IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 14 February 2024, accepted 26 March 2024, date of publication 1 April 2024, date of current version 5 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3383474

RESEARCH ARTICLE

Blockchain-Based Secure Content Caching and Computation for Edge Computing

ELIF BOZKAYA-ARAS, (Member, IEEE)
Department of Computer Engineering, National Defence University Turkish Naval Academy, 34942 İstanbul, Turkey
BTS Group, 34469 İstanbul, Turkey

e-mail: ebozkaya@dho.edu.tr

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) 1515 Frontier Research and Development Laboratories Support Program for BTS Advanced Artificial Intelligence (AI) Hub: BTS Autonomous Networks and Data Innovation Laboratory under Project 5239903.

ABSTRACT The current explosion in user traffic necessitates the placement of edge servers in proximity to the Internet of Things (IoT) devices, allowing computation tasks to be offloaded to edge servers. This strategy aims to minimize the average delay of traffic requests by enabling user/IoT devices to locally execute time-sensitive tasks or offload them to edge servers within the edge computing paradigm. This new paradigm will also allow to cache contents at edge servers, but considering such an increasing number of user requests and limited storage capability of edge servers, selection in edge caching decisions is challenging. In addition, while there is general consensus that this technology may provide a variety of benefits, there are serious questions about its security implications. This is because malicious users can manipulate the caching decisions of the edge servers by sending fake traffic requests, which reduces the caching efficiency of the resource-constrained edge servers. Driven by these issues, in this paper, we propose a blockchain-based content caching and computation strategy to validate the authenticity of cached content and thus prevent unauthorized requests from malicious users. Specifically, the Proof of Stake (PoS) consensus mechanism is presented to handle low computational work, validate the process of blocks, and manage the transactions between edge servers and legitimate users. Then, a Deep Q Network (DQN)-based solution is proposed to intelligently develop an effective content caching and computation strategy. According to performance evaluation, the proposed model significantly outperforms the conventional caching strategies. It improves the cache hit rate by up to 8.2% on average and reduces the response delay by up to 7.45% on average.

INDEX TERMS Blockchain, content caching, PoS, edge computing, Internet of Things.

I. INTRODUCTION

Industry 5.0 will fundamentally transform the industrial environment by maximizing human-robot collaboration-enabled automation and digitization. In this transformation, 6G is expected to integrate innovative technologies including the Internet of Things (IoT), Artificial Intelligence (AI), Machine Learning (ML), and Blockchain. Recently, edge computing has also emerged as a critical technology for designing and analyzing next-generation wireless networks. This technology has been developed around the ''edge servers at the network edge'' to offload the computation tasks and reduce processing latency by adding new perspectives to network management.

According to *Ericsson Mobility Report* [1], total mobile network traffic reached around 160 *EB* per month at the end of 2023 and is expected that it will raise 563 *EB* per month by the end of 2029. We can expect that Internet-related user traffic will increase tremendously, most of which will be video traffic. Tackling this traffic increase has become a priority for network operators in the communications industry. While handling traffic demands is a major challenge, the advances in the edge computing paradigm could offer a solution.

6G networks will be challenging to realize due to the stringent latency and computing constraints for most

The associate editor coordinating the review of this manuscript and approving it for publication was Rakesh Matam.

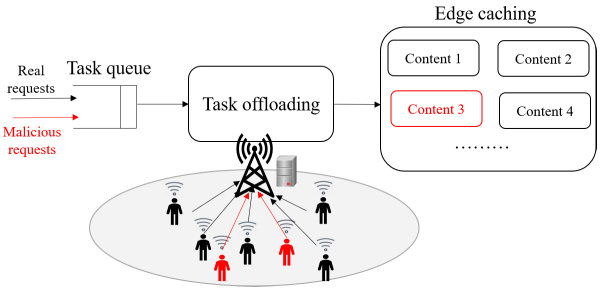

FIGURE 1. An illustration of malicious task requests in the edge computing system.

latency-sensitive tasks (i.e., video streaming, XR-type services) [2]. On one hand, the edge computing paradigm enables both services and network functions to be placed close to the users to reduce latency and bandwidth, and thereby, computation tasks are offloaded from user devices to the edge servers [3], [4]. On the other hand, a fundamental issue in mobile edge computing is determining the content caching strategy at the edge servers; that is, which data should be stored at the network edge to maximize system performance since obtaining content from the remote content servers experiences huge latency [5]. This helps in reducing latency, improving Quality of Service (QoS) and network efficiency, and enhancing user experience. However, as seen in Fig. 1, malicious users can manipulate the caching decisions of the edge server by sending fake computation requests, which decreases the cache hit rate and increases service cost. In this context, edge security cannot be ignored to prevent malicious requests and inefficient network management. Thus, in this paper, we propose a novel framework for secure content caching and computation for edge computing with the collaboration of blockchain technology to efficiently manage cached content and validate the authenticity of the content.

Blockchain is a decentralized and distributed technology that provides transparency, security, and trust mechanisms by keeping a tamper-resistant record of transactions [6]. Essentially, blockchain can ensure the data integrity of transmission between edge servers and user devices by storing the critical information in the blockchain. In addition, blockchain can be used to implement security mechanisms, such as authentication between edge servers and user devices with smart contracts [7]. Smart contracts are programs on the blockchain for executing an agreement and reliable transactions without third-party intervention. Once the smart contract is implemented on the blockchain, its code cannot be changed. This enables the integrity, security, and authenticity of transactions. In our model, blockchain is used to validate the authenticity of the cached content, thereby preventing unauthorized requests from malicious users.

As a result, in this paper, we propose a blockchain-based secure content caching and computation framework in edge computing systems. Given the large amount of traffic in today's wireless networks, we design a content caching solution with an improved cache hit rate and shorter average delay. We integrate the blockchain technology into a content caching strategy to prevent unauthorized user requests and allow traceable, irreversible, and reliable transactions. This paper also explores a novel content caching approach to improve the QoS that uses a Deep Q Network (DQN)-based solution. To achieve this, we first formulate the task computation time for local computation and edge server computation, and then content requests are assigned to a popularity class as high popularity and low popularity. By employing the popularity classification in conjunction with a DQN-based solution, our model outperforms the conventional caching strategies.

The main contributions of this paper are as follows:

- We design a blockchain-based secure content caching and computation model in the edge computing system. We propose the Proof of Stake (PoS) consensus mechanism to prevent malicious requests to validate the process of blocks and manage the transactions between edge servers and legitimate users.
- We also present an efficient algorithm for content caching strategy based on DQN to intelligently decide the contents on edge servers. By doing this, we aim to improve content retrieval time and decrease average delay on resource-constrained edge servers, and ensure that each request is served with better QoS.
- We conduct a set of experiments and demonstrate cache hit rate, average delay, and offloading ratio performance by comparing traditional cache replacement strategies. We show that our model can effectively enhance system performance in an edge computing system.

The remaining work of this paper is organized as follows. Related work on edge caching and blockchain-based security solutions are discussed in Section II. Blockchain-based network architecture and threat model are presented in Section III. The proposed blockchain-based content caching and computation model is given in Section IV. The performance evaluation of the proposed model is discussed in Section V and finally, Section VI presents the conclusion and future work.

II. RELATED WORK

In this section, we summarize the research in two categories: edge caching and blockchain-based security.

A. EDGE CACHING

Edge computing technology allows data to be processed on components close to the source. However, offloading the tasks to the edge servers and caching the contents create difficulties due to the limited storage capacity of edge servers.

In [8], the authors consider the challenges of the heterogeneity of the network and spatial-temporal characteristics of content popularity. They model the edge caching problem as a Markov Decision Process (MDP) and propose an actor-critic learning method to improve system performance in terms of average delay, cache hit rate, and traffic offloading

ratio. In [9], the authors aim to optimize edge caching and computation offloading by designing an intelligent framework between user devices and edge nodes. For each request, the edge node decides whether to cache or not. Accordingly, a cache replacement problem is also modeled as an MDP and solved by the Deep Reinforcement Learning method. Although these works optimize the caching and communication between user devices and edge nodes in mobile edge systems, due to the poor security of computation and storage in edge computing architecture, there are problems with malicious task requests. In [10], the authors address the problem of caching interrupts due to the vehicle movement, and a caching decision strategy is defined based on a Recursive Deep Reinforcement Learning algorithm to decrease the service delay and improve the caching hit ratio. Similarly, in [11], the authors focus on vehicle movement and consider roadside units as edge nodes to cache the popular contents. They model the edge caching problem and propose a coalition game-based distributed caching scheme to improve caching resource utilization and hit ratio. In [12], the authors investigate resource allocation and caching strategies for cloud-to-things systems to minimize latency and freshness of information (i.e., Age of Information). However, these studies still have limitations because the integration of blockchain into edge computing systems improves the security of data transmission. In [13], the authors propose a searchable and secure edge caching scheme for intelligent 6G systems. In the study, edge nodes pre-cache user-requested data based on the user's position and direction. However, due to security threats of edge computing systems, an attacker may send large numbers of fake content to the cache to decrease the cache-hit rate of legitimate users. Thus, the security of both edge nodes and user devices has a significant impact on the development of secure edge caching systems [14].

This state-of-the-art motivates us to intelligently build a decision-making process for secure content caching at edge nodes. However, designing a secure content caching and task computation architecture is still open. In this paper, we explicitly bridge that gap by defining a blockchain-based framework and presenting a deep learning-based solution.

B. BLOCKCHAIN-BASED SECURITY

IoT/user devices generate large amounts of content, and computationally expensive operations impede the development of models on user devices. To overcome this limitation, the edge computing paradigm is applied to reduce storage and computation cost, and blockchain is used to secure data storage and computation at the network edge.

Blockchain is a promising technology in the computing domain when it comes to securing information shared between different entities of a network, such as between edge servers and user devices. Moreover, by leveraging smart contracts, blockchain can provide transparent execution, where the outcome is verified and approved by a majority of the mining nodes within the network [15]. Smart contracts run on the blockchain, include predefined rules to be executed by communicating parties and provide access control only to authorized users in a decentralized manner. Thus, the storage of content data on edge servers can be restricted by smart contracts only according to the requests of authorized/legitimate users so that it defines how requests can be managed and what steps to take when malicious nodes are detected. The consensus mechanisms ensure the correct implementation of smart contracts [16]. Consensus mechanisms include various types of consensus algorithms, such as Proof of Work (PoW), Proof of Elapsed Time (PoET), Proof of Stake (PoS), etc. These consensus algorithms are used to build trust and properly store the transactions on the blocks [17].

In [5], the authors address the security issues, where edge servers may return false results or viruses to users. In this regard, they present a trust management procedure between users and edge nodes. To achieve this, a trust degree of the edge nodes is defined and updated by users based on the QoS. Then, an algorithm is proposed to manage the caching resources of edge nodes. However, the authors do not discuss authentication efficiency and additional costs for the blockchain-based scheme. In [18], the authors propose a distributed blockchain cloud architecture to allocate edge cache resources and improve the QoS in IoT networks. They present a content selection algorithm for edge nodes and aim to improve the utilization of cache space. Blockchain is used to build a distributed architecture to address the security challenge of increasing data volume. However, the proposed content cache strategy does not consider different contents to improve the utilization of cache space. In [19], the authors consider data tampering and eavesdropping attacks and present a blockchain-assisted framework to provide the security of historical data and optimize content caching probability for 6G networks. In [20], the authors focus on edge computing for speeding up the response time of traffic requests and blockchain for ensuring the security of data transmission. A blockchain-based algorithm is designed to predict the popular files and decide which files to cache to improve the cache hit rate. Similarly, in [21], the authors address the challenge of content caching in edge computing-assisted blockchain networks and propose a deep reinforcement learning-based solution to improve the system performance in terms of transmission delay and caching reward. In these works, popular files are cached on the edge nodes and thereby, IoT/user devices can access popular content from edge nodes faster than obtaining the same content from the remote cloud. However, it is not possible to cache all contents at the network edge and these studies do not consider the computation capability of edge nodes, and how to securely offload the tasks is not discussed.

In [22], the authors present a game theory approach to optimize the edge servers and user devices in blockchain networks. Then, the caching and pricing solutions are formulated for a higher caching utilization. In [23], the

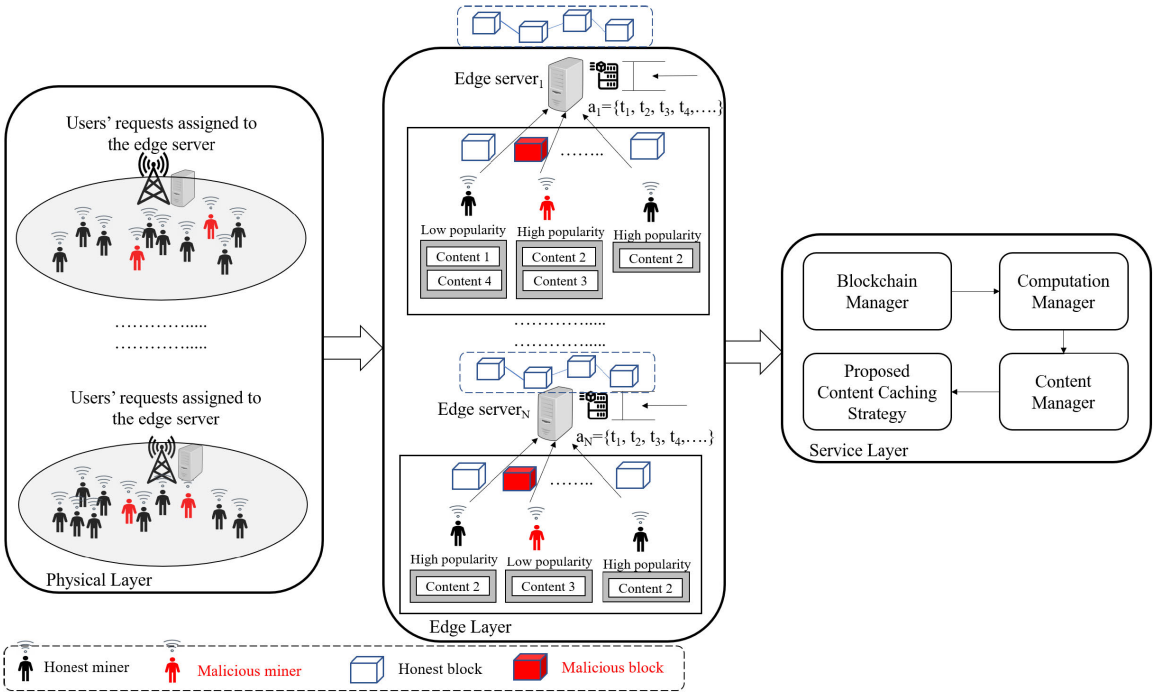

FIGURE 2. Proposed blockchain-based model for content caching and computation.

authors design a distributed and trusted authentication system in a blockchain-based edge network. They consider an elliptic curve cryptography-based approach to guarantee security at the network edge. An edge caching strategy is defined and compared with traditional caching strategies in terms of cache hit rate and delay. Although a distributed authentication mechanism protects the data validity, when the number of user requests increases, the network becomes congested due to the limited cache size and the time required to create a new block for each participating entity to the network. In [24], the authors present a combination of centralized and blockchain-based authentication architecture for edge computing-based IoT systems. They consider centralized authentication for edge nodes and blockchain-based authentication for IoT devices. Authentication efficiency is evaluated in terms of computation costs of transactions, processing and storage resources, and processing time in the blockchain network. However, a computation offloading and edge caching strategy will inevitably bring communication overhead to the network, which is not considered in the proposed model.

Although these studies have mostly focused on allocating caching resources of edge nodes, how to prevent unauthorized requests and manage caching resources is not addressed.

III. SYSTEM MODEL

A. NETWORK ARCHITECTURE

The proposed network architecture of the blockchain-based content caching and computation model is given in Fig. 2. As shown in the figure, the proposed model consists of three components: (i) Physical Layer, (ii) Edge Layer, and (iii) Service Layer. The physical layer includes $N = (1, 2, .., i, .., N)$ users and $M = (1, 2, .., j, .., M)$ edge server-enabled base stations at fixed locations. The locations of edge servers are defined as $L(j)$, where $j \in M$. Here, edge servers have a limited storage capacity to cache contents and serve the users within the communication range. Users may execute the computation tasks locally or offload them to the edge layer as seen in Fig. 2. The connection between the user and the edge server is expressed as follows:

$$\begin{bmatrix} e_{1,1} & e_{1,2} & \dots & e_{1,M} \\ e_{2,1} & e_{2,2} & \dots & e_{2,M} \\ \dots & \dots & \dots & \dots \\ e_{N,1} & e_{2,2} & \dots & e_{N,M} \end{bmatrix} \tag{1}$$

where $e_{i,j} \in \{0, 1\}$, and $e_{i,j} = 1$ indicates that user i is in the communication area of edge server j, otherwise it is equal to 0.

Edge servers have an important role in processing user requests at the network edge and caching these requests with blockchain. Each edge server connects to the cloud server and downloads the requested content if it is not in the cache. The blockchain guarantees the edge security and authenticity of the cached contents.

The service layer is responsible for the execution of the proposed blockchain-based content caching and computation model. This layer runs the blockchain manager, computation manager, content manager, and proposed DQN-based content caching strategy modules, which will be detailed in the next section.

B. THREAT MODEL

As seen in Fig. 2, unauthorized/malicious users can make requests to increase computational density and latency on the edge servers. For example, an edge server can receive fake content requested by a malicious user, execute the computation task and transmit it to the malicious user. Also, the requested content can be cached. In this case, malicious users can hamper or even shut down the system operations since the cache can be filled with the contents of the malicious users.

IV. PROPOSED BLOCKCHAIN-BASED CONTENT CACHING AND COMPUTATION MODEL

The most common services launched by the service providers are Fixed Wireless Access (FWA), Enhanced Mobile Broadband (eMBB), and XR-based services [1]. FWA provides wireless broadband connectivity for different indoor and outdoor use cases. Accordingly, users can generate different content types. The Radiocommunication Sector of ITU (ITU-R) has determined the following scenarios for 2020 and beyond [25]: (i) *eMBB* for addressing the surge in data rates, high user density and providing substantial traffic capacity for hotspots, (ii) *Ultra Reliable and Low Latency Communications (URLLC)* for stringent requirements such as latency, reliability, throughput etc. based on delay-sensitive applications, and (iii) *Massive Machine Type Communications (mMTC)* for a massive number of connected devices to transmit low volumes of delay-insensitive data. Here, each content may have a different popularity. Thus, in this paper, we classify the contents into high-popularity content and low-popularity content.

In this section, the service layer functionalities and blockchain construction are explained as follows.

A. BLOCKCHAIN MANAGER

The blockchain manager is responsible for managing the transactions received from legitimate users and smart contracts between the edge server and users. The details are as follows:

• *Smart contract:* Smart contract is responsible for executing a code on blockchain that allows traceable, irreversible, and reliable transactions. In this paper, edge servers are designed to run smart contracts.

The blockchain manager in the edge server stores the user device information and adds the transaction in blocks and it is responsible to create a smart contract between the edge server and users [26]. A smart contract is designed to execute a blockchain process, which stores the caching transaction in a local database. The communication diagram between the edge server and the user device is illustrated in Fig. 3 and explained below.

- *Step 1:* Each user device registers to the associated edge server and sends its device information. Then, the edge server sends to the blockchain manager to add the blockchain network. The blockchain manager generates a user device ID and sends it to the edge server. A smart contract is created and signed between the edge server and the user device.
- *Step 2:* The user creates a transaction which includes transaction id, user device id, edge server id, nonce, content request, the hash value of the previous block, and the hash value of the whole transaction. Each user sends the transaction to the associated edge server. The transaction is encrypted with a secret key using Secure Hash Algorithm (SHA)-256.
- *Step 3:* The edge server verifies the transaction with the same secret key using SHA-256.
- *Step 4:* The user requests are executed by the edge server and sent to the user.
- *Step 5:* Also, the content requests may be stored in a local database by the edge server according to the proposed content caching strategy.

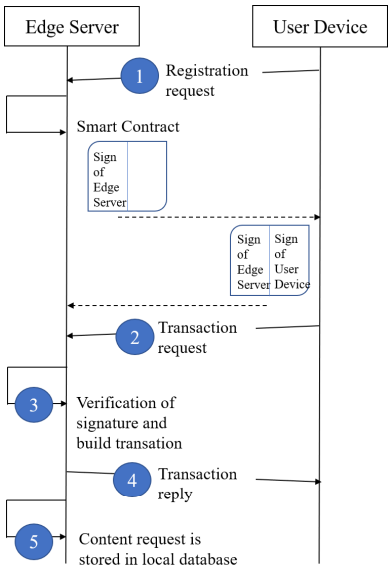

FIGURE 3. Communication diagram between edge server and user device.

• *Transaction:* The legitimate users send traffic/content requests to the edge server in the form of transactions as seen in Fig. 4. This transaction cannot be copied by a malicious user since a chained checksum of SHA-256 is implemented to maintain a trust relationship between the edge server and the user.

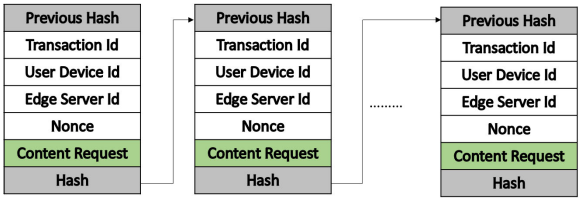

FIGURE 4. Transaction block.

• *Consensus mechanism:* The main goal of the consensus mechanism is to build a trust relationship between distributed

nodes [7]. The security of the blockchain depends on the consensus mechanism [6]. Transactions are recorded on the blockchain via a consensus algorithm. There are different consensus algorithms, such as PoW, PoET, PoS, etc. PoW, one of the most common consensus algorithms, solves computationally intensive problems and requires a lot of computational resources [17]. This is because PoW provides a mechanism to establish decentralized consensus in blockchain networks by motivating mining nodes to dedicate computational resources to validate transactions. However, this requires a lot of time and high energy consumption to complete. Therefore, this approach has led to the development of alternative consensus mechanisms like PoET. In the PoET algorithm, a random waiting time is assigned to each node. After waiting for the specified time, the first node that gets the shortest waiting time is the winner and may add its block to the network [17]. However, PoET is based on the assumption that the trusted execution environment is secure and reliable. If the trusted execution environment is controlled by a single participant, a risk of centralization may arise, which potentially manipulates the random waiting time process. In our model, PoS is chosen to validate and add the transactions to the blockchain. PoS is executed based on randomly generated stakes of users accepted into the blockchain. The reason for choosing the PoS is to meet the latency requirements of computation tasks. PoS solves long delays and extensive energy consumption problems in the transaction processing of PoW [27].

The pseudo-code of the PoS-based secure processing is given in Alg. 1. In our approach, the blockchain is used to connect multiple edge servers, thus enabling information sharing between edge servers on the network. When a user device requests to establish a secure connection with the associated edge server, the device first sends a registration request along with its certificate. The certificate is signed by a trusted authority with its private key and can be verified with its public key. Thus, an edge server can verify the legitimate user's certificate. If the certificate is verified, the user device can send content requests to the edge server, otherwise, the user device is not included in the system (lines 5-11). Then, the blockchain infrastructure runs the smart contract which stores and verifies the transactions to avoid malicious requests. The smart contract is responsible for the management of each entity, including edge servers and IoT/user devices, and provides the interface for the recording of secure content on edge servers. Thereby, network security can be handled by multiple edge servers thanks to the smart contracts. Accordingly, each server receives the content requests from user devices and creates the transactions to add the blockchain (lines 12-15). As a new block is added to the blockchain, the functions, called the *Computation Manager* and *Content Manager*, which will be detailed in the next subsection, are executed for the content caching and computation strategy according to the cache capacity of the edge server (lines 16-18).

Algorithm 1 PoS-Based Secure Processing

1: **Require:** Certificate, Traffic requests from all user devices
2: **Ensure:** Updated blockchain
3: Initialize the miners for M edge servers
4: **for all** miner $j \in M$ do in parallel **do**
5: **for all** user in the coverage area of miner j **do**
6: *Registration_Request*(*Certificate*)
7: **if** authentication is verified **then**
8: Add to the blockchain as an honest miner
9: **else**
10: Label the user as a malicious miner
11: **end if**
12: Each edge server receives content request
13: Generate a transaction by defining its own stake
14: Calculate the whole hash of the transaction block
15: Add it to the blockchain
16: **while** There is a new block in blockchain **do**
17: Send the block to the *Computation_Manager* (Section IV-B) and *Content_Manager* (Section IV-C) functions to update the content caching and computation strategy according to the cache capacity of the edge server
18: **end while**
19: **end for**
20: **end for**

B. COMPUTATION MANAGER

As the number of computation tasks is increased, delay-sensitive tasks cannot be locally executed by the user devices. In this case, users offload the tasks to the edge servers. In the paper, we classify the contents as high popularity and low popularity. The reason to implement this is because improving the cache hit rate will also reduce the service time of computation tasks.

In this subsection, we will define the computation time and details of the delay analysis.

1) Local Computation Time: Let's assume that the computing capability of an IoT/user device i as Υ_i, the required CPU cycle per bit as c_i, the size of computation task as χ_i. Then, the computation time of the task at the user device is calculated as [28].

$$t_i^{local} = \chi_i c_i / \Upsilon_i \tag{2}$$

2) Edge Server Computation Time: Computation time of the task at the j_{th} edge server to process the i_{th} IoT/user device request is calculated as

$$t_i^{edge_j} = \chi_i c_j / \Upsilon_{i,j} \tag{3}$$

where $\Upsilon_{i,j}$ is the computing capability of j_{th} edge server for i_{th} IoT/user device.

Each server has a capacity of C such that when the edge server is overloaded, waiting time in the queue is also analyzed.

3) Queuing Delay: In this module, each content request is assigned to a popularity class; high popularity and low popularity. We assume that the arrivals of the higher popularity class and lower popularity class have mean arrival rates λ_1 and λ_2 with Poisson distribution, respectively. The total arrival rate is $\lambda \equiv \lambda_1 + \lambda_2$. In our popularity-based selection, the highest popularity data packet is served ahead of the lowest popularity, but there is no preemption. Each edge server has a service capability and we model it with an M/M/1 queuing system. The service distribution of both classes is exponential with the same rate μ.

According to the M/M/1 queuing system, the expected number of traffic requests in the system is expressed as follows for the high popularity class, $L_q^{(1)}$ and the low popularity class, $L_q^{(2)}$, respectively [29].

$$L_q^{(1)} = \frac{\lambda_1 \rho}{\mu - \lambda_1} \tag{4}$$

and

$$L_q^{(2)} = \frac{\lambda_2 \rho}{(\mu - \lambda_1)(1 - \rho)} \tag{5}$$

where $\rho = \lambda_1/\mu + \lambda_2/\mu$.

Then, according to the equation $W_q^{(i)} = \lambda_i L_q^{(i)}$, waiting time in the queue is expressed as follows for the high popularity class, $W_q^{(1)}$ and the low popularity class, $W_q^{(2)}$, respectively [29].

$$W_q^{(1)} = \frac{\rho}{\mu - \lambda_1} \tag{6}$$

and

$$W_q^{(2)} = \frac{\rho}{(\mu - \lambda_1)(1 - \rho)} \tag{7}$$

C. CONTENT MANAGER

Caching policies decide which tasks are cached and when they are removed from the cache based on an optimization problem. In this paper, we aim to optimize the service delay and securely cache the contents to improve the cache hit rate based on the proposed deep learning-based approach.

We assume that there are $\mathbb{F} = \{1, 2, .., f, \ldots F\}$ different contents, and users may request these contents. We classify the content requests into two categories: high popularity and low popularity. Content popularity is defined as the probability distribution of content requests from all users in the system and measures the interest of users in the contents [30]. It is modeled by a Mandelbrot-Zipf (MZipf) distribution. MZipf distribution aims to model the occurrence frequency of events with parameters that control the scaling behavior of the distribution and describes the probability of requesting a content, f. It is defined as follows [31].

$$P_f = \frac{(R_f + \epsilon)^{-\alpha}}{\sum_{i \in \mathbb{F}} (R_i + \epsilon)^{-\alpha}} \tag{8}$$

where R_f is the rank of content f in the descending order of content popularity, $\alpha > 0$ is the skewness factor, and $\epsilon \geq 0$ is the plateau factor [30]. The skewness factor quantifies the degree of asymmetry in a distribution. It serves as a measure indicating how much a particular distribution deviates from a standard normal distribution. The plateau factor controls the shape of the distribution of the left-most part. It determines the proportion of total requests attributed to user devices with the lowest ranks [31].

Each edge server can cache various contents to meet the requirements of computation requests from user/IoT devices. Once a user device offloads the computation task to the edge server, the edge server can check whether the requested content is in the cache. If it is not in the cache, the edge server can download it from neighbor edge servers or cloud server. At the same time, the proposed strategy is updated to increase the cache hit rate. In addition, increasing the cache size can improve the cache hit rate. However, this results in more memory and high cost. Therefore, instead of traditional cache replacement policies such as First In First Out, Least Recently Used, implementing an intelligent content caching and computation strategy makes it possible to improve the cache hit rate without increasing the cost.

D. SECURITY ANALYSIS

One of the critical steps in this model is to provide a secure connection for each participating entity to the network to avoid malicious users. Therefore, we implement an authentication mechanism based on the blockchain infrastructure. Attackers/malicious users may send fake content requests and cause high computation overhead on the edge server. In order to prevent this, each user device has a certificate, $C = E_{K_{auth}^-}[ID_i, K_i^+, Time]$ from a trusted authority that contains the user device ID, the public key of the user device (K_i^+), and the expiration date. The certificate is signed by the trusted authority with its private key, (K_{auth}^-). The edge node can verify the certificate using the public key of the trusted authority. If a malicious node sends a fake certificate, the user's request will be rejected. In addition, we use the blockchain to guarantee the authenticity of the cached content with the hash value since a chained checksum is implemented. Each block contains content request and hash value with nonce. Here, even if the same content is requested from the edge server more than once by the user, the hash value will be unique each time due to the nonce. Each block is encrypted with a secret key using SHA-256. Only legitimate entities can access secret keys. The smart contract can verify the correctness of the block between the edge server and the user device. Thus, a malicious device cannot impersonate legitimate devices.

E. DEEP Q NETWORK-BASED CONTENT CACHING STRATEGY

Given the above problem formulation, our goal is to develop an effective content caching and computation strategy scheme that can satisfy the QoS requirement of each user. The service layer performs comprehensive analysis with a DQN-based method to minimize the service delay and improve cache efficiency.

1) ARCHITECTURE OF DEEP Q NETWORK

The traditional Q learning algorithm is one of the widely used model-free reinforcement learning methods. It provides the agent with the ability to find and learn the best action in a given state without the need to create maps of domains [32]. However, traditional Q learning cannot handle the task computation strategy since the requests of users are unpredictable and change dynamically. As the number of traffic requests increases, the amount of memory required to save and update the Q-table will increase. In this regard, we use a neural network to learn the environment, give a content caching and offloading decision, and approximate the Q-values.

This paper proposes a Deep Q Network (DQN) based scheme whose main idea is to obtain a more stable training procedure and then design a learning agent able to compute an effective strategy. DQN is an application of Deep Reinforcement Learning (DRL). DQN has two neural networks. The first neural network, called the main network, is used to update the network parameters in each iteration, and the second, called the target network, is used to compute the target and it has the same architecture with the first network. Initially, all DQNs are initialized with random parameters and the agent chooses the minimum cost (computation delay) at each iteration, which also improves the cache hit rate.

The environment dynamics are defined as follows.

- State: The state contains all the network information to give the best action. The input of the network state is a three-dimensional array. Each user submits its content request in the form (λ_i, R, D, L) to the edge server. Here, Here, λ_i is the packet arrival rate. R indicates the data rate, D is the delay constraints and L is the packet length. According to the relationship between the edge servers and the users, defined in Eq. 1, the workload of each edge server is tracked.
- Action: An agent performs an action by calculating the local computation time and edge server computation based on the content popularity. The chosen action defines the next states. Neural networks can help the agent learn the best actions. Accordingly, the user can locally process the task or offload it to the edge server.
- Reward: The reward function is the action taken by the agent to make the right decision in the interaction with the environment over time. Our objective is to maximize rewards to minimize task computation time through cached resources. If the requested content f is in the cache, the hit is 1, otherwise it is 0. The cache hit rate is calculated by dividing all hits by the total number of requests. Thus, we consider the computation time as the key performance indicator to increase the cache hit rate. The reward function is defined as follows.

$$r(t) = -\sum_{i=1}^{N} \left(x_{i,j,f} P_f (t_i^{edge} + W_q^{(\cdot)}) + (1 - x_{i,j,f}) t_i^{local} \right) \tag{9}$$

where $x_{i,j,f} \in \{0, 1\}$ is a binary variable. $x_{i,j,f} = 1$ means that user device i requests the content f from edge server j, otherwise $x_{i,j,f} = 0$ and it means that the task is executed by the user device. $W_q^{(\cdot)}$ indicates the waiting time in the queue as given in Eqs. 6 and 7. Apparently, maximizing the expected reward equals minimizing the overall computation time based on the content popularity.

- Loss function: DQN learns the features in the state and the loss function uses the difference between best actions as indicated by the Q value of both the main network and target network. The loss function is the mean squared error of the predicted Q value and the target Q value, Q*, which updates the weight optimizing the cost.

2) OPERATION PROCEDURE OF DQN

The main components of the DQN are described in the previous subsection. During the procedure, each model will compute an offloading strategy to minimize the cost and improve cache efficiency. The main steps of the proposed strategy are summarized as follows.

• *Step 1:* Every user submits the content requests to the nearest edge server.

• *Step 2:* The contents are classified as high and low popularity content. All experiences and computation requirements are stored in an experience pool, which is represented by (s_t, a_t, r_t, s_{t+1}). This is where the biggest difference between DQN and Q learning emerges. In Q learning, the current and past experiences are included in the process, while in DQN, the learning process is more effective and improved with previous experiences included in the experience pool. Thus, DQN updates the weights using the loss function.

• *Step 3:* At each state, an action is determined whether to reconfigure the content caching and offloading strategy to minimize the cost.

• *Step 4:* The next action is defined by the maximum output of the Q network. Here, local and edge server computation times are performed. The challenge is to find a content caching and task computation strategy that specifies how to decide the cache content and perform tasks with an acceptable QoS.

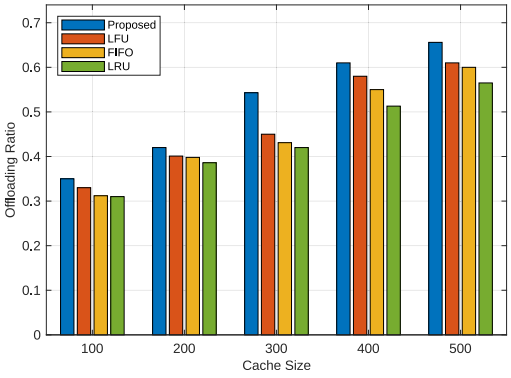

FIGURE 5. Task offloading ratio with different cache replacement strategies.

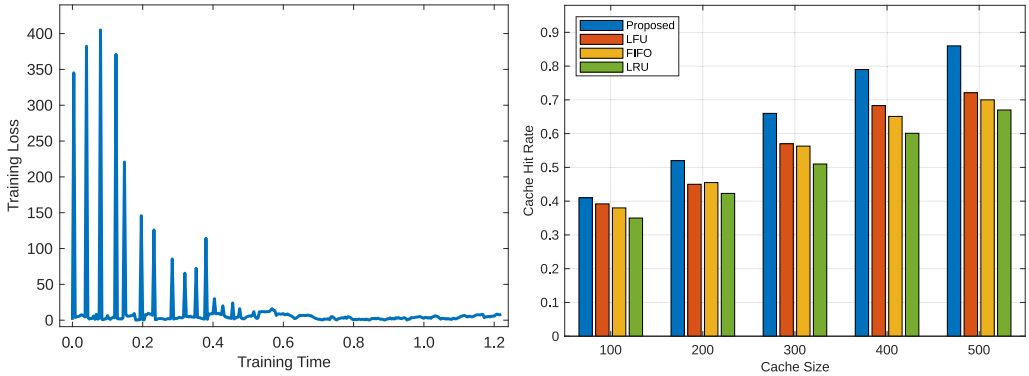

FIGURE 6. (a) Training loss with the proposed edge caching strategy (b) Cache hit rate with different cache replacement strategies.

• *Step 5:* The procedure from Step 3 to 4 is repeated until the system finds a solution that fulfills a required reward value.

V. PERFORMANCE EVALUATION

The simulation is implemented in an edge computing scenario, where a 1000×1000 environment with 23 base stations and a varying number of users, similar to [33]. The arrival of computation tasks transmitted by each user of its associated edge server-enabled base station has a Poisson distribution with a rate of $(0, 0.2]$. The total number of contents is $F = 10$ and the size varies between $100KB$ and $1MB$. MZipt distribution is applied to simulate the content popularity characteristic. The cache size of each edge server is the same and is set to $[100MB - 500MB]$ in different experiments. The users are randomly distributed and each user is assigned to the nearest edge server. The required CPU cycle for a task is 1000 Megacycles.

We carry out a DQN algorithm and choose the hyperbolic tangent activation function and Adam optimizer. We use a single-layer fully-connected feed-forward neural network, which consists of 200 neurons. The parameters for each edge server are the same. The experience pool size is set to 2000, the batch size is 256, the episode number is set as 200, the discount factor is 0.9, the exploration probability is set as 0.001, the learning rate is set as 0.05 and the period of replacing the target Q network is set as 250.

We consider different content caching strategies. Accordingly, content caching strategies can be divided into two categories; reactive cache placement strategies and proactive cache placement strategy. Reactive cache placement strategies are simple and effective policies in traditional architectures. Thus, we compare our results with the following reactive edge caching strategies:

- Proposed Approach: The proposed DQN-based proactive edge caching strategy
- First In First Out (FIFO): As the cache is full, this algorithm checks the content list and deletes the content with the longest waiting time in the cache [34].
- Least Recently Used (LRU): The LRU algorithm arranges the cache list according to the recent use of contents. The most recently unused content is always deleted from the cache [34].
- Least Frequently Used (LFU): When the cache is full, the least frequently used content is removed from the cache [35].

A. PERFORMANCE PARAMETERS

We measure the following performance parameters to evaluate our proposed model.

- Cache Hit Rate: It is the ratio of the number of hits to the number of all requests.
- Average Delay: It is measured by the sum of the task computation time and queuing delay.
- PoS Processing Time: To validate the transactions in the blockchain, the processing time is measured for an acceptable QoS requirement.
- Offloading Ratio: It is the ratio of the total number of offloaded tasks to the edge server to the total number of tasks.

B. RESULTS

In Fig. 5, we first illustrate the relationship between task offloading ratio and cache size. As seen in the figure, a larger cache size means that more contents can be cached and users can offload the computation tasks to the edge servers for processing. The proposed model has a much higher offloading ratio compared to traditional strategies. This achieves significant improvement on both cache hit rate and service time, thereby the storage and computation limitations of user devices can be alleviated.

Fig. 6(a) shows the training loss of the proposed model during the training time. The proposed edge caching strategy shows a fast convergence speed and small fluctuations within a certain range. This is because the proposed model first collects the transaction requests of users in each training step, then decides the content caching strategy according to the task offloading decision for the later training. In Fig. 6(b),

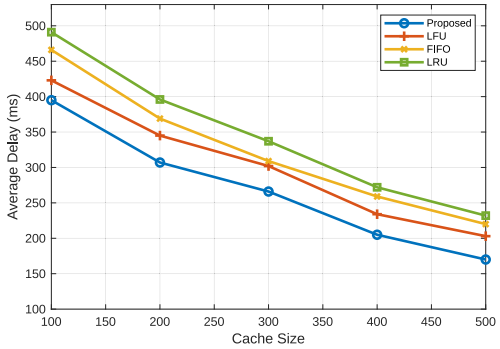

FIGURE 7. Average delay with different cache replacement strategies.

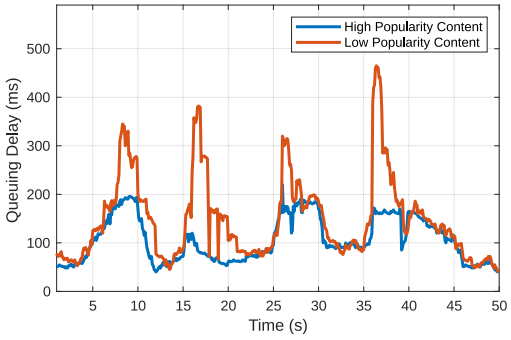

FIGURE 8. Queue Delay (ms) over time (s) for high popularity content and low popularity content.

we show the relationship between cache hit rate and cache size for different cache replacement strategies. As the cache size increases, more contents can be cached in the edge servers and the efficiency of different strategies also increases. As seen in the figure, the cache hit rate is significantly better than the LFU, FIFO, and LRU algorithms. Our model shows an average increase of 8.2%, 7.94%, and 7.78% in cache hit rate compared to LFU, FIFO, and LRU, respectively.

Fig. 7 compares the average delay for different cache replacement strategies. As expected, as the cache size increases, the average delay also decreases since more contents are cached in the edge servers. The proposed DQN-based content caching strategy outperforms the traditional strategies. The performance improvement and the importance of secure content caching can be seen in the figure. In particular, compared to traditional strategies, the proposed model increases content retrieval time and decreases delay. This reduces the data traffic and provides better QoS. When compared to LFU, FIFO, and LRU, our model shows an average increase of 6.4%, 7.12%, and 7.45% in service delay, respectively.

In Fig. 8, we also provide queue delay over simulation time according to the Eqs. 6-7 when the cache size is equal to 500. As explained in Section IV-B, each content request has been assigned to a popularity class. As seen in the figure, with this approach, it has been observed that more popular content is cached and thus there is an improvement in both

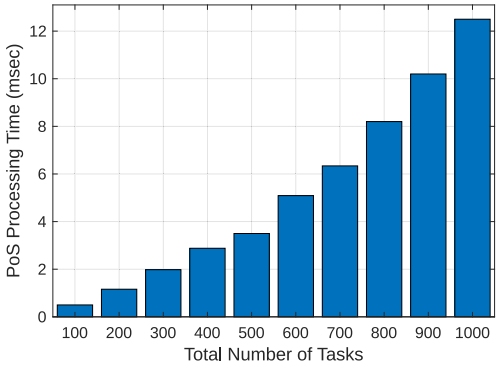

FIGURE 9. PoS processing time for edge servers as the number of computation tasks increases.

cache hit rate and service time in subsequent requests. With the proposed model, the resources of local and edge devices are observed and an acceptable QoS is provided to the users for edge caching services.

Finally, in Fig. 9, we observe the PoS processing time for edge security. As the number of computation tasks is increased, the processing time is given. Here, PoS-based processing is executed by edge servers to validate and add the transactions to the blockchain. When the number of tasks increases, the computation requests wait longer to be processed by edge servers. Although PoS processing time increases the average delay, it prevents unauthorized requests from malicious users and provides the authenticity of cached content for edge security.

VI. CONCLUSION

In this paper, we propose a blockchain-based secure content caching and computation model in edge computing systems. We demonstrate how blockchain can enhance security to prevent malicious traffic requests while providing authenticity of the cached content and trustworthiness. First, we design a blockchain-based network architecture to define a secure content caching and computation strategy composed of the physical layer, edge layer, and service layer. Second, we give the communication diagram between edge servers and user devices to manage the transactions received from legitimate users and present PoS-based secure processing between edge servers and users. Then, we formulate the content caching and computation problem and present a DQN-based strategy to increase the cache hit rate and decrease the service delay. Eventually, the effectiveness of our proposed framework is compared with traditional cache replacement strategies. Simulation results have demonstrated that the proposed blockchain-based secure content caching and computation model can provide both cache efficiency and delay minimization. In this problem domain, energy efficiency is also an essential factor. Ignoring the consumed energy by IoT/user devices and edge servers means that a significant amount of consumed electricity. Hence, in future work, we will also analyze our proposed model in terms of energy efficiency.

REFERENCES

[1] (Nov. 2023). *Ericsson Mobility Report*. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2023

[2] E. Bozkaya, T. Bilen, M. Erel-Özçevik, and Y. Özçevik, "Energy-aware task scheduling for digital twin edge networks in 6G," in *Proc. Int. Conf. Smart Appl., Commun. Netw. (SmartNets)*, Jul. 2023, pp. 1–6.

[3] E. Bozkaya, "Digital twin-assisted and mobility-aware service migration in mobile edge computing," *Comput. Netw.*, vol. 231, Jul. 2023, Art. no. 109798. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128623002438

[4] E. Bozkaya, B. Canberk, and S. Schmidt, "Digital twin-empowered resource allocation for 6G-enabled massive IoT," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, Jun. 2023, pp. 727–732.

[5] Q. Xu, Z. Su, and Q. Yang, "Blockchain-based trustworthy edge caching scheme for mobile cyber-physical system," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1098–1110, Feb. 2020.

[6] E. Bozkaya, M. Erel-Özçevik, T. Bilen, and Y. Özçevik, "Proof of evaluation-based energy and delay aware computation offloading for digital twin edge network," *Ad Hoc Netw.*, vol. 149, Oct. 2023, Art. no. 103254. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870523001749

[7] H. Xue, D. Chen, N. Zhang, H.-N. Dai, and K. Yu, "Integration of blockchain and edge computing in Internet of Things: A survey," *Future Gener. Comput. Syst.*, vol. 144, pp. 307–326, Jul. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22003521

[8] C. Wang, R. Li, X. Wang, T. Taleb, S. Guo, Y. Sun, and V. C. M. Leung, "Heterogeneous edge caching based on actor-critic learning with attention mechanism aiding," *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 6, pp. 3409–3420, Nov./Dec. 2023.

[9] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep. 2019.

[10] H. Wu, B. Wang, H. Ma, and L. Xing, "Collaborative caching relay algorithm based on recursive deep reinforcement learning in mobile vehicle edge network," *Ad Hoc Netw.*, vol. 152, Jan. 2024, Art. no. 103313. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870523002330

[11] J. Lin, S. Huang, H. Zhang, X. Yang, and P. Zhao, "A novel coalition game based distributed cooperative content caching in mobile edge networks," *Veh. Commun.*, vol. 44, Dec. 2023, Art. no. 100689. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214209623001195

[12] I. Zyrianoff, L. Gigli, F. Montori, L. Sciullo, C. Kamienski, and M. Di Felice, "CACHE-IT: A distributed architecture for proactive edge caching in heterogeneous IoT scenarios," *Ad Hoc Netw.*, vol. 156, Apr. 2024, Art. no. 103413. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870524000246

[13] C. Wang, T. Zhou, J. Shen, W. Wang, and X. Zhou, "Searchable and secure edge pre-cache scheme for intelligent 6G wireless systems," *Future Gener. Comput. Syst.*, vol. 140, pp. 129–137, Mar. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22003296

[14] B. Bolat-Akça and E. Bozkaya, "Software-defined intrusion detection system for DDoS attacks in IoT edge networks," in *Proc. IEEE Int. Conf. Dependable, Autonomic Secure Comput., Int. Conf. Pervasive Intell. Comput., Int. Conf. Cloud Big Data Comput., Int. Conf. Cyber Sci. Technol. Congr. (DASC/PiCom/CBDCom/CyberSciTech)*, Nov. 2023, pp. 672–677.

[15] T. A. Butt, R. Iqbal, K. Salah, M. Aloqaily, and Y. Jararweh, "Privacy management in social Internet of Vehicles: Review, challenges and blockchain based solutions," *IEEE Access*, vol. 7, pp. 79694–79713, 2019.

[16] S. Wang, Y. Yuan, X. Wang, J. Li, R. Qin, and F.-Y. Wang, "An overview of smart contract: Architecture, applications, and future trends," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 108–113.

[17] H. Guo and X. Yu, "A survey on blockchain technology and its security," *Blockchain, Res. Appl.*, vol. 3, no. 2, Jun. 2022, Art. no. 100067. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2096720922000070

[18] H. Wang, Y. Li, X. Zhao, and F. Yang, "An algorithm based on Markov chain to improve edge cache hit ratio for blockchain-enabled IoT," *China Commun.*, vol. 17, no. 9, pp. 66–76, Sep. 2020.

[19] W. Sun, S. Li, and Y. Zhang, "Edge caching in blockchain empowered 6G," *China Commun.*, vol. 18, no. 1, pp. 1–17, Jan. 2021.

[20] 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 Things J.*, vol. 9, no. 16, pp. 14151–14161, Aug. 2022.

[21] M. Chen, G. Wu, Y. Zhang, Y. Lin, Y. Zhang, and J. Li, "Distributed deep reinforcement learning-based content caching in edge computing-enabled blockchain networks," in *Proc. 13th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2021, pp. 1–5.

[22] Y. Yang, Z. Liu, Z. Liu, Y. Xie, K. Y. Chan, and X. Guan, "Joint optimization of edge computing resource pricing and wireless caching for blockchain-driven networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6661–6670, Jun. 2022.

[23] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1972–1983, Mar. 2020.

[24] O. A. Khashan and N. M. Khafajah, "Efficient hybrid centralized and blockchain-based authentication architecture for heterogeneous IoT systems," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 35, no. 2, pp. 726–739, Feb. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1319157823000113

[25] *IMT Vision—Framework and Overall Objectives of the Future Development of IMT for 2020 and Beyond*, document Recommendation ITU-R M.2083-0, Sep. 2015. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.2083-0-201509-I!!PDF-E.pdf

[26] T. Bilen, M. Erel-Özçevik, E. Bozkaya, and Y. Özçevik, "Work-in-progress: Merkle tree-based secure routing for digital twin-assisted aircraft network in 6G wireless," in *Proc. IEEE Int. Black Sea Conf. Commun. Netw. (BlackSeaCom)*, Jul. 2023, pp. 420–425.

[27] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: Fundamentals, applications and opportunities," *IEEE Access*, vol. 7, pp. 85727–85745, 2019.

[28] N. Kiran, C. Pan, S. Wang, and C. Yin, "Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks," *J. Commun. Netw.*, vol. 22, no. 1, pp. 1–11, Feb. 2020.

[29] D. Gross, J. F. Shortle, J. M. Thompson, and C. M. Harris, *Fundamentals of Queueing Theory*, 4th ed. Hoboken, NJ, USA: Wiley, 2008.

[30] X. Li, X. Wang, P.-J. Wan, Z. Han, and V. C. M. Leung, "Hierarchical edge caching in device-to-device aided mobile networks: Modeling, optimization, and design," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1768–1785, Aug. 2018.

[31] M. Hefeeda and O. Saleh, "Traffic modeling and proportional partial caching for peer-to-peer systems," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1447–1460, Dec. 2008.

[32] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 279–292, 1992.

[33] C. Yi, J. Cai, T. Zhang, K. Zhu, B. Chen, and Q. Wu, "Workload reallocation for edge computing with server collaboration: A cooperative queueing game approach," *IEEE Trans. Mobile Comput.*, vol. 22, no. 5, pp. 3095–3111, May 2023.

[34] G. Jia, G. Han, H. Xie, and J. Du, "Hybrid-LRU caching for optimizing data storage and retrieval in edge computing-based wearable sensors," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1342–1351, Apr. 2019.

[35] M. Milon Uddin and J. Park, "360 degree video caching with LRU & LFU," in *Proc. IEEE 12th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Dec. 2021, pp. 45–50.

ELIF BOZKAYA-ARAS (Member, IEEE) received the M.Sc. and Ph.D. degrees in computer engineering from Istanbul Technical University, İstanbul, Turkey, in 2015 and 2020, respectively. She is currently an Assistant Professor with the Department of Computer Engineering, National Defence University, Turkish Naval Academy, İstanbul. She was a Visiting Researcher with the Faculty of Computer Science, University of Vienna, Austria, from February 2019 to August 2019. She is a recipient of the IEEE INFOCOM Best Paper Award, in 2018. She has also been awarded the "Best Ph.D. Thesis" by Istanbul Technical University, in 2020. Her research interests include edge computing, cloud computing, AI-enabled aerial networks, digital twin, and the IoT networks.