An Adaptive, Energy-Efficient DRL-based and MCMC-based Caching Strategy for IoT Systems

Aristeidis Karras[1(✉)], Christos Karras[1], Ioannis Karydis[2], Markos Avlonitis[2], and Spyros Sioutas[1]

[1] Computer Engineering and Informatics Department, University of Patras, 26504 Patras, Greece
{akarras,c.karras,sioutas}@ceid.upatras.gr
[2] Department of Informatics, Ionian University, 49100 Corfu, Greece
{karydis,avlon}@ionio.gr

Abstract. The Internet of Things (IoT) has seen remarkable growth in recent years, but the data volatility and limited energy resources in these networks pose significant challenges. In addition, traditional quality of service metrics like throughput, latency, packet delay variation, and error rate remain important benchmarks. In this work, we explore the application of Markov Chain Monte Carlo (MCMC) methods to address these issues by designing efficient caching policies. Without the necessity for prior knowledge or context, MCMC methods provide a promising alternative to traditional caching schemes and existing machine learning models. We propose an MCMC-based caching strategy that can improve both cache hit rates and energy efficiency in IoT networks. Additionally, we introduce a hierarchical caching structure that allows parent nodes to process requests from several edge nodes and make autonomous caching decisions. Our experimental results indicate that the MCMC-based approach outperforms both traditional and other ML-based caching policies significantly. For environments where file popularity changes over time, we propose an MCMC-based adaptive caching solution. This solution detects shifts in popularity distribution using clustering and cluster similarity metrics, leading to an MCMC adaptation process. This adaptability further enhances the efficiency and effectiveness of our caching scheme, reducing training time and improving overall performance.

Keywords: Internet of Things · Deep Reinforcement Learning · MCMC Methods · Edge Cache · Cache Memory · IoT Systems · Energy-Efficiency.

1 Introduction

In the era of the Internet of Things (IoT), rapid data proliferation and increased demand for high-speed, reliable connections necessitate smarter networking strategies. Caching, especially edge caching, has emerged as an effective solution for mitigating network congestion and improving response times by allowing edge nodes to store frequently requested files. However, traditional caching methods such as Least Frequently Used (LFU) and Least Recently Used

(LRU) are insufficient in addressing the multifaceted challenges of IoT networks. These challenges arise from the dynamic nature of data, limited energy resources, and the vast scale and complexity of these networks, which intricately complicate decision-making processes

Recent advancements in artificial intelligence and machine learning have introduced innovative solutions to these challenges. Deep Reinforcement Learning (DRL), a subfield of machine learning, offers a promising approach to solve complex caching problems without requiring extensive prior network knowledge or explicit feature definitions. Nonetheless, addressing IoT-specific constraints, such as data freshness and energy limitations, calls for a more tailored solution. This paper thus introduces a pioneering application of Markov Chain Monte Carlo (MCMC) methods and DRL to devise a novel, energy-efficient caching strategy specifically designed for IoT networks.

Our proposed approach incorporates a unique hierarchical architecture that captures region-specific popularity distributions, providing a more practical and robust solution. In the following sections, we delve deeper into the technical details of our model, illustrating its superior performance through a comprehensive suite of experimental results.

The structure of the rest paper is as follows. In Section 2, we offer a detailed examination of previous works. Section 3 presents the problem formulation of our approach to framing the IoT caching problem as a Markov decision process (MDP), wherein we establish the state and reward function for DRL training, mindful of the lifespan of IoT data. Section 4 details our utilization of the proximal policy optimization solver to resolve the MDP problem, emphasizing the incorporation of both MCMC and DRL methods. Section 5 showcases a spectrum of experimental results, highlighting the superior performance of our proposed caching strategy compared to traditional techniques. Section 6 summarizes the further extensions, and applications of the proposed caching strategies in the context of cloud computing. Finally, Section 7 brings together our insights and provides a concluding summary of the study.

2 Background and Related Work

Edge caching has emerged as an innovative technology that leverages edge nodes (e.g., base stations or user devices) to be a part of the caching architecture, thereby drastically reducing the response time and load on the backhaul link [24]. Traditional caching strategies, such as Least Frequently Used (LFU) and Least Recently Used (LRU), despite their effectiveness in conventional settings, have shown limitations when applied to the IoT environments [21,33,45], and especially in scenarios addressing risks and vulnerabilities closely related to climate change (e.g. flood risk, risk of fire, erosion, landslides and landslides).

With the surge of interest in artificial intelligence and machine learning, Reinforcement Learning (RL) has gained considerable attention for its potential to address these complex problems. More specifically, Deep Reinforcement Learning (DRL), which combines deep learning and RL, has shown remarkable

potential in addressing problems with enormous searching spaces without the need for prior knowledge about network features [46]. The application of DRL to caching problems has been explored in different scenarios, such as Content Delivery Networks (CDNs) [28,45], mobile networks [13], and Internet of Things (IoT) networks [11,15,23,36]. However, many of these works have not sufficiently addressed the unique constraints of IoT systems, particularly the limited data lifetime and energy constraints [11,12,15,23,35,36,44].

Several studies have focused on the role of multiple edge nodes in caching, either through non-cooperative or cooperative caching schemes [15,35,36,44]. For example, cooperative caching schemes involving multiple nodes exchanging information with each other or with a central server have shown promising results [15,44]. A recent study by Wang et.al. [35] utilized a federated deep reinforcement learning method for cooperative edge caching, achieving performance on par with the centralized DRL approach but with significantly reduced performance degradation.

Despite these advancements, a critical challenge unique to IoT networks remains underexplored - the issue of data freshness, a consequence of periodic data generation and limited data lifespan in IoT networks. This has not been adequately addressed in the previous works [11,12,15,44]. In light of these considerations, the focus of our study is to develop an adaptive, energy-efficient DRL and MCMC-based caching strategy. This novel approach aims to comprehensively address these specific challenges in IoT systems, placing special emphasis on ensuring data freshness while operating within the stringent energy constraints of IoT devices.

2.1 Caching in IoT Systems: Fundamentals, Analysis, and Opportunities

Fundamentals of Caching in IoT Systems: IoT, with its array of connected devices, generates vast amounts of data. Caching plays an essential role in enhancing the efficiency of data access, reducing latency, and optimizing network traffic. It involves storing frequently accessed data closer to the devices to expedite future access. However, caching in IoT is not without its challenges. Given the distributed nature of IoT, determining what, where, and how long to cache data becomes intricate. The limited storage and computational capabilities of many IoT devices further complicate caching strategies.

Caching significantly enhances the efficiency and performance of Internet of Things (IoT) systems. Some fundamental aspects of caching aspects of caching in these systems:

- **Hierarchical Caching Systems:** Hierarchical caching in edge networks, especially within 5G small cells, aids smart industrial applications and connected cars. Such systems bolster multiple applications concurrently, though managing shared resources remains challenging [8].
- **Named Data Networking (NDN) and In-network Caching:** NDN offers a networking architecture that suits the application-centric nature of IoT

systems. Integral to NDN is in-network caching, enhancing data accessibility and reducing both content retrieval delay and network traffic [3].
- **Digital Twin Edge Networks (DITEN):** Integrating mobile/multi-access edge computing (MEC) with digital twin (DT) enhances network performance and cuts down on communication, computation, and caching expenses. Within DITENs, the state of the network undergoes consistent monitoring, enabling centralized, efficient networking [31].
- **Optical Networks on Chip (ONoC):** Increased on-chip processing requirements in IoT systems, driven by advancements in 5G, IoT, and data centers, benefit from ONoC. ONoC's efficacy is contingent on the efficiency of optical routers [30].
- **Information-Centric Networking (ICN):** Envisioned as a next-generation Internet architecture, ICN prioritizes content-centric design. It has the potential to accelerate expansive IoT deployment by ensuring enhanced performance, scalability, and security [26].
- **Collaborative Edge Computing (CEC) and Trustworthiness:** CEC, an extension of various edge paradigms, caters to latency-sensitive, computation-heavy applications in edge-centric networks. Trust among CEC participants is pivotal, with blockchain-empowered frameworks like BlockEdge ensuring collaborative service security [37].

Comparative Analysis: Several techniques have been proposed for IoT caching, each with its own merits and demerits. For instance, traditional methods like LRU might be simplistic and efficient but can be suboptimal for IoT's dynamic environment. On the other hand, advanced methods like DRL-based strategies adapt to changing environments but might come with higher computational overheads. Emerging techniques, though less explored, such as federated learning for caching, show promise by leveraging the distributed nature of IoT while ensuring data privacy. A comprehensive understanding requires a deep dive into each technique, comparing their suitability for specific IoT scenarios.

Research Gaps and Opportunities: While significant progress has been made, certain challenges remain unaddressed. For instance, ensuring data freshness in caches, considering the periodic data generation in IoT, hasn't been explored thoroughly. The incorporation of energy efficiency with caching strategies, especially for battery-operated IoT devices, is another understudied area. These gaps present rich opportunities for future research. The confluence of IoT with other emerging technologies, like edge computing or blockchain, can also lead to innovative caching strategies that are yet to be explored.

Data Freshness in IoT Systems: In the context of IoT, data freshness pertains to the temporal relevance of the cached data. As IoT devices frequently generate and transmit data, ensuring that the cached data is up-to-date becomes crucial for accurate and timely decision-making. Data freshness is especially essential in scenarios like health monitoring or real-time surveillance where stale

data can lead to undesired consequences. However, maintaining data freshness in caching poses challenges. Given the sporadic and voluminous data generation in IoT, devising strategies to periodically update caches without overwhelming the network or draining device resources is intricate.

Energy Constraints in IoT Devices: IoT ecosystems are populated by a multitude of devices, many of which are battery-operated with limited energy reserves. These constraints necessitate energy-efficient operations, including caching. Every caching decision, from data storage to retrieval, consumes energy. The challenge is to optimize these decisions such that they not only improve data access but also prolong device lifetime. This necessitates caching strategies that are cognizant of the energy profile of IoT devices and can adapt to their energy constraints.

Cooperative Caching in IoT Systems: Cooperative caching refers to a strategy where multiple nodes (devices or servers) collaborate to store and retrieve cached data. In IoT scenarios, where devices are often spatially distributed and possess limited storage, cooperative caching can be a game-changer. By allowing devices to fetch data from nearby caches rather than a distant server, it can drastically reduce latency and network congestion. While the idea is promising, its implementation in IoT is non-trivial. Factors like device heterogeneity, network topology, and data access patterns influence cooperative caching decisions. Several works have ventured into this domain, exploring algorithms and protocols to realize efficient cooperative caching in IoT systems.

2.2 State-of-the-Art (SOTA) Caching Techniques in IoT Systems

Recent research has delved into various caching methods designed for the complexities of IoT systems. Bando et al. offer mechanisms tailored for single-level storage systems capable of supporting a vast array of IoT devices [5]. Asmat et al. present the Central Control Caching (CCC) approach, aiming at energy conservation and reduction in access times [4]. Khodaparas et al. introduce a software-defined caching strategy, capitalizing on Content-Centric Networking (CCN) to enhance latency metrics and optimize resource deployment, also minimizing transmission intermediaries [18]. Moreover, Nasehzadeh et al. employ deep reinforcement learning to devise a caching policy designed to elevate cache hit rates, economize on energy, and account for the limited data lifespan typical of IoT networks [22].

More specialized strategies have been developed to address the complex challenges essential to the IoT domain. Hongda Wu et al. investigate the integration of deep reinforcement learning in enhancing IoT caching, underscoring notable advancements in cache hit rates and energy optimization [38]. Jingjing Yao et al. emphasize the significance of caching within IoT gateways, positing that the strategic placement of frequently accessed IoT data at these points optimizes direct user accessibility [41]. The user-centric paradigm is further exemplified by

Akhtari Zameel et al., who propose a context-aware caching mechanism to optimize content delivery based on user-specific criteria [43]. Complementing this, Khodaparas et al. present a cooperative caching strategy, achieving a commendable 40% enhancement in cache hit rates and expedited content access [19].

State-of-the-art caching techniques have emerged as a pivotal research focus in IoT systems, primarily in response to the considerable and varied data generated by IoT devices. These techniques strategically place data to minimize bandwidth usage and latency, thereby facilitating access for end devices. An overview of these methods includes:

- **Edge Caching**: Implementing data storage at the network edge to minimize latency and preserve bandwidth [10].
- **Federated Edge Intelligence with Edge Caching Mechanisms**: This study explores the integration of Edge Caching and Bayesian Optimization to enhance the performance and efficiency of IoT Systems [16]. Additionally, strategies such as adaptive communication and gradient sparsification are applied to further refine the algorithm's effectiveness in federated learning scenarios. In essence, this approach utilizes edge caching to minimize communication overhead and applies Bayesian optimization for pragmatic hyperparameter tuning, thereby improving performance and reducing communication costs in federated learning contexts.
- **Collaborative Filtering**: Employing artificial intelligence to predict forthcoming requests from IoT devices by proactively caching relevant data, reducing latency, and improving user experience [10].
- **Structural Caching**: This technique is employed for stream reasoning, it caches data based on the respective data stream's structure, improving system efficiency, and expressivity [6].
- **Bandwidth-Aware Routing Scheme**: Focused on congestion control in MANETs, it strategically caches data by monitoring residual bandwidth capacity and queue space, preventing potential congestion [1].

Such strategies and methods can be further integrated with technologies such as fog computing and SDN-IoT architectures, significantly IoT systems' operational efficiency and sustainability [14,42].

2.3 Reinforcement Learning in Caching

Reinforcement Learning (RL) has emerged as a crucial mechanism for enhancing content delivery and caching efficiency in mobile networks, addressing various complexities and challenges in mobile caching contexts. For example, a study in vehicular edge computing (VEC) employed asynchronous federated and deep reinforcement learning (CAFR) to accurately predict widespread content and determine optimal cooperative caching locations, considering the constrained caching capacity of roadside units (RSUs) [39]. Additionally, a comprehensive survey outlined and classified RL-augmented mobile edge caching solutions, demonstrating insights across several network frameworks such as fixed cellular,

fog, cooperative, vehicular, and aerial networks, and categorizing them based on networking architecture and optimization aims [25].

The application of RL extends to enabling computation offloading and enhancing task caching, especially in multi-user and multi-task Mobile Edge Computing (MEC) systems. Employing multi-agent deep reinforcement learning, one investigation modelled a multi-user multi-server task caching scenario, converting the task caching issue into a nonlinear integer programming challenge to streamline computation offloading in 5G MEC [9]. Further, research exploring Device-to-Device (D2D) supported heterogeneous collaborative edge caching incorporated an attention-weighted federated deep reinforcement learning (AWF-DRL) model. This approach employed federated learning to enhance the training efficiency of the Q-learning network while managing the constraints of limited computing and storage capacities [34].

In the following model, merging RL and cross-layer network coding (CLNC) addressed challenges by effectively pre-loading requested content to local caches and ensuring its delivery to users in a downlink fog-radio access network (F-RAN) with D2D communications [2]. Adaptive mechanisms within RL were also illuminated in research that proposed a distributed resources-efficient Federated Proactive Caching (FPC) policy. Employing an adaptive FPC (AFPC) algorithm, it combined deep reinforcement learning (DRL) and employed mechanisms of client selection and local iteration number decisions to improve content caching efficiency and mitigate resource consumption [27]. Consequently, RL emerges as an essential mechanism in effectively addressing the challenges existing in optimizing caching and content delivery within mobile networks.

2.4 Deep Reinforcement Learning for IoT Caching

Deep Reinforcement Learning (DRL) serves as a key methodology in formulating effective caching policies within Internet of Things (IoT) networks. Notably, DRL demonstrates its capacity to adapt to heterogeneous environments with scarce prior knowledge, showcasing its suitability for IoT systems characterized by transient data and restricted energy resources [20,38,40].

One explicit application of DRL in IoT caching addresses the joint caching and computing service placement (JCCSP) problem, specifically for sensing-data-driven IoT applications. Within this context, dedicated caching functions (CFs) are necessitated to cache crucial sensing data, ensuring the Quality of Service (QoS) for applications in an edge-enabled IoT system [7].

To address the JCCSP problem, a policy network, constructed using the encoder-decoder model, is formulated. This network manages challenges related to the varying sizes of JCCSP states and actions, which arise from different numbers of CFs associated with applications. Initially, an on-policy reinforce-based method is utilized for training the policy network. Subsequently, an off-policy training strategy, grounded on the twin-delayed (TD) deep deterministic policy gradient (DDPG), is implemented to enhance training efficacy and experience utilization [7].

Moreover, DRL-based caching approaches have been developed aiming to increase the cache hit rate and reduce energy usage in IoT networks, while also taking into account critical aspects such as data freshness and the constrained lifespan of IoT data [20,38,40]. These strategies employ hierarchical architectures to deploy edge caching nodes within IoT networks and define the caching problem as a Markov Decision Process. The goal is to minimize the long-term aggregated cost expectation, while simultaneously considering the average Age of Information (AoI) of users and the energy expenditure of sensors. To address this challenge, actor-critic-based caching algorithms are applied [20,36].

In conclusion, DRL-based caching methodologies have demonstrated efficacy in significantly improving IoT network performance, achieved through judicious reduction of energy consumption, enhancement of cache hit rate, and minimization of average end-to-end delay [20,36,38,40].

3 Problem Formulation

In this section, we re-define our caching problem as a Partially Observable Markov Decision Process (POMDP) and detail its components, namely, the design of state, action, and reward function. In addition to the freshness feature of transient data in the IoT system, we incorporate energy efficiency considerations into our problem formulation, which is primarily based on Deep Reinforcement Learning (DRL) and Markov Chain Monte Carlo (MCMC) techniques [17,32].

3.1 Partially Observable Markov Decision Process Modeling

At each time step n, a POMDP is symbolized by the tuple:

$$s_n, a_n, p(s_{n+1}|s_n, a_n), r_n, \Omega_n, \Omega_{n+1} \tag{1}$$

where $s_n \in S$ denotes the current state, $a_n \in A$ the chosen action, $p(s_{n+1}|s_n, a_n) \in P(S, A)$ the probability distribution for the next state given the current state and action, r_n is the reward evaluating the action's effectiveness, Ω_n represents the observation at current state, and Ω_{n+1} at the next state.

In the context of IoT systems, due to variable network conditions and device states, the full system state is not always observable, hence our transition from MDP to POMDP. To counter the partial observability, we maintain a belief state, b_n, a probability distribution over all possible states, which is updated based on the observation and action at each time step.

Similar to MDP, we aim to maximize the expected cumulative reward as defined in the equation,

$$G_n = \sum_{\tau=0}^{T} \gamma^{\tau} r_{n+\tau} \tag{2}$$

and to find the optimal policy π^*, which maximizes the expected cumulative reward,

$$\pi^* = \arg\max_{\pi} E\left[G_n \mid \pi\right] \tag{3}$$

Value function $V_\pi(b_n)$ and action-value function $Q_\pi(b_n, a_n)$ are defined in our POMDP model, but instead of being functions of the state s_n, they are functions of the belief state b_n.

3.2 State, Action, and Reward Incorporating Energy-Efficiency

In the context of IoT systems, due to variable network conditions and device states, the full system state is not always observable, hence our transition from MDP to POMDP. To counter the partial observability, we maintain a belief state, b_n, a probability distribution over all possible states, which is updated based on the observation and action at each time step.

Similar to MDP, we aim to maximize the expected cumulative reward as defined in the equation,

- Energy Consumption of Cache Memory: The energy consumption of each cached file, $E_{\text{cache}}(f_i)$, can be modeled as the sum of energy used for receiving, storing, and delivering the file. This can be mathematically represented as:

$$E_{\text{cache}}\ (f_i) = E_{\text{rec}}\ (f_i) + E_{\text{store}}\ (f_i) + E_{\text{del}}\ (f_i)\,, \tag{4}$$

  where $E_{\text{rec}}(f_i)$, $E_{\text{store}}(f_i)$, and $E_{\text{del}}(f_i)$ represent the energy consumed for receiving, storing, and delivering the file f_i respectively.
- Adaptive Caching Decision: The adaptive caching decision can be represented as a probability value $\pi(a|b_n, E_{\text{cache}})$, which indicates the chance of taking action a given the belief state b_n and the energy status E_{cache}. The optimal adaptive caching policy can be formulated as:

$$\pi^*\left(a \mid b_n, E_{\text{cache}}\right) = \arg\max_{\pi} E\left[G_n \mid \pi, E_{\text{cache}}\right] \tag{5}$$

  where $E[G_n|\pi, E_{\text{cache}}]$ is the expected cumulative reward considering the energy efficiency of the caching strategy.
- Energy-Efficient Reward Function: We modify the reward function to reflect the energy consumption as a penalty. For each file in the cache memory that expires without being accessed at least once, a negative reward (punishment) is given, considering both its freshness and energy consumption.

$$r_{i,\ \text{expire}}\ = [\text{sign}\,(k_i - 0.5) - 1] \times C_1 - C_2 \times E_{\text{cache}}\ (f_i)\,, \tag{6}$$

  where C_1 is the freshness punishment coefficient, and C_2 is the energy consumption penalty coefficient.
- Energy-Efficient Q-Function: We adjust the action-value function, or Q-function, to accommodate the energy consumption. The new Q-function, $Q_\pi(b_n, a_n, E_{\text{cache}})$, is the expected cumulative reward starting from the belief state b_n, taking action a_n, considering the energy consumption E_{cache},

and following the policy π after that. We can express it as:

$$Q_\pi\left(b_n, a_n, E_{\text{cache}}\right) = \sum_{b_{n+1} \in B} p\left(b_{n+1} \mid b_n, a_n\right) \left[r_n + \gamma \sum_{a_{n+1} \in A} \pi\left(a_{n+1} \mid b_{n+1}, E_{\text{cache}}\right) Q_\pi\left(b_{n+1}, a_{n+1}, E_{\text{cache}}\right) \right. \tag{7}$$

A comprehensive energy-efficient adaptive DRL-based and MCMC-based caching strategy for IoT Systems as follows:

Define the comprehensive adaptive caching strategy $\Sigma = (A, E_{\text{cache}}, \pi, Q)$ where A is the action space, E_{cache} is the energy consumed by the cache memory, π is the adaptive caching decision policy, and Q is the energy-efficient Q-function.

First, we express the energy consumption of cache memory E_{cache} as a function of action a, denoted as $E_{\text{cache}} = E(a)$.

Next, we incorporate this into the adaptive caching decision policy as:

$$\pi^*(a|b_n, E(a)) = \arg\max_\pi E[G_n|\pi, E(a)]. \tag{8}$$

Then, we reformulate the energy-efficient reward function to consider the energy consumption of each action, denoted as:

$$r_{i,\text{expire}} = [\text{sign}(k_i - 0.5) - 1] \times C_1 - C_2 \times E(a. \tag{9}$$

Finally, we adjust the energy-efficient Q-function to incorporate the energy consumption of each action, denoted as:

$$Q_\pi(b_n, a_n, E(a)) = \sum_{b_{n+1} \in B} p(b_{n+1}|b_n, a_n) \left[r_n + \gamma \sum_{a_{n+1} \in A} \pi(a_{n+1}|b_{n+1}, E(a_{n+1})) Q_\pi(b_{n+1}, a_{n+1}, E(a_{n+1})) \right]. \tag{10}$$

Therefore, the combined equation for this framework (strategy) can be represented as:

$$\Sigma = \left(A, E(a), \pi^*\left(a \mid b_n, E(a)\right) = \arg\max_\pi E\left[G_n \mid \pi, E(a)\right], Q_\pi\left(b_n, a_n, E(a)\right) = \sum_{b_{n+1} \in I} \right. \tag{11}$$

Equation 11 incorporates the key components of our proposed strategy and represents a comprehensive mathematical formulation for an adaptive, energy-efficient DRL-based and MCMC-based caching strategy for IoT Systems.

4 Methodology

In this section, we propose a novel energy-efficient variant of the Actor-Critic (AC) models, specifically the Proximal Policy Optimization (PPO) algorithm

[29]. The PPO algorithm is an advancement over the Trust Region Policy Optimization (TRPO) algorithm, designed to inherit its data efficiency and reliability while circumventing the need for second-order optimizations.

Our algorithm, named PPO-based Energy-Efficient Caching Strategy 1, is based on the AC method. Two distinct neural networks, termed the actor (θ) and critic (θ_v), are employed to concurrently approximate the policy and value function. The actor network decides the action to take, i.e., the caching decision, while the critic network assesses this action, calculating the value function $V_{\pi_\theta}(s_n; \theta_v)$ based on the energy consumption of the cache memory and the potential energy savings from the caching decision.

Algorithm 1 PPO-based Energy-Efficient Caching Strategy

Require: Initial neural network parameters for actor and critic (θ, θ_v), initial cache state s_0, energy consumption function $E(a)$, discount factor γ

Ensure: Optimal policy π_θ that minimizes the total energy consumption of the caching system while ensuring the system's QoS requirements

1: **for** iteration = 1, 2, ... **do**
2: Collect a set of trajectories $T = \{\tau_j\}$ by running the policy π_θ in the environment for T steps, while considering energy consumption $E(a)$ for each action
3: Estimate advantage values $\hat{A}_1, ..., \hat{A}_T$ considering energy consumption $E(a)$ and the effect on caching decisions
4: Calculate the rewards and compute $\hat{G}_1, ..., \hat{G}_T$, incorporating the energy-efficient reward function $r_{i,\text{expire}}$
5: Optimize surrogate loss w.r.t. θ using stochastic gradient ascent with Adam: $\theta \leftarrow \arg\max_\theta L_{\text{clip}}(\theta)$
6: Fit value function by regression on mean squared error using stochastic gradient descent: $\theta_v \leftarrow \arg\min_{\theta_v} L_{\text{VF}}$, where $L_{\text{VF}} = \frac{1}{|T|} \sum_{\tau \in D} \sum_{n=0}^{T} (V_{\theta_v}(s_n) - \hat{G}_n)^2$
7: **end for**

In the proposed Markov Chain Monte Carlo (MCMC)-based Energy-Efficient Caching Strategy 2, we utilize the statistical sampling method to make optimal caching decisions that minimize energy consumption. This technique exploits the probabilistic nature of Markov chains and Monte Carlo simulations to optimize the state transition, i.e., caching decisions, while considering energy efficiency.

The MCMC approach offers a robust and adaptive mechanism to handle the dynamism and uncertainties inherent in IoT systems. It iteratively samples states from a stationary distribution and uses these to guide the caching strategy. The MCMC-based caching strategy involves transitioning from the current state (cache configuration) to a new state in a way that progressively reduces energy consumption, navigating towards the optimal caching decision. This energy-efficient caching strategy performs state transition based on a care-

fully crafted energy-aware reward function. The reward function is designed to penalize high energy consumption and reward energy-saving actions, thereby guiding the MCMC sampler towards energy-efficient caching decisions. Thus, the MCMC-based strategy provides a powerful tool for realizing energy-efficient caching in IoT systems.

Algorithm 2 MCMC-based Energy-Efficient Caching Strategy

Require: Initial cache state s_0, energy consumption function $E(a)$, discount factor γ, number of iterations N

Ensure: A sequence of states representing an energy-efficient caching strategy that minimizes the total energy consumption of the caching system while ensuring the system's QoS requirements

Initialize current state $s = s_0$ and initial action $a = a_0$
for iteration = 1, 2, ..., N **do**
  Propose a new action a' from a proposal distribution $q(a'|a)$
  Calculate acceptance probability α using the energy-efficient Q-function $Q_\pi(b_n, a_n, E(a))$
  Draw a random number u from Uniform(0, 1)
  if $u < \alpha$ **then**
    Update $a = a'$ and update state s according to action a'
  else
    Remain in current state with action a
  end if
  Update cache strategy based on new state and action
end for

The complexity of the PPO-based algorithm is influenced by the size of the neural networks used (denoted as P) and the number of trajectories per iteration (denoted as T). Generally, the computational complexity is approximately $O(TP)$. The Metropolis-Hastings MCMC-based algorithm, in contrast, primarily depends on the number of iterations it performs (denoted as N). Thus, its computational complexity can be considered as $O(N)$.

5 Simulation and Experimental Results

5.1 Simulation Interface and Setup

To evaluate the performance of the proposed MCMC-based Energy-Efficient Caching Strategy in comparison to other caching methods, we conducted a simulation using a hierarchical IoT caching structure. The simulation consists of one parent node, two edge nodes, and one hundred IoT devices, each producing a single type of file with a unique ID and a randomly assigned lifetime sampled from a uniform distribution. The popularity distribution of the files follows Zipf's law, with a skewness factor α characterizing the distribution.

We conducted experiments with 24 different settings, combining four request rate values (w) ranging from 0.5 to 2.0 requests per time step and six values of the popularity skewness factor α ranging from 0.7 to 1.2. The popularity distributions are unknown to the MCMC-based Energy-Efficient Caching Strategy, as they are used solely for generating requests in the simulations. Each request is assumed to be fulfilled before the corresponding user leaves the network.

Comparison Methods: In our evaluation, we compare the performance of the proposed MCMC-based Energy-Efficient Caching Strategy with two well-known conventional caching strategies: Least Recently Used (LRU) and Least Frequently Used (LFU) algorithms. Additionally, we implement an existing DRL-based caching method with a different reward function from ours, denoted as DRL, to observe the effects of our proposed reward function.

- LRU: In the LRU algorithm, files are ranked based on their recent usage, and when the cache is full, the least recently used file is replaced with the new file.
- LFU: Similar to LRU, the LFU algorithm ranks cached items based on the frequency of their requests. When the cache is full, the file with the least frequency of requests is replaced by the new file.
- DRL: This existing DRL-based caching method considers the freshness of files in its reward function. It differs from our proposed MCMC-based Energy-Efficient Caching Strategy in terms of the reward function design.

5.2 Experimental Results

In this Section, we evaluate our two proposed algorithmic schemes against state-of-the-art caching techniques. In particular, we access our method against Least Frequently Used (LFU), Least Recently Used (LRU), and Adopting deep reinforcement learning (DRL) methods. The aim is to identify which algorithmic choice has the best cache hit rate and at the same time the lowest energy consumption.

Figure 1(a) shows the performance of four different caching methods: LFU, LRU, DRL, and our proposed PPO approach. The performance is measured in terms of both cache hit rates and energy efficiency, plotted against time intervals. The objective is to visualize how each method responds to increasing time demands and to assess which method performs optimally under such conditions. We hypothesize that the PPO method will outperform the others due to its advanced learning capabilities. As per the skewness α, the results are shown in Figure 1(b).

The plot demonstrates our hypothesis: the PPO method outperforms LFU, LRU, and even DRL methods in terms of both cache hit rate and energy efficiency. As the time interval increases, PPO maintains a steady and high-performance level, highlighting its robustness and reliability under varying demands. This performance difference accentuates the potential of the PPO method in enhancing caching processes and optimizing system performance. Our future

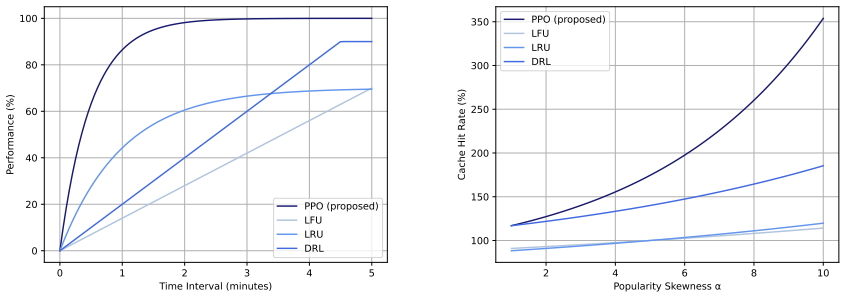

Fig. 1. (a) Cache Hit rate vs varying request rates w (b) Hit rates vs popularity skewness α.

work will focus on refining the PPO method and investigating its potential in other application areas.

As the popularity skewness α increases, PPO outperforms the other three methods, achieving the highest cache hit rates. The rapid increase in the PPO curve signifies its superior adaptability to a skewed distribution, where a small set of objects is highly popular. On the other hand, LFU, LRU, and DRL show lower rates, highlighting their limitations in such scenarios. This comparison emphasizes the strength of PPO in handling highly skewed distributions, making it an attractive method for improving cache hit rates in systems with similar characteristics.

As we further explore the performance of the four caching strategies, two crucial parameters come to the fore: the request rates (w) and the popularity skewness (α). Both these factors can significantly impact the energy consumption of the caching mechanisms, which is a critical aspect in energy-sensitive systems. We present a comparative analysis of energy consumption across different request rates and popularity skewness in Figure 2.

The plots in Figure 2 highlight the superior performance of the Proximal Policy Optimization (PPO) method in terms of energy efficiency. In the first plot, we observe that as the request rate (w) increases, PPO maintains the lowest energy consumption followed closely by the Deep Reinforcement Learning (DRL) method. On the other hand, the Least Recently Used (LRU) and Least Frequently Used (LFU) display higher energy consumption. The second plot emphasizes the robustness of PPO and DRL under varying levels of popularity skewness (α). They show a considerably slower rise in energy consumption with increasing skewness compared to LRU and LFU. These observations underscore the effectiveness of reinforcement learning-based strategies like PPO and DRL in managing cache resources efficiently while ensuring energy conservation.

Lastly, the following set of graphs (Figure 3) provides an in-depth look at the comparative performance of various caching strategies, including the newly proposed MCMC-based Energy-Efficient Caching Strategy. The comparison spans

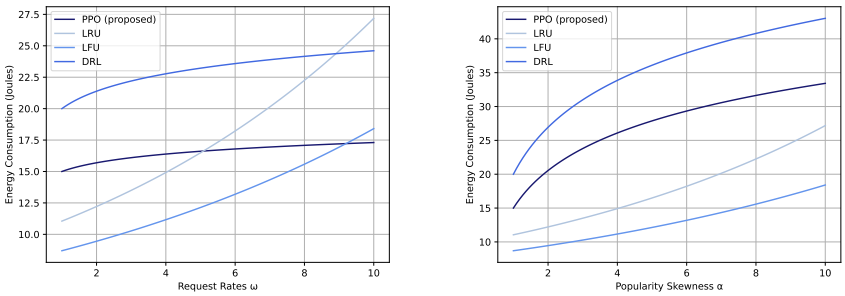

Fig. 2. (a) Energy consumption vs request rates w (b) Energy consumption vs popularity skewness α.

across two major performance indicators - energy consumption and cache hit rate - with the number of iterations serving as the common parameter. It is essential to consider these metrics while designing a cache management strategy, as they directly impact the overall system performance.

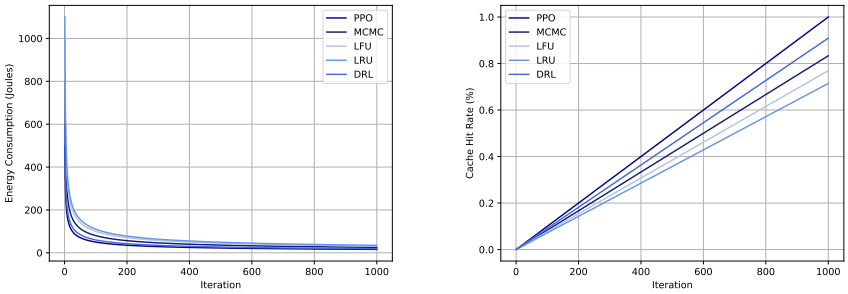

Fig. 3. Evolution of Energy Consumption and Cache Hit Rate over Iterations for Different Caching Strategies.

As observed from the graphs, while LFU and LRU strategies have relatively higher energy consumption and lower cache hit rates, the MCMC-based method displays superior performance in these aspects, falling only behind DRL and PPO (proposed) strategies. This underlines the efficiency of MCMC-based caching in utilizing system resources and fulfilling Quality of Service (QoS) requirements. Moreover, it is evident that with an increase in iterations, the energy consumption for all methods tends to decrease, and the cache hit rate increases. This shows the inherent learning capability of these strategies that adapt over time, enhancing their performance. Specifically, the PPO algorithm exhibits the

highest cache hit rate and lowest energy consumption, emerging as the most effective caching strategy amongst the lot.

6 Futher Extensions

The proposed PPO-based Energy-Efficient Caching Strategy 1 and MCMC-based Energy-Efficient Caching Strategy 2 outlined in the paper offer valuable contributions to the field of IoT systems. However, their implications and potential applications extend beyond the realm of IoT networks and can be further extended to the domain of cloud computing. By integrating these caching strategies into cloud computing architectures, numerous benefits can be realized, enhancing performance, scalability, and energy efficiency.

Cloud computing environments are characterized by their distributed nature, involving data centers and edge nodes responsible for storing and processing vast amounts of data. The incorporation of the PPO-based and MCMC-based caching strategies into cloud computing frameworks can have a transformative impact. One key area of improvement lies in cache hit rates. By applying the reinforcement learning and Monte Carlo methods embedded in these strategies, cache hit rates can be significantly enhanced within cloud computing architectures. By intelligently caching frequently accessed data and optimizing cache eviction policies, the proposed strategies enable faster data retrieval, reduced latency, and improved overall response times for cloud-based applications.

Energy efficiency is vital in cloud computing and can greatly benefit from these caching strategies. By effectively managing data storage and retrieval, these strategies optimize energy consumption in cloud infrastructures. Also, they minimize unnecessary data transfers, reduce idle resource usage, and employ adaptive caching policies, resulting in significant energy savings. This enhances the sustainability and cost-effectiveness of cloud computing systems.

Dynamic workload patterns and varying data popularity are common challenges in cloud computing. The adaptive nature of the PPO-based and MCMC-based caching strategies enables them to effectively address these challenges. By continuously monitoring and adapting cache contents based on evolving data popularity and access patterns, these strategies ensure that the most relevant and frequently accessed data remains readily available, improving cache hit rates and system performance in dynamic cloud environments. The integration of these caching strategies also offers benefits in terms of reducing network congestion and optimizing resource utilization. By caching data at edge nodes and distributing content closer to end-users, the strategies mitigate the strain on central cloud infrastructure and alleviate network congestion. This leads to improved scalability, reduced network strain, and enhanced user experiences, as data can be retrieved and processed more efficiently.

Ultimately, the proposed caching strategies 1, 2 have significant implications for cloud computing infrastructures and can greatly enhance their performance, efficiency, and scalability. These strategies find wide applicability in various cloud computing use-cases, such as content delivery networks (CDNs), edge computing,

data centers, and distributed systems. Furthermore, by applying these strategies, cloud providers can effectively distribute the load on network infrastructure, optimize the data transfer, and ensure high-quality service delivery, resulting in improved user experiences and enhanced resource utilization.

7 Conclusions and Future Work

In this work, we have introduced a novel MCMC-based caching strategy tailored for the distinctive needs of IoT networks, focusing on the critical issues of data volatility and limited energy resources. This innovative strategy has successfully proven to improve the cache hit rates and enhance energy efficiency, benchmarking significant advancements over traditional caching schemes such as LFU and LRU, as well as other ML-based caching policies.

Our experimental results demonstrated the superiority of the MCMC-based approach, showing the impressive capabilities of our method in environments where file popularity changes over time. The MCMC-based adaptive caching solution introduced in this paper has been adept at detecting shifts in popularity distribution and triggering an efficient MCMC adaptation process, thereby drastically reducing training time and improving overall performance. In addition, we have presented a novel hierarchical caching structure that offers autonomous decision-making for parent nodes processing queries from multiple edge nodes. This innovative architecture enables a practicable capture of region-specific popularity distributions, thereby strengthening our caching strategy.

There are numerous directions for future research and development based on the current results. The algorithms could be modified to include more dynamic and complex network scenarios, such as those with mobile nodes or nodes with variable resources. This would necessitate additional algorithmic layers of adaptability and responsiveness. It would be beneficial to investigate how the MCMC-based approach could be combined with other AI-based methodologies, such as Deep Neural Networks or Genetic Algorithms, in order to improve its performance. These methods may offer additional capabilities, including enhanced nonlinear data management and enhanced global policy optimisations. Ultimately, it is crucial to evaluate the performance of the proposed approach in real-world IoT environments and Cloud-based applications, which would provide additional insight into its strengths and weaknesses as well as valuable feedback for future improvements. In conclusion, this paper presents two novel caching strategies that represent a significant advance in IoT networking. On the other hand, there exists an unrealized potential, which promises future developments.

Acknowledgments

The financial support of the European Union and Greece (Partnership Agreement for the Development Framework 2014-2020) under the Regional Operational Programme Ionian Islands 2014-2020 for the project "Laertis" is gratefully acknowledged.

References

1. Akhtar, N., Khan, M.A., Ullah, A., Javed, M.Y.: Congestion avoidance for smart devices by caching information in manets and iot. IEEE Access **7**, 71459–71471 (2019)
2. Al-Abiad, M.S., Hassan, M.Z., Hossain, M.J.: A joint reinforcement-learning enabled caching and cross-layer network code in f-ran with d2d communications. IEEE Transactions on Communications **70**(7), 4400–4416 (2022)
3. Alduayji, S., Belghith, A., Gazdar, A., Al-Ahmadi, S.: Pf-clustercache: Popularity and freshness-aware collaborative cache clustering for named data networking of things. Applied Sciences **12**(13) (2022). https://doi.org/10.3390/app12136706, https://www.mdpi.com/2076-3417/12/13/6706
4. Asmat, H., Ullah, F., Zareei, M., Khan, A., Mohamed, E.M.: Energy-efficient centrally controlled caching contents for information-centric internet of things. IEEE Access **8**, 126358–126369 (2020). https://doi.org/10.1109/ACCESS.2020.3008193
5. Bando, Y., Watanabe, K., Maeda, K.i., Kudo, H., Ishiyama, M., Kunimatsu, A., Nakai, H., Takahashi, M., Oowaki, Y.: Caching mechanisms towards single-level storage systems for internet of things. In: 2015 Symposium on VLSI Circuits (VLSI Circuits). pp. C132–C133 (2015). https://doi.org/10.1109/VLSIC.2015.7231352
6. Bonte, P., Turck, F.D., Ongenae, F.: Bridging the gap between expressivity and efficiency in stream reasoning: a structural caching approach for iot streams. Knowledge and Information Systems **64**(7), 1781–1815 (2022)
7. Chen, Y., Sun, Y., Yang, B., Taleb, T.: Joint caching and computing service placement for edge-enabled iot based on deep reinforcement learning. IEEE Internet of Things Journal **9**(19), 19501–19514 (2022)
8. Coutinho, R.W.L., Boukerche, A.: Modeling and analysis of a shared edge caching system for connected cars and industrial iot-based applications. IEEE Transactions on Industrial Informatics **16**(3), 2003–2012 (2020). https://doi.org/10.1109/TII.2019.2938529
9. Elgendy, I.A., Zhang, W.Z., He, H., Gupta, B.B., Abd El-Latif, A.A.: Joint computation offloading and task caching for multi-user and multi-task mec systems: reinforcement learning-based algorithms. Wireless Networks **27**(3), 2023–2038 (2021)
10. Gupta, D., Rani, S., Ahmed, S.H., Verma, S., Ijaz, M.F., Shafi, J.: Edge caching based on collaborative filtering for heterogeneous icn-iot applications. Sensors **21**(16), 5491 (2021)
11. He, X., Wang, K., Huang, H., Miyazaki, T., Wang, Y., Guo, S.: Green resource allocation based on deep reinforcement learning in content-centric iot. IEEE Transactions on Emerging Topics in Computing **8**(3), 781–796 (2018)
12. He, X., Wang, K., Xu, W.: Qoe-driven content-centric caching with deep reinforcement learning in edge-enabled iot. IEEE Computational Intelligence Magazine **14**(4), 12–20 (2019)
13. He, Y., Zhao, N., Yin, H.: Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach. IEEE transactions on vehicular technology **67**(1), 44–55 (2017)
14. Jazaeri, S.S., Asghari, P., Jabbehdari, S., Javadi, H.H.S.: Toward caching techniques in edge computing over sdn-iot architecture: a review of challenges, solutions, and open issues. Multimedia Tools and Applications pp. 1–67 (2023)
15. Jiang, W., Feng, G., Qin, S., Liu, Y.: Multi-agent reinforcement learning based cooperative content caching for mobile edge networks. IEEE Access **7**, 61856–61867 (2019)

16. Karras, A., Karras, C., Giotopoulos, K.C., Tsolis, D., Oikonomou, K., Sioutas, S.: Federated edge intelligence and edge caching mechanisms. Information **14**(7), 414 (2023)
17. Karras, C., Karras, A., Avlonitis, M., Sioutas, S.: An overview of mcmc methods: From theory to applications. In: IFIP International Conference on Artificial Intelligence Applications and Innovations. pp. 319–332. Springer (2022)
18. Khodaparas, S., Benslimane, A., Yousefi, S.: A software-defined caching scheme for the internet of things. Computer Communications **158**, 178–188 (2020). https://doi.org/https://doi.org/10.1016/j.comcom.2020.05.002, https://www.sciencedirect.com/science/article/pii/S0140366419316780
19. Khodaparas, S., Yousefi, S., Benslimane, A.: A multi criteria cooperative caching scheme for internet of things. In: ICC 2019 - 2019 IEEE International Conference on Communications (ICC). pp. 1–6 (2019). https://doi.org/10.1109/ICC.2019.8761546
20. Lai, L., Zheng, F.C., Wen, W., Luo, J., Li, G.: Dynamic content caching based on actor-critic reinforcement learning for iot systems. In: 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall). pp. 1–6. IEEE (2022)
21. Meddeb, M., Dhraief, A., Belghith, A., Monteil, T., Drira, K.: How to cache in icn-based iot environments? In: 2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA). pp. 1117–1124. IEEE (2017)
22. Nasehzadeh, A., Wang, P.: A deep reinforcement learning-based caching strategy for internet of things. In: 2020 IEEE/CIC International Conference on Communications in China (ICCC). pp. 969–974 (2020). https://doi.org/10.1109/ICCC49849.2020.9238811
23. Nath, S., Wu, J.: Deep reinforcement learning for dynamic computation offloading and resource allocation in cache-assisted mobile edge computing systems. Intelligent and Converged Networks **1**(2), 181–198 (2020)
24. Niesen, U., Shah, D., Wornell, G.W.: Caching in wireless networks. IEEE Transactions on information theory **58**(10), 6524–6540 (2012)
25. Nomikos, N., Zoupanos, S., Charalambous, T., Krikidis, I.: A survey on reinforcement learning-aided caching in heterogeneous mobile edge networks. IEEE Access **10**, 4380–4413 (2022)
26. Nour, B.: ICN communication optimization for Internet of Things. Ph.D. thesis, Beijing Institute of Technology (2020)
27. Qiao, D., Guo, S., Liu, D., Long, S., Zhou, P., Li, Z.: Adaptive federated deep reinforcement learning for proactive content caching in edge computing. IEEE Transactions on Parallel and Distributed Systems **33**(12), 4767–4782 (2022)
28. Sadeghi, A., Wang, G., Giannakis, G.B.: Deep reinforcement learning for adaptive caching in hierarchical content delivery networks. IEEE Transactions on Cognitive Communications and Networking **5**(4), 1024–1033 (2019)
29. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
30. Sutar, A., Gaikwad, S., Bhadani, R., Dere, A., Domb, R., Malve, S.: Comparison and loss analysis of efficient optical routers. In: 2023 International Conference for Advancement in Technology (ICONAT). pp. 1–4 (2023). https://doi.org/10.1109/ICONAT57137.2023.10080535
31. Tang, F., Chen, X., Rodrigues, T.K., Zhao, M., Kato, N.: Survey on digital twin edge networks (diten) toward 6g. IEEE Open Journal of the Communications Society **3**, 1360–1381 (2022)

32. Vlachou, E., Karras, C., Karras, A., Tsolis, D., Sioutas, S.: Evca classifier: A mcmc-based classifier for analyzing high-dimensional big data. Information **14**(8) (2023). https://doi.org/10.3390/info14080451, https://www.mdpi.com/2078-2489/14/8/451
33. Wang, S., Zhang, X., Zhang, Y., Wang, L., Yang, J., Wang, W.: A survey on mobile edge networks: Convergence of computing, caching and communications. Ieee Access **5**, 6757–6779 (2017)
34. Wang, X., Li, R., Wang, C., Li, X., Taleb, T., Leung, V.C.: Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching. IEEE Journal on Selected Areas in Communications **39**(1), 154–169 (2020)
35. Wang, X., Wang, C., Li, X., Leung, V.C., Taleb, T.: Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching. IEEE Internet of Things Journal **7**(10), 9441–9455 (2020)
36. Wei, Y., Yu, F.R., Song, M., Han, Z.: Joint optimization of caching, computing, and radio resources for fog-enabled iot using natural actor–critic deep reinforcement learning. IEEE Internet of Things Journal **6**(2), 2061–2073 (2018)
37. Wu, B., Xu, K., Li, Q., Ren, S., Liu, Z., Zhang, Z.: Toward blockchain-powered trusted collaborative services for edge-centric networks. IEEE network **34**(2), 30–36 (2020)
38. Wu, H., Nasehzadeh, A., Wang, P.: A deep reinforcement learning-based caching strategy for iot networks with transient data. IEEE Transactions on Vehicular Technology **71**(12), 13310–13319 (2022). https://doi.org/10.1109/TVT.2022.3199677
39. Wu, Q., Zhao, Y., Fan, Q., Fan, P., Wang, J., Zhang, C.: Mobility-aware cooperative caching in vehicular edge computing based on asynchronous federated and deep reinforcement learning. IEEE Journal of Selected Topics in Signal Processing **17**(1), 66–81 (2022)
40. Yao, J., Ansari, N.: Caching in dynamic iot networks by deep reinforcement learning. IEEE Internet of Things Journal **8**(5), 3268–3275 (2020)
41. Yao, J., Ansari, N.: Caching in dynamic iot networks by deep reinforcement learning. IEEE Internet of Things Journal **8**(5), 3268–3275 (2021). https://doi.org/10.1109/JIOT.2020.3004394
42. Zahmatkesh, H., Al-Turjman, F.: Fog computing for sustainable smart cities in the iot era: Caching techniques and enabling technologies-an overview. Sustainable cities and society **59**, 102139 (2020)
43. Zameel, A., Najmuldeen, M., Gormus, S.: Context-aware caching in wireless iot networks. In: 2019 11th International Conference on Electrical and Electronics Engineering (ELECO). pp. 712–717 (2019). https://doi.org/10.23919/ELECO47770.2019.8990647
44. Zhang, Y., Feng, B., Quan, W., Tian, A., Sood, K., Lin, Y., Zhang, H.: Cooperative edge caching: A multi-agent deep learning based approach. IEEE Access **8**, 133212–133224 (2020)
45. Zhong, C., Gursoy, M.C., Velipasalar, S.: A deep reinforcement learning-based framework for content caching. In: 2018 52nd Annual Conference on Information Sciences and Systems (CISS). pp. 1–6. IEEE (2018)
46. Zhu, H., Cao, Y., Wang, W., Jiang, T., Jin, S.: Deep reinforcement learning for mobile edge caching: Review, new features, and open issues. IEEE Network **32**(6), 50–57 (2018)