 1M contains around 1,000,000 ratings from 6,040 anonymous users for 3,883 movie items. Moreover, the iQIYI dataset [42], published by the iQIYI website, contains about 10,000,000 requests from 100,000 users for 1,900,000 video items. These datasets provide user serial numbers, item indexes, timestamp labels, and user context information. Specifically, we select the user gender, age, and occupation as user features, and the ratings are considered as user requests. The datasets are split into the training (70%), validation (10%), and testing (20%) sets, respectively.

TABLE I
DESCRIPTION OF DATASET DETAILS

| Dataset | Quantity | Users | Movies/Videos |
|---|---|---|---|
| MovieLens 100K [41] | 100,000 | 943 | 1,682 |
| MovieLens 1M [41] | 1,000,000 | 6,040 | 3,883 |
| iQIYI [42] | 10,000,000 | 100,000 | 1,900,000 |

**Parameter Settings.** Based on the above real-world testbed and datasets, we simulate the scenario of multi-edge collaborative caching that consists of one cloud data center, 5∼20 edge nodes, and 943∼100,000 users. The cloud data center stores the complete real-world datasets, each edge node is equipped with a fixed size of cache space, and users are randomly distributed in the service zone of each edge node. We implement the *RoCoCache* based on Python 3.8 and Tensorflow 2.4.0. Specifically, the hyper-parameter $\alpha$ in the multi-dimensional cache space partitioning is 512 [34], the size of the DC-VAE hidden category space $K$ is 128, the dimension $D$ of the category vector $v_c$ is 16, the number of FL communication rounds $r_{\max}$ is 30, the batch size in DC-VAE is 32, the number of training epochs $c_{\max}$ is 300, and the learning rate $\eta$ is 0.001. According to experimental tests, the delays of content requests by accessing local edge nodes, collaborative edge nodes, and the cloud data center are around [2, 4] ms, [15, 20] ms, and [18, 23] ms, respectively.

**Comparison Approaches.** We compare the *RoCoCache* with the optimum and the following state-of-the-art methods.

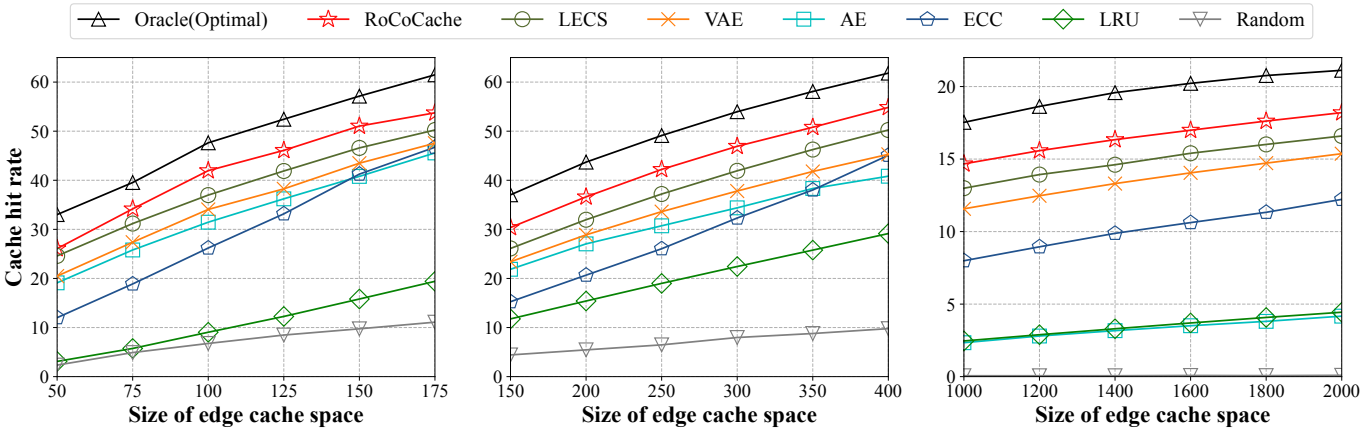
- *Oracle* [43] foreknows all prior information of future user requests, and thus it can obtain the optimal cache hit rate with the limited cache space.
- *Random* [44] randomly selects the requested contents of users to conduct proactive caching.
- *Least Recently Used (LRU)* [23] ranks the least recently used contents according to the request time.
- *Learning-based Cooperative Strategy (LECS)* [26] combines Temporal Convolution Network (TCN) based content popularity prediction with dynamic programming for cooperative caching.
- *Edge Cooperative Caching (ECC)* [45] integrates a neural collaborative filter for content popularity prediction and a greedy algorithm for content delivery.
- *Auto-Encoder (AE)* [46] first reconstructs the input data by using the encoder to compress hidden layers, and then the predicted distribution of content popularity is obtained from the output matrix.
- *Variational Auto-Encoder (VAE)* [15] improves the *AE* and uses continuous variables in hidden layers to reconstruct the compressed input data.

**Attack Models.** We evaluate the robustness of the *RoCoCache* by using the following two attack models.

- *Sign Flipping Attack (SFA)* [47] generates adversarial model updates by reversing the normal model updates, denoted by $w_m^{(r+1)} = -\mu w_m^{(r)}$, where $\mu > 0$.
- *Gaussian Noise Attack (GNA)* [48] generates adversarial model updates by adding the Gaussian random noise to the normal model updates.

### B. Experiment Results and Analysis

**Comparison with State-of-the-Arts.** We conduct comparison experiments in terms of cache hit rate on various datasets with different sizes of edge cache space. As shown in Fig. 6, as the increasing size of edge cache space, the cache hit rate of all methods shows a growing trend. Since the *Oracle* foreknows the prior information of future user requests, it achieves the theoretically-optimal results. The *Random* reveals the worst performance because its caching strategy is blind. The *AE* and *VAE* exhibit good cache hit rates since they compress high-dimensional user requests to low-dimensional representations




(a) Cache hit rate on MovieLens 100K. (b) Cache hit rate on MovieLens 1M. (c) Cache hit rate on iQIYI.

Fig. 6. Comparison between the *RoCoCache* and state-of-the-art methods on various datasets with different sizes of edge cache space.

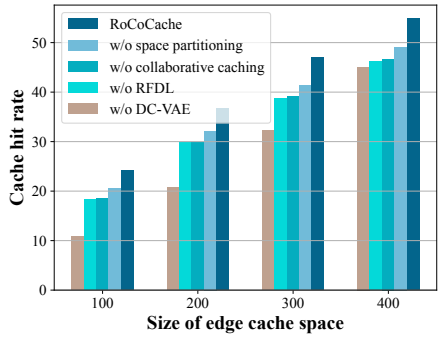



Fig. 7. Ablation studies for the *RoCoCache*.

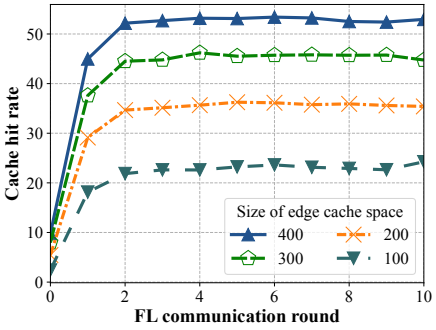



Fig. 8. Convergence of the *RoCoCache*.

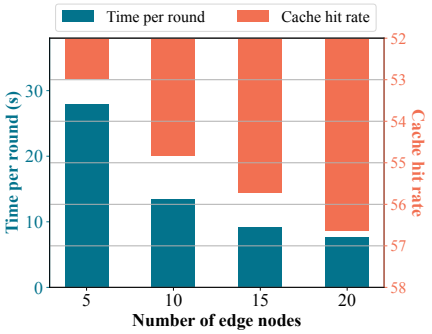



Fig. 9. Training efficiency of the *RoCoCache*.

and learn the potential relationships between user features and requested contents. However, the *AE* suffers from the issue of posterior collapse on iQIYI datasets due to the disparity between the large data volumes and the assumption of a single distribution. Consequently, the *AE* exhibits an extremely low cache hit rate. By using clustering in the continuous hidden space, the *VAE* owns a better ability to reconstruct input distribution than the *AE*, and thus the *VAE* can obtain a more accurate prediction of content popularity and higher cache hit rate. Moreover, the growth of the *ECC* on cache hit rate exhibits a diminishing trend as the edge cache space increases, consistently performing worse than the *RoCoCache*. This is because the collaborative filter employed by the *ECC* cannot precisely capture the content popularity, thereby resulting in inefficient utilization of edge cache space. The *LECS* employs TCN-based content popularity prediction, exhibiting a growth rate comparable to other popularity-driven methods. However, when facing content popularity with similar patterns but different changing magnitudes, the TCN struggles to establish clear boundaries for content caching, seriously affecting the cache hit rate. The *LRU* can well react to burst and sparse content requests but struggles to handle the changeable trend of content popularity. Therefore, the cache performance of the *LRU* is inferior to the *AE*, *VAE*, *ECC*, and *LECS*. Compared to other methods, the *RoCoCache* shows higher cache hit rates that approximate the optimum. This is because the *RoCoCache* realizes the perceptual optimization of edge cache space by multi-dimensional cache space partitioning, and meanwhile solves the posterior collapse problem in *VAE*, leading to more accurate content popularity prediction.

**Ablation Studies.** We conduct ablation studies on the datasets of MovieLens 1M to evaluate the effectiveness of each component designed in *RoCoCache*. As shown in Fig. 7, the cache hit rates of all methods incline as the size of edge cache space increases. In the case of the *RoCoCache w/o DC-VAE*, the DC-VAE-based content popularity prediction is replaced with the classic collaborative filtering. The absence of the DC-VAE significantly hampers the ability of the *RoCoCache* to capture time-series patterns, leading to the lowest cache hit rate in all cases. For the *RoCoCache w/o RFDL*, the cache hit rate is constrained by the cache performance of a single edge node and cannot benefit from the aggregated global model in FL. For the *RoCoCache w/o collaborative caching*, although the global model is applied, the lack of content sharing between edge nodes forces each edge node to retrieve missing content from the cloud data center, resulting in a lower cache hit rate. For the *RoCoCache w/o space partitioning*, there is an obvious increase in cache hit rate due to unrestricted access to neighboring edge nodes' cache space. Therefore, the ablation studies demonstrate the effectiveness of the components designed in *RoCoCache*.

**Convergence Analysis.** Fig. 8 analyze the convergence of

the *RoCoCache* with an increasing number of FL communication rounds on the datasets of MovieLens 1M. The experiments are conducted with the consideration of various sizes of edge cache space to ensure comprehensive testing. At the initial stage, the contents are randomly selected to store in the cache space of edge nodes, resulting in low cache hit rates. After one round of FL communication, the *RoCoCache* generates a preliminary prediction model of content popularity, and thus the cache hit rate is rapidly enhanced. At this time, the *RoCoCache* can achieve more than 80% of its optimal cache performance under the scenarios with different sizes of edge cache space, where the size of edge cache space determines the growth range of cache hit rates. It is worth noting that the *RoCoCache* tends to converge after only six FL communication rounds under different scenarios, demonstrating the excellent convergence of the *RoCoCache*.

**Training Efficiency.** We evaluate the training efficiency of the *RoCoCache* on the datasets of MovieLens 1M under the scenarios with varying numbers of edge nodes. As illustrated in Fig. 9, the per-round time of FL training decreases with an increasing number of edge nodes. When the amount of user content requests remains constant, more edge nodes for collaborative caching can effectively enhance the training efficiency of the *RoCoCache*. Moreover, with the increasing number of edge nodes, the *RoCoCache* can better capture diverse user preferences and improve the cache hit rate by using multi-edge collaboration. The results verify that the *RoCoCache* can adapt to various multi-edge scenarios while achieving excellent training efficiency and cache hit rate.

**Caching Efficiency.** We test the caching efficiency of different methods under the scenario with 10 edge nodes in terms of the delay of content requests. The *Uncollaborative* represents the *RoCoCache* without collaborative caching, and the *Distributed* only caches one copy of contents on each edge node according to the content popularity. As shown in Table II, the delay of content requests declines as the size of edge cache space increases. The *RoCoCache* exhibits the best caching efficiency because it can handle content requests by using different ways and accurately predicting the content popularity. The *Uncollaborative* does not use collaborative caching, and thus it needs to forward the requests of missing contents to the remote cloud, leading to low caching efficiency. Moreover, due to the low cache hit rate, the *Distributed* needs to constantly send content requests to other edge nodes and the remote cloud. Therefore, compared to the *RoCoCache*, the other two methods result in excessive delay.

**Robustness Analysis.** We evaluate the robustness of the *RoCoCache* from two aspects. We first test the ability of the *RoCoCache* to detect adversarial model updates. Fig. 10 illustrates the performance of the residual-based detection model when exposed to 30% and 40% of model updates are adversarial. The separation between the adversarial model updates (red) and the normal model updates (blue) indicates the difference between the two update types in terms of gradient mean and variance. For detecting the *SFA*, the growing proportion of adversarial model updates increases the difficulty of using residual-based detection. In this case, the *RoCoCache* can still distinguish adversarial model updates. For detecting the *GNA*, the larger proportion of Gaussian noise seriously affects the variance of the parameter ranking matrix, and the separation becomes more pronounced.

TABLE II
DELAY (MS) COMPARISON OF CONTENT REQUESTS BETWEEN THE *RoCoCache* AND OTHER METHODS

| Datasets | Cache size | Metohds | | |
|---|---|---|---|---|
| | | *RoCoCache* | *Uncollaborative* | *Distributed* |
| **MovieLens 100K** | 50 | 19.0816 | 19.3212 | 19.7387 |
| | 100 | 16.5272 | 16.7929 | 18.584 |
| | 150 | 14.4693 | 14.8043 | 18.1529 |
| | 200 | 12.7358 | 13.0538 | 18.0789 |
| **MovieLens 1M** | 100 | 19.0244 | 19.1439 | 21.7948 |
| | 200 | 16.5192 | 16.7222 | 20.5736 |
| | 300 | 14.581 | 14.7805 | 19.2618 |
| | 400 | 13.002 | 13.2009 | 18.1803 |
| **iQIYI** | 1400 | 20.2359 | 20.3439 | 21.4851 |
| | 1600 | 20.0793 | 20.1831 | 21.4117 |
| | 1800 | 19.9279 | 20.0292 | 21.3482 |
| | 2000 | 19.767 | 19.8906 | 21.2917 |

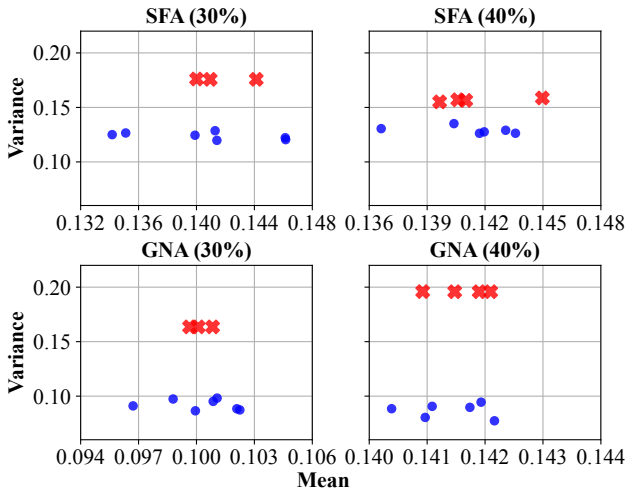
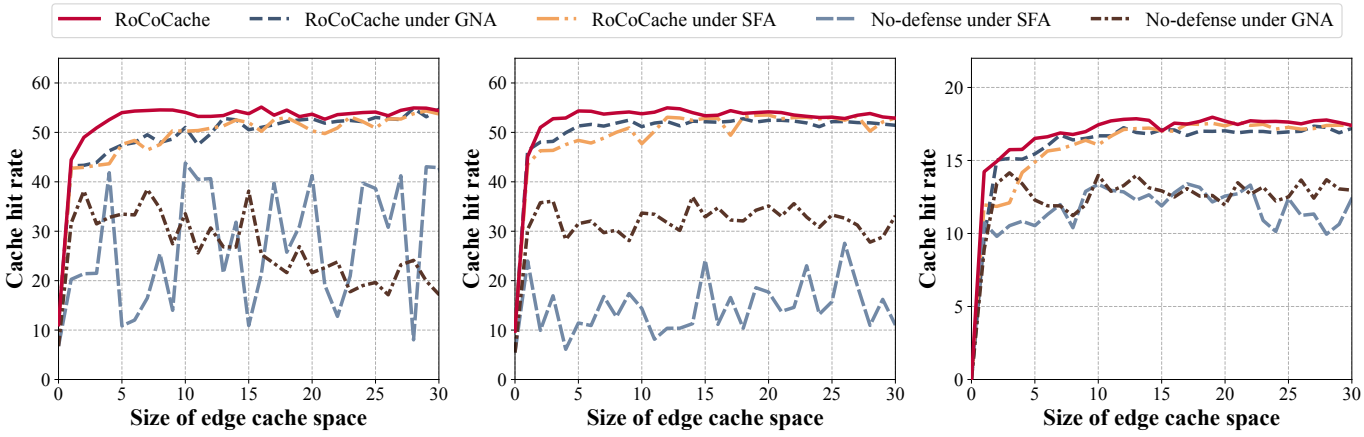



Fig. 10. Performance of residual-based detection for various attacks.

Secondly, Fig. 11 describes the robust performance of the *RoCoCache* under different attacks and defenses when the proportion of adversarial model updates is 30%. When there is no attack, the *RoCoCache* achieves the ideal cache hit rate. Under the attacks of the *SFA* and *GNA*, the *RoCoCache* can still converge after around 20 FL communication rounds and approximate the ideal result. When there is no defense, the cache performance is severely impacted by the *SFA* and *GNA*. Under this situation, it is evident that the adjustment of cache policy is unlikely to significantly impove the cache hit rate. The *SFA* leads to the issue of sign reversion, severely compromising federated aggregation and rendering the globally-shared model invalid. Meanwhile, the cache performance is seriously affected by the *GNA* under the no-defense situation. This is because the Gaussian noise changes the weighted mean and geometric median of the globally-shared model,




(a) Robust performance on MovieLens 100K. (b) Robust performance on MovieLens 1M. (c) Robust performance on iQIYI.

Fig. 11. Robust performance of the *RoCoCache* under different attacks and defenses.

increasing the difficulty of model training. The results verify the strong robustness of the *RoCoCache*, providing accurate identification of adversarial model updates within complex network environments and ensuring rapid model convergence.

## VI. CONCLUSION

In this paper, we propose *RoCoCache*, a resilient collaborative caching framework that uniquely integrates RFDL with proactive caching strategies. First, we design a multi-dimensional cache space partitioning mechanism to optimize edge cache space, providing accurate content recommendations within user interval classification. Next, we develop a DC-VAE based content popularity prediction algorithm, solving the posterior collapse and enhancing the prediction accuracy. Finally, we create a training mode and proactive cache replacement strategy with RFDL for better adaptability and robustness in complex multi-edge environments. Using the real-world testbed and datasets, we demonstrate that the *RoCoCache* achieves a higher cache hit rate than state-of-the-art methods and approximates the optimum. Through ablation studies, we verify the effectiveness of the components designed in *RoCoCache* for improving cache performance. Moreover, the *RoCoCache* exhibits excellent training and caching efficiency under various scenarios. Besides, the *RoCoCache* is able to identify adversarial model updates in complex network environments, validating its good robustness.

Regarding the contributions of the *RoCoCache*, it introduces an innovative robust FL-based collaborative caching framework for multi-edge systems. In *RoCoCache*, the FL is used to unite multiple edge nodes to collaboratively train the popularity prediction model without uploading the content to the remote cloud, protecting user privacy. However, a secure FL framework still needs to be further explored since the existing solutions exhibit non-intuitive susceptibility against malicious attacks [22]. To this end, we design a robust and efficient FL framework for protecting privacy and security, which opens up the research directions of optimizing collaborative caching with high robustness in complex multi-edge systems.

In higher security-constrained and user-centered multi-edge scenarios, the RFDL-based cooperative caching faces some challenges that are worth further research: (1) Privacy and security issues in multi-edge collaborative caching are critical since the popularity-based prediction might inadvertently expose user-sensitive information. Although FL alleviates privacy-leakage risks to a certain extent, it still reveals the vulnerability in model gradients. (2) The common assumption that edge nodes are reliable is flawed because there exist various malicious attacks in real-world scenarios. (3) User mobility brings huge challenges since the content might be outdated as users move to other edge nodes, necessitating the development of mobility-aware caching schemes. (4) Green caching has become increasingly important, and thus it is necessary to design energy-efficient caching strategies to reduce the carbon footprint and operational costs in communication networks.

## APPENDIX

### A. Proof of Lemma 1

According to Kolmogorov's strong law of large numbers, the mean and variance of $\widetilde{R}$ are formulated as

$$\lim_{M\to\infty}\lim_{\theta\to\infty}\text{mean}_m = \bar{\mu}_b\cdot\mathbb{M}(m\in E_b)+\bar{\mu}_a\cdot\mathbb{M}(m\in E_a), \tag{30}$$

$$\lim_{M\to\infty}\lim_{\theta\to\infty}\text{var}_m = \bar{s}_b^2\cdot\mathbb{M}(m\in E_b)+\bar{s}_a^2\cdot\mathbb{M}(m\in E_a). \tag{31}$$

(i) We denote $B_\vartheta(x)$ and $G_\vartheta(x)$ as the cumulative distributions of $B(\cdot)$ and $G(\cdot)$, where $b_\vartheta(x)$ and $g_\vartheta(x)$ are their density functions, respectively. Moreover, the expected rank of $x$ among all participants in the $\vartheta^{th}$ row of the model parameter matrix $\widetilde{R}$ is defined as

$$rank_\vartheta(x) = |E_b|(1-B_\vartheta(x))+|E_a|(1-G_\vartheta(x))+1. \tag{32}$$

For the $\vartheta^{th}$ row, there is

$$\bar{\mu}_b = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta}Y_{b\vartheta}, \bar{\mu}_a = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta}Y_{a\vartheta}, \tag{33}$$

$$\bar{s}_b^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{b\vartheta}, \bar{s}_a^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{a\vartheta}, \tag{34}$$

in which

$$Y_{b\vartheta} = \int_{-\infty}^{\infty} rank_\vartheta(x) b_\vartheta(x) dx, \tag{35}$$

$$Y_{a\vartheta} = \int_{-\infty}^{\infty} rank_\vartheta(x) g_\vartheta(x) dx, \tag{36}$$

$$Z_{b\vartheta} = \int_{-\infty}^{\infty} \left(rank_\vartheta(x) - Y_{b\vartheta}\right)^2 b_\vartheta(x) dx, \tag{37}$$

$$Z_{a\vartheta} = \int_{-\infty}^{\infty} \left(rank_\vartheta(x) - Y_{a\vartheta}\right)^2 g_\vartheta(x) dx. \tag{38}$$

Next, we proceed to analyze the differences in the ranking distribution of $\widetilde{R}$ between normal nodes and adversarial nodes based on the derived Eqs. (33) and (34).

(ii) For GNA, it can be observed from the symmetry of the Gaussian distribution that

$$\lim_{M\to\infty} Y_{b\vartheta} = \lim_{M\to\infty} Y_{a\vartheta} = \lim_{M\to\infty} \mathbb{E}(\widetilde{R}_{\vartheta,m}) = \frac{M+1}{2}, \quad 1 \le m \le M, 1 \le \vartheta \le \theta. \tag{39}$$

Therefore, the mean of the parameter ranks according to Eqs. (35) and (36) can be represented as

$$\bar{\mu}_b = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Y_{b\vartheta} = \frac{M+1}{2}, \tag{40}$$

$$\bar{\mu}_a = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Y_{a\vartheta} = \frac{M+1}{2}. \tag{41}$$

When the number of nodes $M \to \infty$, the variance of $B$ and $G$ becomes clear as follows

$$\lim_{M\to\infty} Z_{b\vartheta} = s_{b,\vartheta}^2, \lim_{M\to\infty} Z_{a\vartheta} = s_{a,\vartheta}^2, \tag{42}$$

$$\bar{s}_b^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{b\vartheta}, \bar{s}_a^2 = \lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{a\vartheta}, \tag{43}$$

where $\bar{s}_b^2$ and $\bar{s}_a^2$ are complex functions. $\bar{s}_b^2 = \bar{s}_a^2$ only if $B$ and $G$ own the same distribution. In this case, the attacks of malicious nodes become invalid. Hence, the efficacy of the attack by SFA malicious nodes manifests solely when they manipulate $\bar{s}^2$, thereby falling into the trap set by our devised detection method.

(iii) For SFA, the values of $\bar{\mu}_b$ and $\bar{\mu}_a$ differ depending on whether the mean ranking of $\vartheta^{th}$ row of $\widetilde{R}$, denoted as $\mu_\vartheta$, is positive or negative. Specifically, when $\mu_\vartheta{>}0$, there are

$$\lim_{M\to\infty} Y_{b\vartheta} = \lim_{M\to\infty} \mathbb{E}(\widetilde{R}_{m,\vartheta}) = \frac{|E_b|+1}{2}, \tag{44}$$

$$\lim_{M\to\infty} Y_{a\vartheta} = \lim_{M\to\infty} \mathbb{E}(\widetilde{R}_{m,\vartheta}) = \frac{M+|E_b|+1}{2}. \tag{45}$$

Conversely, when $\mu_\vartheta{<}0$, there are

$$\lim_{M\to\infty} Y_{b\vartheta} = \lim_{M\to\infty} \mathbb{E}(\widetilde{R}_{m,\vartheta}) = \frac{M+|E_a|+1}{2}, \tag{46}$$

$$\lim_{M\to\infty} Y_{a\vartheta} = \lim_{M\to\infty} \mathbb{E}(\widetilde{R}_{m,\vartheta}) = \frac{|E_a|+1}{2}. \tag{47}$$

According to Eq. (33), there are

$$\begin{aligned}\bar{\mu}_b &= \lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Y_{b\vartheta} \\ &= \rho\cdot\frac{|E_b|+1}{2} + (1-\rho)\cdot\frac{M+|E_a|+1}{2},\end{aligned} \tag{48}$$

$$\begin{aligned}\bar{\mu}_a &= \lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Y_{a\vartheta} \\ &= \rho\cdot\frac{M+|E_b|+1}{2} + (1-\rho)\cdot\frac{|E_a|+1}{2},\end{aligned} \tag{49}$$

where $\rho = \lim_{\theta\to\infty}\sum_{\vartheta=1}^{\theta}\mathbb{M}(\mu_\vartheta{>}0)/\theta$.

Moreover, the variance of $B$ and $G$ can be derived from

$$\begin{aligned}\bar{s}_m^2 &= \lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta} Z_{m,\vartheta} \\ &= \lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta}\mathbb{E}(\widetilde{R}_{m,\vartheta}-\bar{\mu}_m)^2 \\ &= \lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta}(\mathbb{E}(\widetilde{R}_{m,\vartheta}^2) - 2\bar{\mu}_m\mathbb{E}(\widetilde{R}_{m,\vartheta}) + \bar{\mu}_m^2),\end{aligned} \tag{50}$$

in which

$$\begin{aligned}\lim_{M\to\infty}\mathbb{E}(\widetilde{R}_{m,\vartheta}^2) &= O_{[1,|E_b|]}^2\cdot\mathbb{M}(m\in E_b) \\ &\quad + O_{[|E_b|+1,M]}^2\cdot\mathbb{M}(m\in E_a), if\ \mu_\vartheta{>}0,\end{aligned} \tag{51}$$

$$\begin{aligned}\lim_{M\to\infty}\mathbb{E}(\widetilde{R}_{m,\vartheta}^2) &= O_{[1,|E_a|]}^2\cdot\mathbb{M}(m\in E_a) \\ &\quad + O_{[|E_a|+1,M]}^2\cdot\mathbb{M}(m\in E_b), if\ \mu_\vartheta{<}0,\end{aligned} \tag{52}$$

where $O_{[o,o']}^2 = \frac{1}{o'-o+1}\sum_{x=o}^{o'} x^2$.

$\bar{\mu}_m^2$ in Eq. (50) can be calculated by

$$\bar{\mu}_m = \mathbb{E}(\widetilde{R}_{m,\vartheta}), \tag{53}$$

$$\bar{\mu}_m = \bar{\mu}_b\cdot\mathbb{M}(m\in E_b) + \bar{\mu}_a\cdot\mathbb{M}(m\in E_a), \tag{54}$$

$$\bar{\mu}_m^2 = \bar{\mu}_b^2\cdot\mathbb{M}(m\in E_b) + \bar{\mu}_a^2\cdot\mathbb{M}(m\in E_a). \tag{55}$$

Thus, Eq. (50) can be converted to

$$\begin{aligned}&\lim_{M\to\infty}\lim_{\theta\to\infty}\frac{1}{\theta}\sum_{\vartheta=1}^{\theta}(\mathbb{E}(\widetilde{R}_{m,\vartheta}^2) - \bar{\mu}_m^2) \\ &= [\rho\cdot O_{[1,|E_b|]}^2 + (1-\rho)\cdot O_{[|E_a|+1,M]}^2 - \bar{\mu}_b^2)]\cdot\mathbb{M}(m\in E_b) \\ &+ [\rho\cdot O_{[1,|E_a|]}^2 + (1-\rho)\cdot O_{[|E_b|+1,M]}^2 - \bar{\mu}_a^2)]\cdot\mathbb{M}(m\in E_a).\end{aligned} \tag{56}$$

Furthermore, the variance of $B$ and $G$ can be extracted as

$$\bar{s}_b^2 = \rho\cdot O_{[1,|E_b|]}^2 + (1-\rho)\cdot O_{[|E_a|+1,M]}^2 - \bar{\mu}_b^2), \tag{57}$$

$$\bar{s}_a^2 = \rho\cdot O_{[1,|E_a|]}^2 + (1-\rho)\cdot O_{[|E_b|+1,M]}^2 - \bar{\mu}_a^2). \tag{58}$$

It can be stated that $B = G$ and $(\bar{u}_b, \bar{s}_b^2) = (\bar{u}_a, \bar{s}_a^2)$ only if $\rho = \frac{1}{2}$ and $|E_a| = |E_b|$. This condition implies that failed detection occurs. Therefore, the proposed residual-based detection method can promise effective detection of adversarial attacks when the majority of nodes are normal.

### B. Proof of Lemma 2

The globally-shared model is defined as $w^{(*)}$ with the parameter matrix $V$ and it is updated by

$$w^{(*)} = \sum_{m=1}^{M'} \frac{|d_m|}{|d|} w_m^r. \tag{59}$$

Specifically, the parameter matrix obtained from $e_m$ is denoted as $A$. The CCA is used to calculate the similarity between the local models filtered by the residual-based detection and the average one, aiming to identify linear combinations of variables in $V$ and $A$ that exhibit maximum correlation. Hence, the above similarity is defined as

$$\kappa_m = \frac{Cov(V', A')}{\sqrt{Var(V')}\sqrt{Var(A')}}, \tag{60}$$

where $V' = VQ$ and $A' = AW$ are the linear combinations of $V$ and $A$, respectively.

To make simplification, we redefine

$$\begin{aligned} &Cov(V,V) = C_{VV},\ Cov(V,A) = C_{VA},\\ &Cov(A,V) = C_{AV},\ Cov(A,A) = C_{AA}. \end{aligned} \tag{61}$$

Furthermore, according to

$$Cov(V', A') = \frac{1}{j-1}\sum_{j=1}^{\theta}(V_j' - \mathbb{E}(V_j'))^T(A_j' - \mathbb{E}(A_j')), \tag{62}$$

where $\mathbb{E}(V_j') = \mathbb{E}(A_j') = 0$ since $V'$ and $A'$ are centralized $V$ and $A$, Eq. (60) can be converted to

$$\begin{aligned} \kappa_m &= \frac{\mathbb{E}(V'^T A')}{\sqrt{\mathbb{E}(V'^T V')}\sqrt{\mathbb{E}(A'^T A')}}\\ &= \frac{V'^T A'}{\sqrt{V'^T V'}\sqrt{A'^T A'}}\\ &= \frac{Q^T V^T A W}{\sqrt{Q^T V^T V Q}\sqrt{W^T A^T A W}}\\ &= \frac{Q^T C_{VA} W}{\sqrt{Q^T C_{VV} Q}\sqrt{W^T C_{AA} W}}. \end{aligned} \tag{63}$$

Therefore, $\kappa_m$ can be maximized by optimizing $Q^T C_{VA} W$, as the terms $Q^T C_{VV} Q$ and $W^T C_{AA} W$ can be scaled to 1.

According to the Lagrange multiplier method, there are

$$f(Q,W) = Q^T C_{VA} W - \frac{\lambda_1}{2}(Q^T C_{VV} Q - 1) - \frac{\lambda_2}{2}(W^T C_{AA} W - 1), \tag{64}$$

$$\frac{\partial f}{\partial Q} = C_{VA} W - \lambda_1 C_{VV} Q, \tag{65}$$

$$\frac{\partial f}{\partial Q} = C_{AV} Q - \lambda_2 C_{AA} W, \tag{66}$$

$$\begin{cases} C_{VA} W - \lambda_1 C_{VV} Q = 0,\\ C_{AV} Q - \lambda_2 C_{AA} W = 0, \end{cases} \tag{67}$$

$$\begin{cases} Q^T C_{VA} W - \lambda_1 Q^T C_{VV} Q = 0,\\ W^T C_{AV} Q - \lambda_w W^T C_{AA} W = 0, \end{cases} \tag{68}$$

$$\lambda_1 = \lambda_2 = \lambda. \tag{69}$$

Next, Eqs. (67) and (68) can be converted to

$$C_{VA} W = \lambda C_{VV} Q, \tag{70}$$

$$Q = \frac{1}{\lambda} C_{VV}^{-1} C_{VA} W, \tag{71}$$

$$\frac{1}{\lambda} C_{AV} C_{VV}^{-1} C_{VA} W = \lambda C_{AA} W, \tag{72}$$

$$C_{AA}^{-1} C_{AV} C_{VV}^{-1} C_{VA} W = \lambda^2 W, \tag{73}$$

$$C_{VV}^{-1} C_{VA} C_{AA}^{-1} C_{AV} Q = \lambda^2 Q. \tag{74}$$

It is evident that $W$ and $Q$ are the eigenvector sets of $C_{AA}^{-1} C_{AV} C_{VV}^{-1} C_{VA}$ and $C_{VV}^{-1} C_{VA} C_{AA}^{-1} C_{AV}$, where $\lambda^2$ is the common eigenvalue. The CCA contributes to finding the eigenvectors with the maximum eigenvalue for calculating the similarity in Eq. (63).

When confronted with SFA, $V$ can be linearly transformed into the matrix $A$, and there exists a non-singular matrix $\bar{O}$ that satisfies $A = V\bar{O}$. Hence, there is

$$\begin{aligned} C_{AA} &= \mathbb{E}[(A - \mathbb{E}(A))(A - \mathbb{E}(A))^T],\\ &= \mathbb{E}[(V\bar{O} - \mathbb{E}(V\bar{O}))(V\bar{O} - \mathbb{E}(V\bar{O}))^T],\\ &= \mathbb{E}[(V - \mathbb{E}(V)\bar{O}((V - \mathbb{E}(V))\bar{O})^T],\\ &= \mathbb{E}[(V - \mathbb{E}(V))\bar{O}\bar{O}^T(V - \mathbb{E}(V))^T], \end{aligned} \tag{75}$$

where $\bar{O}\bar{O}^T = I$, and thus there is

$$C_{AA} = C_{VV}. \tag{76}$$

According to the property of the covariance,

$$C_{VA} = C_{AV}, \tag{77}$$

and it follows

$$C_{AA}^{-1} C_{AV} C_{VV}^{-1} C_{VA} = C_{VV}^{-1} C_{VA} C_{AA}^{-1} C_{AV}, \tag{78}$$

which leads to the same eigenvalues ($\lambda^2$) and eigenvectors. By selecting the maximum eigenvalue and the corresponding eigenvectors $(Q, W)$, the maximum object in Eq. (63) becomes $C_{VA}$, which is negative when $A$ is opposite of $V$, denoted by

$$C_{VA}^{\mathrm{SFA}} < 0. \tag{79}$$

Therefore, the negative $C_{VA}$ leads to the negative $\kappa_m$.

When facing GNA, the CCA similarity is analyzed by

$$\kappa_m = \frac{Cov(V, A)}{\sqrt{Var(V)}\sqrt{Var(A)}}. \tag{80}$$

Suppose the parameter matrix $V$ that is tampered by GNA is denoted as $V + G$, where $G$ is the Gaussian noise matrix, the CCA similarity can be converted to

$$\kappa_m' = \frac{Cov(V + G, A)}{\sqrt{Var(V + G)}\sqrt{Var(A)}}. \tag{81}$$

Since the GNA occurs, there tends to be a larger variance, and thus

$$Var(V + G) > Var(V). \tag{82}$$

Due to the independence of $G$ from $V$, $Cov(V + G, A)$ and $Cov(V, A)$ do not differ significantly, and there is

$$Cov(V + G, A) \approx Cov(V, A). \tag{83}$$

Thus, the GNA induces the decrease of $\kappa_m$, indicated by

$$\kappa_m > \kappa_m'. \tag{84}$$

Based on the above analysis, we demonstrate that the SFA leads to the negative value of $\kappa_m$ and the GNA causes the reduced value of $\kappa_m$. Therefore, the CCA similarity can facilitate recalibrating the aggregated weights, thereby reaching the goal of robust federated aggregation.

## REFERENCES


[1] D. Ayepah-Mensah, G. Sun, G. O. Boateng, S. Anokye, and G. Liu, "Blockchain-enabled federated learning-based resource allocation and trading for network slicing in 5g," *IEEE/ACM Transactions on Networking (ToN)*, vol. 32, no. 1, pp. 654–669, 2024.

[2] S. Duan, D. Wang, J. Ren, F. Lyu, Y. Zhang, H. Wu, and X. Shen, "Distributed artificial intelligence empowered by end-edge-cloud computing: A survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 591–624, 2023.

[3] Y. Liao, Y. Xu, H. Xu, Z. Yao, L. Wang, and C. Qiao, "Accelerating federated learning with data and model parallelism in edge computing," *IEEE/ACM Transactions on Networking (ToN)*, vol. 32, no. 1, pp. 904–918, 2024.

[4] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Transactions on Intelligent Transportation Systems (TITS)*, vol. 24, no. 2, pp. 2169–2182, 2023.

[5] X. Zhou, Z. Ke, and T. Qiu, "Recommendation-driven multi-cell cooperative caching: A multi-agent reinforcement learning approach," *IEEE Transactions on Mobile Computing (TMC)*, pp. 1–13, 2023.

[6] P. Lin, Z. Ning, Z. Zhang, Y. Liu, F. R. Yu, and V. C. M. Leung, "Joint optimization of preference-aware caching and content migration in cost-efficient mobile edge networks," *IEEE Transactions on Wireless Communications (TWC)*, pp. 1–1, 2023.

[7] Y. Liu, Y. Mao, X. Shang, Z. Liu, and Y. Yang, "Distributed cooperative caching in unreliable edge environments," in *IEEE Conference on Computer Communications (INFOCOM)*, pp. 1049–1058, IEEE, 2022.

[8] T. Zong, C. Li, Y. Lei, G. Li, H. Cao, and Y. Liu, "Cocktail edge caching: Ride dynamic trends of content popularity with ensemble learning," *IEEE/ACM Transactions on Networking (ToN)*, vol. 31, no. 1, pp. 208–219, 2022.

[9] S. Chen, Z. Yao, X. Jiang, J. Yang, and L. Hanzo, "Multi-agent deep reinforcement learning-based cooperative edge caching for ultra-dense next-generation networks," *IEEE Transactions on Communications (TCOM)*, vol. 69, no. 4, pp. 2441–2456, 2020.

[10] Y. Hui, Z. Su, and T. H. Luan, "Collaborative content delivery in software-defined heterogeneous vehicular networks," *IEEE/ACM Transactions on Networking (ToN)*, vol. 28, no. 2, pp. 575–587, 2020.

[11] C.-C. Lin, Y. Chiang, and H.-Y. Wei, "Collaborative edge caching with multiple virtual reality service providers using coalition games," in *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, 2023.

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[13] D. Qiao, S. Guo, D. Liu, S. Long, P. Zhou, and Z. Li, "Adaptive federated deep reinforcement learning for proactive content caching in edge computing," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 33, no. 12, pp. 4767–4782, 2022.

[14] S. Krishnendu, B. Bharath, N. Garg, V. Bhatia, and T. Ratnarajah, "Learning to cache: Federated caching in a cellular network with correlated demands," *IEEE Transactions on Communications (TCOM)*, vol. 70, no. 3, pp. 1653–1665, 2021.

[15] Z. Yu, J. Hu, G. Min, Z. Wang, W. Miao, and S. Li, "Privacy-preserving federated deep learning for cooperative hierarchical caching in fog computing," *IEEE Internet of Things (IoT) Journal*, vol. 9, no. 22, pp. 22246–22255, 2021.

[16] T. Li and L. Song, "Federated adaptive bandits aided caching for heterogeneous edge servers with uncertainty," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1904–1909, IEEE, 2022.

[17] Q. Wu, Y. Zhao, Q. Fan, P. Fan, J. Wang, and C. Zhang, "Mobility-aware cooperative caching in vehicular edge computing based on asynchronous federated and deep reinforcement learning," *IEEE Journal of Selected Topics in Signal Processing (JSTSP)*, vol. 17, no. 1, pp. 66–81, 2023.

[18] Z. Chen, B. Xiong, X. Chen, G. Min, and J. Li, "Joint computation offloading and resource allocation in multi-edge smart communities with personalized federated deep reinforcement learning," *IEEE Transactions on Mobile Computing (TMC)*, vol. 23, no. 12, pp. 11604–11619, 2024.

[19] Y. Wan, Y. Qu, W. Ni, Y. Xiang, L. Gao, and E. Hossain, "Data and model poisoning backdoor attacks on wireless federated learning, and the defense mechanisms: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, pp. 1–1, 2024.

[20] G. Hu, H. Li, W. Fan, and Y. Zhang, "Efficient byzantine-robust and privacy-preserving federated learning on compressive domain," *IEEE Internet of Things (IoT) Journal*, vol. 11, no. 4, pp. 7116–7127, 2024.

[21] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 2938–2948, PMLR, 2020.

[22] B. Mao, J. Liu, Y. Wu, and N. Kato, "Security and privacy on 6g network edge: A survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 1095–1127, 2023.

[23] G. Quan, J. Tan, and A. Eryilmaz, "Counterintuitive characteristics of optimal distributed lru caching over unreliable channels," *IEEE/ACM Transactions on Networking (ToN)*, vol. 28, no. 6, pp. 2461–2474, 2020.

[24] D. Li, H. Zhang, D. Yuan, and M. Zhang, "Learning-based hierarchical edge caching for cloud-aided heterogeneous networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 22, no. 3, pp. 1648–1663, 2022.

[25] S. Liu, C. Zheng, Y. Huang, and T. Q. S. Quek, "Distributed reinforcement learning for privacy-preserving dynamic edge caching," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol. 40, no. 3, pp. 749–760, 2022.

[26] X. Zhang, Z. Qi, G. Min, W. Miao, Q. Fan, and Z. Ma, "Cooperative edge caching based on temporal convolutional networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 9, pp. 2093–2105, 2022.

[27] Z. Jin, T. Song, and W.-K. Jia, "An adaptive cooperative caching strategy for vehicular networks," *IEEE Transactions on Mobile Computing (TMC)*, pp. 1–17, 2024.

[28] J. Peng, Q. Li, X. Tang, D. Zhao, C. Hu, and Y. Jiang, "A cooperative caching system in heterogeneous edge networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 7, pp. 7635–7649, 2024.

[29] L. Cui, X. Su, Z. Ming, Z. Chen, S. Yang, Y. Zhou, and W. Xiao, "Creat: Blockchain-assisted compression algorithm of federated learning for content caching in edge computing," *IEEE Internet of Things (IoT) Journal*, vol. 9, no. 16, pp. 14151–14161, 2020.

[30] D. Feng, G. Huang, C. Feng, B. Cao, Z. Wang, and X. Xia, "Eaps: Edge-assisted privacy-preserving federated prediction systems," in *IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2023.

[31] T. Xie, N. Nambiar, T. He, and P. McDaniel, "Attack resilience of cache replacement policies: A study based on ttl approximation," *IEEE/ACM Transactions on Networking (ToN)*, vol. 30, no. 6, pp. 2433–2447, 2022.

[32] S. Manzoor and A. N. Mian, "Robust federated learning-based content caching over uncertain wireless transmission channels in frans," in *International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, pp. 1–6, IEEE, 2021.

[33] S. Li, J. Xu, M. van der Schaar, and W. Li, "Trend-aware video caching through online learning," *IEEE Transactions on Multimedia (TMM)*, vol. 18, no. 12, pp. 2503–2516, 2016.

[34] Y. Jiang, Y. Wu, F.-C. Zheng, M. Bennis, and X. You, "Federated learning-based content popularity prediction in fog radio access networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 21, no. 6, pp. 3836–3849, 2021.

[35] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1076–1089, 2017.

[36] Y. Jiang, Y. Wu, F. Zheng, M. Bennis, and X. You, "Federated learning-based content popularity prediction in fog radio access networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 21, no. 6, pp. 3836–3849, 2022.

[37] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[38] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[39] W. Zhu, B. Z. H. Zhao, S. Luo, and K. Deng, "Mandera: Malicious node detection in federated learning via ranking," *arXiv preprint arXiv:2110.11736*, 2021.
[40] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *International conference on machine learning*, pp. 3519–3529, PMLR, 2019.
[41] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TIIS)*, vol. 5, no. 4, pp. 1–19, 2015.
[42] S. Qiu, Q. Fan, X. Li, X. Zhang, G. Min, and Y. Lyu, "Oa-cache: Oracle approximation-based cache replacement at the network edge," *IEEE Transactions on Network and Service Management (TNSM)*, vol. 20, no. 3, pp. 3177–3189, 2023.
[43] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.
[44] S. Müller, O. Atan, M. van der Schaar, and A. Klein, "Context-aware proactive content caching with service differentiation in wireless networks," *IEEE Transactions on Wireless Communications (TWC)*, vol. 16, no. 2, pp. 1024–1036, 2016.
[45] Y. Chen, Y. Liu, J. Zhao, and Q. Zhu, "Mobile edge cache strategy based on neural collaborative filtering," *IEEE Access*, vol. 8, pp. 18475–18482, 2020.
[46] Z. Yu, J. Hu, G. Min, H. Lu, Z. Zhao, H. Wang, and N. Georgalas, "Federated learning based proactive content caching in edge computing," in *IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2018.
[47] W. Chen, Z. Zhang, X. Hu, and B. Wu, "Boosting decision-based black-box adversarial attacks with random sign flip," in *European Conference on Computer Vision (ECCV)*, pp. 276–293, Springer, 2020.
[48] S. Li, Y. Cheng, W. Wang, Y. Liu, and T. Chen, "Learning to detect malicious clients for robust federated learning," *arXiv preprint arXiv:2002.00211*, 2020.

**Zheyi Chen** is a Professor and Qishan Scholar with the College of Computer and Data Science at the Fuzhou University, China. He received his Ph.D. degree in Computer Science from the University of Exeter, UK, in 2021, and M.Sc. degree in Computer Science and Technology from the Tsinghua University, China, in 2017, respectively. His research interests include cloud-edge computing, resource optimization, deep learning, and reinforcement learning. Dr. Chen has published over 40 research papers in reputable international journals and conferences such as IEEE TPDS, IEEE JSAC, IEEE TMC, IEEE INFOCOM, ACM SIGKDD, IEEE TII, IEEE ComMag, IEEE TCC, IEEE IoTJ, and IEEE ICC.

**Jie Liang** is working toward his M.S. degree in Computer Science with the College of Computer and Data Science at the Fuzhou University, China. He received his B.S. degree in Software Engineering from the Fujian University of Technology, China. His research interests include edge caching and federated learning.

**Zhengxin Yu** is currently a senior research associate with the School of Computing and Communications at the Lancaster University, UK. She received the Ph.D. degree in Computer Science from the University of Exeter, UK. Her research interests focus on Federated Learning, Deep Learning, Cyber Security, and Multi-access Edge Computing.

**Hongju Cheng** (Senior Member, IEEE) is currently a professor in the College of Computer and Data Science, Fuzhou University. His interests include internet of things, mobile ad hoc networks, wireless sensor networks, and wireless mesh networks. Prof. Cheng received the B.E. and M.E. degrees in EE from Wuhan University of Hydraulic and Electric Engineering, in 1997 and 2000, respectively, and the PhD in Computer Science from Wuhan University in 2007. Since 2007, he has been with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. He is serving as editor/guest editor for more than 10 international journals. He has published almost 80 papers in international journals and conferences.

**Geyong Min** is a Professor of High Performance Computing and Networking in the Department of Computer Science within the Faculty of Environment, Science and Economy at the University of Exeter, United Kingdom. He received the Ph.D. degree in Computing Science from the University of Glasgow, United Kingdom, in 2003, and the B.Sc. degree in Computer Science from Huazhong University of Science and Technology, China, in 1995. His research interests include future Internet, computer networks, wireless communications, multimedia systems, information security, high-performance computing, ubiquitous computing, modelling, and performance engineering.

**Jie Li** (Fellow, IEEE) received the B.E. degree in computer science from Zhejiang University, Hangzhou, China, the M.E. degree in electronic engineering and communication systems from China Academy of Posts and Telecommunications, Beijing, China. He received the Dr. Eng. degree from the University of Electro-Communications, Tokyo, Japan. He is with Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China where he is a chair professor. His current research interests are in big data and AI, blockchain, edge computing, networking and security, OS, information system architecture. He serves as the director of Shanghai Jiao Tong University Blockchain Research Centre. He was a professor in Department of Computer Science, University of Tsukuba, Japan. He was a visiting Professor in Yale University, USA, Inria Sophia Antipolis and Inria Grenoble-Rhone-Aples, France. He is the co-chair of IEEE Technical Community on Big Data and the founding Chair of IEEE ComSoc Technical Committee on Big Data. He serves as an associated editor for many IEEE journals and transactions. He has also served on the program committees for several international conferences.