# Ratio-Based Offloading Optimization for Edge and Vehicular-Fog Federated Systems: A Multi-Agent TD3 Approach

Frezer Guteta Wakgra, Widhi Yahya, Binayak Kar, *Member, IEEE*, Yuan-Cheng Lai, Ying-Dar Lin, *Fellow, IEEE*, and Seifu Birhanu Tadele

***Abstract*—Edge and vehicular-fog architectures enable a two-tier Multi-access Edge Computing (MEC) system, where computing resources are positioned behind base stations and vehicular-fog nodes are placed closer to user equipment (UE), resulting in lower latency compared to cloud alternatives. Vehicular-fog is a computing resource formed by a group of either stationary or mobile vehicles. In densely populated areas like congested intersections or festivals, UEs can generate concentrated hotspot traffic towards nearby MEC servers, potentially straining the access network MEC (AN-MEC) site behind the base station. To address this, a portion of the traffic can be offloaded, either vertically to nearby vehicular-fog nodes or horizontally to neighboring AN-MEC sites. The control plane rapidly determines traffic offloading locations and ratios within seconds, with the goal of minimizing average system latency. Our work proposes a reinforcement learning (RL)-based multi-agent TD3, which is built on top of the Twin-Delayed Deep Deterministic Policy Gradient (TD3) algorithm to determine optimal offloading ratios. Evaluation results underscore the remarkable decision-making speed of the multi-agent TD3-based approach, which surpasses the single-agent TD3 and simulated annealing (SA) methods by one and five orders of magnitude, respectively. Notably, the average latency of the multi-agent TD3 is better than that of the single-agent TD3, with only a marginal increase of 2 to 6 ms when compared to SA.**

***Index Terms*—Edge, offloading, optimization, RL, TD3, vehicular-fog.**

Manuscript received 18 October 2023; revised 25 April 2024; accepted 9 July 2024. Date of publication 22 July 2024; date of current version 7 November 2024. This work was supported by the National Science and Technology Council (NSTC), Taiwan, under Grant 112-2221-E-011-057 and Grant 111-2221-E-A49-095-MY3. The review of this article was coordinated by Dr. Kaigui Bian. *(Frezer Guteta Wakgra and Widhi Yahya contributed equally to this work.) (Corresponding author: Binayak Kar.)*

Frezer Guteta Wakgra, Binayak Kar, and Seifu Birhanu Tadele are with the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan (e-mail: haftiyam@gmail.com; bkar@mail.ntust.edu.tw; seife.brhan@gmail.com).

Widhi Yahya is with the Faculty of Computer Science, University of Brawijaya, Malang 65145, Indonesia, and also with the Department of Electrical Engineering and Computer Science, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan (e-mail: widhi.yahya@ub.ac.id).

Yuan-Cheng Lai is with the Department of Information Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan (e-mail: laiyc@cs.ntust.edu.tw).

Ying-Dar Lin is with the Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan (e-mail: ydlin@cs.nycu.edu.tw).

Digital Object Identifier 10.1109/TVT.2024.3431549

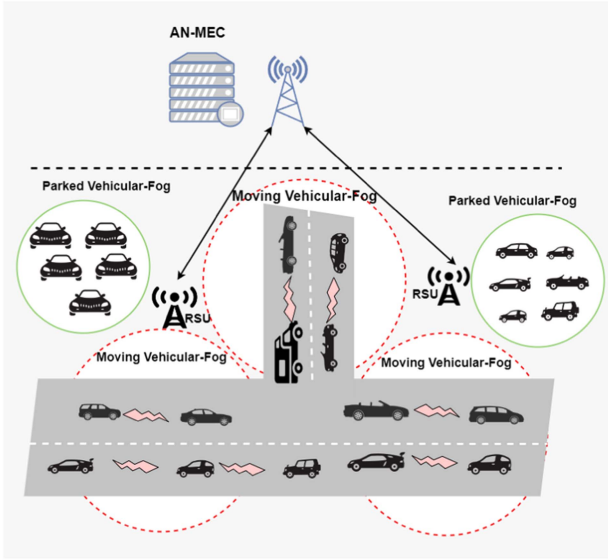



Fig. 1. Vehicular-fog formation in AN-MEC network.

## I. Introduction

In FIFTH-GENERATION (5G) networks, the computing infrastructure is seamlessly integrated within the communication framework, providing significant opportunities through high bandwidth and low latency.

The positioning of edge servers at the periphery of the cellular network is known as multi-access edge computing (MEC), as shown in Fig. 1. MEC constitutes a decentralized system with lower capacity compared to the cloud. Its proximity to the User Equipment (UE) is closer than that of the cloud, rendering it particularly suitable for services that demand low latency [1]. The European Telecommunications Standards Institute (ETSI) has outlined scenarios where the MEC server can be deployed at a base station, known as access network MEC (AN-MEC) [2]. This configuration illustrates a specific scenario where MEC servers are deployed at base stations to optimize and enhance the capabilities of edge computing within mobile networks. AN-MEC brings communication and computation services closer to end-users, enabling efficient handling of latency-sensitive applications and reducing the load on the core network.

An AN-MEC server may face challenges due to incoming user traffic originating from cloud servers, neighboring AN-MECs,

and UE devices. The UE devices, including smartphones, robots, and vehicles, often have tasks beyond their local processing capabilities, prompting them to offload some of these tasks to the AN-MEC for more efficient computation [3]. In densely populated areas like congested road intersections or music festivals, numerous UEs can create hotspot traffic directed towards nearby AN-MEC servers [4], [5]. When an AN-MEC server becomes overloaded, it can mitigate its workload by offloading tasks to neighboring nodes, fog resources, or even the cloud, thereby reducing computational latency caused by server congestion [6]. This dynamic offloading decision highlights the crucial role of the control plane in efficiently managing network resources. In the context of the vehicle-to-everything (V2X) use case for 5G, vehicles are equipped with sensors, computers, and communication devices. These vehicles provide a range of services when they are in use to enhance security, safety, and driving comfort. Additionally, vehicles can offload specific tasks that may be beyond their local processing capacity or engage in data exchange with other system entities, including vehicle-to-vehicle (V2V), and vehicle-to-infrastructure (V2I) [7].

A group of stationary vehicles within a parking lot or at an intersection holds the potential to provide a computing resource known as vehicular-fog [8], as shown in Fig. 1. This vehicular fog can be vertically federated with AN-MECs, and the AN-MEC itself can be horizontally federated with its neighboring AN-MECs. The federation systems mentioned in this study are different from the *federation learning*[1] in artificial intelligence (AI). The federation system represents a collaborative effort among various cloud providers. The collaboration aims to leverage the collective resources available across different cloud platforms to enhance the overall quality of service (QoS) for users. Through this federation system, the vehicular fog and AN-MECs can work together to efficiently complete computing tasks, allowing contributors to earn compensation for each successfully completed task [10].

The optimization of task offloading within edge and vehicular-fog environments, with the objective of minimizing the average system latency, involves determining where and how much to offload [11]. Traditional exhaustive search methods in optimization demand substantial time to converge towards the optimal offloading decision [12], [13]. Reinforcement learning (RL) emerges as a promising approach for offloading optimization, as it learns the optimal solution directly from the environment and can select sub-optimal solutions without waiting for the optimal one. Unlike supervised learning, which hinges on abundant labeled data, RL operates without such a requirement, allowing it to be trained using data batches. Obtaining labeled data proves exceedingly challenging in the context of a dynamic and distributed edge and vehicular-fog system [14], [15], [16], [17].

RL has been employed by the authors of [18], [19], [20], [21], [22], [23], [24] to address edge and vehicular-fog offloading optimization. Most of them consider vertical offloading from a vehicle to a roadside unit (RSU) or edge server, with the UE making those decisions. This work examines the offloading problem as a component of the AN-MEC control plane for handling incoming hotspot traffic at an AN-MEC site co-located with a base station. However, the AN-MEC site faces the challenge of potential overload due to its limited computational capacity, leading to significant computational delays that violate QoS requirements. In order to enhance the QoS by utilizing the available computational resources within the network topology, the AN-MEC control plane decides where to process the incoming traffic. The AN-MEC control plane determines whether to process the incoming traffic locally or offload some traffic horizontally to neighboring AN-MECs or vertically downward to connected vehicular-fog sites, with the objective of minimizing average system latency. We employ a multi-agent twin-delayed deep deterministic policy gradient (Multi-agent TD3) algorithm to determine these offloading decisions, which builds upon the twin-delayed deep deterministic policy gradient (TD3) framework.

The proposed method determines not only the offloading destination but also the ratio of packets that must be offloaded by an AN-MEC site within a continuous search space. These agents are deployed in the AN-MEC control plane to make ratio-based offloading decisions. To the best of our knowledge, this is the first RL-based offloading optimization to consider ratio-based traffic offloading. The ratio represents the destination and the number of traffic that a site must offload. The objective is to minimize the average system latency experienced by arriving hotspot traffic. In short, the main contributions of this work are summarized as follows.

1) We formulate an optimization model for ratio-based vertical and horizontal offloading within edge and vehicular-fog architecture, with the main objective of minimizing average system latency.
2) We transform the aforementioned problem into an equivalent RL environment and subsequently propose a multi-agent twin-delayed deep deterministic policy gradient (Multi-agent TD3) algorithm to solve this issue and make ratio-based offloading decisions.
3) We conduct comprehensive simulations to assess the effectiveness of the RL environment and the performance of the proposed approach, considering several factors including convergence time, decision time, decision quality, and offloading under unpredictable traffic rates. The results demonstrate that the proposed approach outperforms the benchmark solutions.

The remaining sections of this paper are organized as follows: Section II presents the related work. Section III describes the system modeling and problem formulation. Section IV introduces the proposed solution. Section V provides details about the parameter setting and result analysis. Finally, Section VI concludes this work.

## II. Related Works

Since machine learning (ML) can map any input, even with lacking data, it offers a promising solution for the offloading optimization problem in a vehicular-fog system. In this section,

[1]Federated Learning is a collaborative machine learning method where models are trained across decentralized devices, ensuring data privacy by keeping information local [9].

we discuss the earlier studies that employed ML for vehicular-fog offloading and present their agent and network models for offloading optimization.

Guo et al. [16] addressed the issue of offloading in ML based on edge computing. Centralized and decentralized learning are evaluated, and a model with the shortest training latency is constructed. Sonmez et al. [17] employed two stages of ML for orchestrating arrival tasks. In the first stage, supervised ML was used to classify which destination could handle the incoming tasks with a success/failure result, and in the second stage, regression was used to estimate the time required to process the incoming tasks. Both [16] and [17] utilized supervised ML, which requires well-labeled data to produce the desired results (optimum offloading action). In such a dynamic vehicular-fog system, labeling data is costly since labels must be continuously updated in accordance with incoming traffic and the availability of computing resources.

Shi et al. [18] maximized system utility by optimizing offloading and resource allocation in vehicular-fog networks. A vehicle with tasks may perform them locally or offload them to another vehicle with sufficient computing resources. In a base station equipped with a soft actor critic (SAC)-based offloading agent, estimations of available resources and offloading decisions were made. Pan et al. [19] optimize the offloading strategy of a user vehicle (UV) that can offload its task to vehicular-fog server or RSU. A deep Q-learning (DQN)-based strategy was proposed to find the best offloading decision that maximizes system throughput and satisfies ultra-reliable low latency communications (URLLC) traffic. The distributed DQN algorithm was proposed to solve the resource allocation and offloading issues. Khayyat et al. [20] formulated offloading and resource allocation as a binary optimization problem with the objective of minimizing latency and energy. The distributed DQN algorithm was proposed to solve the resource allocation and offloading issues. Lan et al. [21] optimized offloading between vehicle, RSU, and cloud, termed as peer-pool offloading, and offloading between RSUs, termed as fog-pool offloading, by using a DDPG-based algorithm. Xie et al. [24] employed the multi-armed bandit technique to determine the optimal vehicle offloading operation to minimize latency. The ask-carrying vehicle decides whether to offload a task to another vehicle through V2V or V2I. In order to determine the destination of offloaded tasks, the vehicle must actively communicate with the centralized monitoring system, which is an energy-intensive task.

The authors of [18], [22], [23] have addressed offloading optimization into as part of the network control plane problem. Maan and Chaba [22] employed DQN to optimize offloading between V2V and V2I in a Software-Defined Networking (SDN)-based vehicular-fog network. The Kalman filter was used to predict the movement of vehicles, and the prediction was utilized to allocate computing resources. Vemireddy and Rout [23] devised SARSA, an offloading approach based on an on-policy RL algorithm, to reduce RSU energy usage. RSU could execute an incoming task or offload it to vehicular-fog, introducing a cost-benefit trade-off between computing and communication. Shi et al. [18] addressed computation offloading optimization in vehicular-fog computing where a task-carrying vehicle could offload the task to another moving vehicle with a different cost. A soft actor-critic (SAC) algorithm was used to select the service-vehicle and to allocate the computing resources for the offloaded tasks. Zhang et al. [25] investigated computation offloading in the Internet of Vehicles (IoV), aiming to minimize a multi-objective optimization problem encompassing delay, energy consumption, computing load, and privacy entropy of vehicle terminals (VTs). They proposed an RBF-based multi-objective DDQN deep reinforcement learning computation offloading algorithm (RMDDQN-Learning), designed to concurrently optimize multiple objectives. Cui et al. [26] explore cooperative computational offloading among edge servers (ESs) within an IoV architecture. Their objective is to minimize task execution delays by optimizing cooperative offloading strategies. They propose an RL-based approach called an offline centralized training distributed execution framework based on soft actor-critic (OCTDE-ISAC).

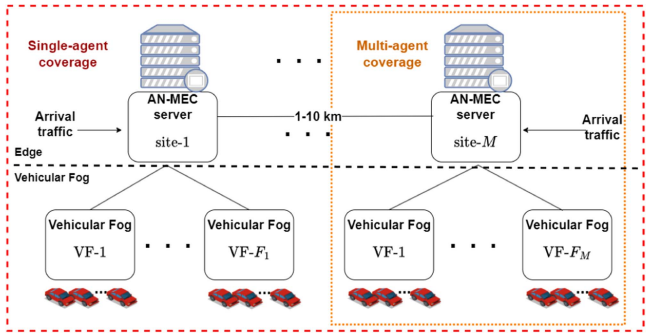



Fig. 2. Edge and vehicular-fog architecture.

The majority of the previous works merely decided the destination of the offloaded tasks, not the amount of traffic to be offloaded. However, Maan et al. [22] proposed ratio-based offloading with vehicle-to-fog-server offloading direction. In this paper, we focus on the offloading of an overloaded AN-MEC site that could offload certain incoming tasks horizontally to its neighbors or vertically downward to vehicular-fog. The Multi-agent TD3 algorithm is used to determine the optimal offloading ratio, which indicates where and how many tasks must be offloaded by an AN-MEC site.

## III. Edge and Vehicular-Fog: System Model and Problem Formulation

This section describes edge and vehicular-fog architecture, problem formulation, and latency models. Table I provides notations used for problem formulation and latency models.

### A. Edge and Vehicular-Fog Architecture With Ratio-Based Offloading

As shown in Fig. 2, an edge and vehicular-fog system comprises $\mathcal{M}$ AN-MEC sites, each connected to a cluster $F_j$ of vehicular-fog sites formed by parked vehicles equipped with computational resources. Both AN-MEC and vehicular-fog sites consist of multiple servers with a combined capacity denoted by $\mu_j^{\mathrm{A}}$ and $\mu_{j,k}^{\mathrm{F}}$, where $j$ and $k$ represent the AN-MEC and vehicular-fog indices, respectively. The AN-MEC sites are interlinked through an optical connection with substantial capacity

TABLE I
NOTATION DEFINITION FOR EDGE AND VEHICULAR-FOG ARCHITECTURE

| Category | Notations | Meaning | Attribute |
|---|---|---|---|
| Topology | $\mathcal{M}$ | Set of AN-MEC sites, $\mathcal{M} = 1, 2, 3, \ldots, M$ | Input |
| | $F_j$ | Set of vehicular-fogs at $j^{th}$ AN-MEC site | Input |
| Capacity | $\mu_j^{\mathrm{A}}$ | Capacity of $j^{th}$ AN-MEC site | Input |
| | $\mu_{j,k}^{\mathrm{F}}$ | Capacity of $k^{th}$ vehicular-fog of $j^{th}$ AN-MEC site | Input |
| | $B_{j,k}^{\mathrm{FA}}$ | Uplink bandwidth at $k^{th}$ vehicular-fog of $j^{th}$ AN-MEC site | Input |
| | $B_{j,k}^{\mathrm{AF}}$ | Downlink bandwidth at $k^{th}$ vehicular-fog of $j^{th}$ AN-MEC site | Input |
| Traffic | $\lambda_j$ | Arrival traffic rate of $j^{th}$ AN-MEC site | Input |
| | $\lambda_j^{\mathrm{A}}$ | Traffic that is served at $j^{th}$ AN-MEC site | Output |
| | $\lambda_{j,k}^{\mathrm{F}}$ | Traffic that is served at $k^{th}$ vehicular-fog of $j^{th}$ AN-MEC site | Output |
| | $\lambda_{j,k}^{\mathrm{AF}}$ | Traffic that is going through link between $k^{th}$ vehicular-fog and $j^{th}$ AN-MEC site | Output |
| Delay | $D_{j\leftrightarrow m}^{\mathrm{AA}}$ | Propagation latency between $j^{th}$ AN-MEC site and $m^{th}$ AN-MEC site | Input |
| | $l_j$ | Average latency of arrival traffic at $j^{th}$ AN-MEC site | Input |
| | $l$ | System average latency | Input |
| Offloading | $P_j^{\mathrm{A}}$ | Probability of arrival traffic being served at $j^{th}$ AN-MEC site | Output |
| | $P_{j\rightarrow m}^{\mathrm{AA}}$ | Probability of arrival traffic being offloaded to $m^{th}$ AN-MEC site | Output |
| | $P_{j,k}^{\mathrm{F}}$ | Probability of arrival traffic at $k^{th}$ vehicular-fog of $j^{th}$ AN-MEC site | Output |

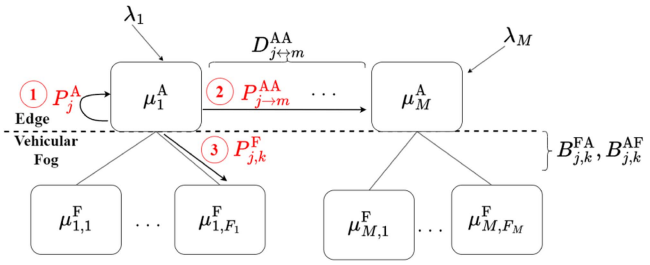



Fig. 3. Edge and vehicular-fog offloading scenarios.

and minimal propagation delay, denoted as $D_{j\leftrightarrow m}^{\mathrm{AA}}$. As a result of the capacity of the link between AN-MEC sites, its queuing latency is negligible. Communication between a vehicular-fog site and an AN-MEC site occurs via a wireless link, featuring uplink and downlink capacities designated as $B_{j,k}^{\mathrm{FA}}$ and $B_{j,k}^{\mathrm{AF}}$, respectively. Hotspot traffic, potentially comprising streaming video feeds or social media updates, originating from densely populated areas such as congested road intersections or music festivals, is generated by UEs at a rate of $\lambda_j$ and directed towards the $j^{th}$ AN-MEC site. This incoming traffic can be efficiently served with probability $P_j^{\mathrm{A}}$ at the source AN-MEC site, with probability $P_{j\rightarrow m}^{\mathrm{AA}}$ at another AN-MEC site, or with probability $P_{j,k}^{\mathrm{F}}$ at a vehicular-fog site. The control plane plays a crucial role in determining the optimal serving location to minimize average system latency. The possible latencies for each of these traffic types are defined as follows.

*1) Traffic Offloaded to $j^{th}$ AN-MEC:* The first type of traffic, shown in Fig. 3, is served at $j^{th}$ AN-MEC behind the base station with the probability $P_j^{\mathrm{A}}$. The latency experienced by this traffic is formulated as

$$l_j^{\mathrm{A}} = \frac{1}{\mu_j^{\mathrm{A}} - \lambda_j^{\mathrm{A}}},$$
$$\text{where } \lambda_j^{\mathrm{A}} = P_j^{A}\lambda_j + \sum_m P_{m\rightarrow j}^{\mathrm{AA}}\lambda_m, \quad \forall m \in \mathcal{M}\backslash\{j\}. \tag{1}$$

*2) Traffic Offloaded to Neighbour AN-MEC's ($m^{th}$ AN-MEC):* In Fig. 3, the overloaded $j^{th}$ AN-MEC could offload $P_{j\rightarrow m}^{\mathrm{AA}}$ traffic to its neighbor, $m^{th}$ AN-MEC. The total latency of offloaded traffic to $m^{th}$ AN-MEC site is formulated as

$$l_{j\rightarrow m}^{\mathrm{AA}} = \frac{1}{\mu_m^{\mathrm{A}} - \lambda_m^{\mathrm{A}}} + 2D_{j\rightarrow m}^{\mathrm{AA}},$$
$$\text{where } \lambda_m^{\mathrm{A}} = P_m^{A}\lambda_m + \sum_j P_{j\rightarrow m}^{\mathrm{AA}}\lambda_{j,}, \forall j \in \mathcal{M}\backslash\{m\}. \tag{2}$$

*3) Traffic Offloaded to $k^{th}$ Vehicular-Fog:* The arrival traffic also could be offloaded downward to a vehicular-fog in Fig. 3. The latency experienced by this traffic is formulated as

$$l_{j,k}^{\mathrm{F}} = \frac{1}{B_{j,k}^{\mathrm{AF}} - \lambda_{j,k}^{\mathrm{AF}}} + \frac{1}{\mu_{j,k}^{\mathrm{F}} - \lambda_{j,k}^{\mathrm{F}}} + \frac{1}{B_{j,k}^{\mathrm{FA}} - \lambda_{j,k}^{\mathrm{AF}}},$$
$$\text{where } \lambda_{j,k}^{\mathrm{AF}} = \sum_{j=1}^{M} P_{j,k}^{\mathrm{F}}\lambda_j, \text{ and } \lambda_{j,k}^{F} = \sum_{j=1}^{M} P_{j,k}^{F}\lambda_j. \tag{3}$$

Total traffic at $k^{th}$ vehicular-fog is the total of offloaded traffic from AN-MEC sites.

### B. Problem Formulation

The average latency experienced by arrival traffic at $j^{th}$ AN-MEC site, $l_j$, can be calculated as

$$l_j = \frac{l_j^{\mathrm{A}}\left(P_j^A \lambda_j\right) + \sum_m l_{j\to m}^{\mathrm{AA}}\left(P_{j\to m}^{\mathrm{AA}} \lambda_j\right) + \sum_k l_{j,k}^{\mathrm{F}}\left(P_{j,k}^{\mathrm{F}} \lambda_j\right)}{\lambda_j}, \quad \forall m \in \mathcal{M}\backslash\{j\}, \quad \forall k \in F_j, \tag{4}$$

whereas the average system latency is denoted as $l$ and can be calculated as

$$l = \frac{\sum_j l_j\left(\lambda_j\right)}{\sum_j \lambda_j}. \tag{5}$$

Hence, the objective function is to minimize the average system latency $l$, and is defined as follows.

$$\text{Minimize}(l), \tag{6}$$

subject to,

$$P_j^A + \sum_M P_{j\to m}^{AA} + \sum_{F_j} P_{j,k}^F = 1, \tag{7}$$

$$0 \le P_j^A, P_{j\to m}^{AA}, P_{j,k}^F \le 1, \tag{8}$$

$$P_j^A \lambda_j + \sum_M P_{j\to m}^{AA} \lambda_j + \sum_{F_j} P_{j,k}^F \lambda_j \le \lambda_j, \tag{9}$$

$$\lambda_j^{\mathrm{A}} \le \mu_j^A, \tag{10}$$

$$\lambda_m^{\mathrm{A}} \le \mu_m^A, \tag{11}$$

$$\lambda_{j,k}^{\mathrm{F}} \le \mu_{j,k}^F. \tag{12}$$

Equations (7) and (8) guarantee that the offloading ratio is bounded within the range of [0, 1] and the sum of these ratios equals 1. Equation (9) safeguards that the offloaded traffic remains below the volume of incoming hotspot traffic. Equations (10), (11), and (12) collectively express that the network's existing resources can effectively accommodate the offloaded traffic.

## IV. Ratio-Based Offloading With Multi-Agent TD3

In this section, we discuss the implementation of ratio-based offloading in edge and vehicular-fog networks, leveraging an RL algorithm for optimal performance. RL is a ML paradigm that guides agents in solving tasks through trial and error. TD3, an off-policy RL algorithm, is specifically designed to mitigate overestimation. It is exclusively applicable in environments with continuous action spaces, making it particularly well-suited for ratio-based offloading. In this paper, we proposed a multi-agent TD3 algorithm, which builds upon the TD3 algorithm to address ratio-based offloading decisions. This algorithm utilizes $M$ agents, each equipped with its own actor and target actor networks.

### A. RL Environment of Edge and Vehicular-Fog Network

In this subsection, we model the formulated optimization problem as a Markov decision process (MDP) [27]. The primary objective of action selection is to maximize the reward function. Within the edge and vehicular-fog network environment, agents are deployed at control planes as shown in Fig. 4. At each discrete time step $t$, each agent perceives a state $s_t$ from a designated state space $S$ and then selects an action $a_t$ from the available action space $\mathcal{A}$. The goal is to transition from the current state $s_t$ to a new state $s_{t+1}$ by following a policy $\pi(a_t, s_t)$, resulting in a reward $r_t$ based on the reward function $R(s_t, a_t)$. This iterative process continues until the agent reaches a terminal state, where the primary objective is to maximize the expected cumulative rewards [28]. The following elaboration offers comprehensive insights into the composition of the state space, action space, and reward function of the edge and vehicular-fog network environment.

TABLE II
State Information for Single and Multi-Agent TD3

| Input type | single-agent TD3 | multi-agent TD3 |
|---|---|---|
| $T_t$ | $\{\lambda_j\}_{j=1}^{M}$ | $\{\lambda_j\}$ |
| $C_t$ | $\left\{\mu_j^{\mathrm{A}}\right\}, \left\{\mu_{j,k}^{\mathrm{V}}\right\}_{j=1,k=1}^{M,F_j}$ | $\left\{\mu_j^{\mathrm{A}}\right\}, \left\{\mu_{j,k}^{\mathrm{V}}\right\}_{k=1}^{F_j}$ |
| $N_t$ | $\left\{B_{j,k}^{\mathrm{FA}}, B_{j,k}^{\mathrm{AF}}\right\}_{j=1,k=1}^{M,F_j}$ | $\left\{B_{j,k}^{\mathrm{FA}}, B_{j,k}^{\mathrm{AF}}\right\}_{k=1}^{F_j}$ |
| $L_t$ | $\{l_j\}_{j=1}^{M}$ | $\{l_j\}$ |

*1) State:* The following components constitute the state information for the edge and vehicular-fog sites: topology $(\mathcal{M}, F_j)$, arrival traffic rate at $j^{th}$ AN-MEC site ($\lambda_j^{\mathrm{A}}$), computation resources ($\mu_j^{\mathrm{A}}$, $\mu_{j,k}^{\mathrm{F}}$), networking resources ($B_{j,k}^{\mathrm{FA}}$, $B_{j,k}^{\mathrm{AF}}$), and system latency ($l_j$). The state arrays are constructed from these elements and serve as inputs to the policy and Q-value networks,

$$T_t = \{\lambda_j\}_{j=1}^{M}, \tag{13}$$

$$C_t = \left\{\mu_j^{\mathrm{A}}\right\}, \left\{\mu_{j,k}^{\mathrm{F}}\right\}_{j=1,k=1}^{M,F_j}, \tag{14}$$

$$N_t = \left\{B_{j,k}^{\mathrm{FA}}, B_{j,k}^{\mathrm{AF}}\right\}_{j=1,k=1}^{M,F_j}, \tag{15}$$

$$\text{and } L_t = \{l_j\}_{j=1}^{M}. \tag{16}$$

The set of arrival traffic rates for all AN-MEC sites at the $t^{th}$ step is denoted by $T_t$. $C_t$ is an array that contains information about the computing capabilities of the AN-MECs and vehicular-fogs. $N_t$ represents a set of networking capacities. Specifically, the computing and networking capacities vary over time in response to the arrival rate of traffic. $L_t$ is a set encompassing all AN-MEC latencies. At the $t^{th}$ time step, the environment state is an array formed by combining $T_t$, $C_t$, $N_t$, and $L_t$ and is denoted as

$$s_t = [T_t, C_t, N_t, L_t]. \tag{17}$$

The state information for single-agent TD3 and multi-agent TD3 offloading scenarios is given in Table II.

*2) Action:* The action represents the offloading ratio at the $j^{th}$ AN-MEC site, comprising probabilities for offloaded traffic

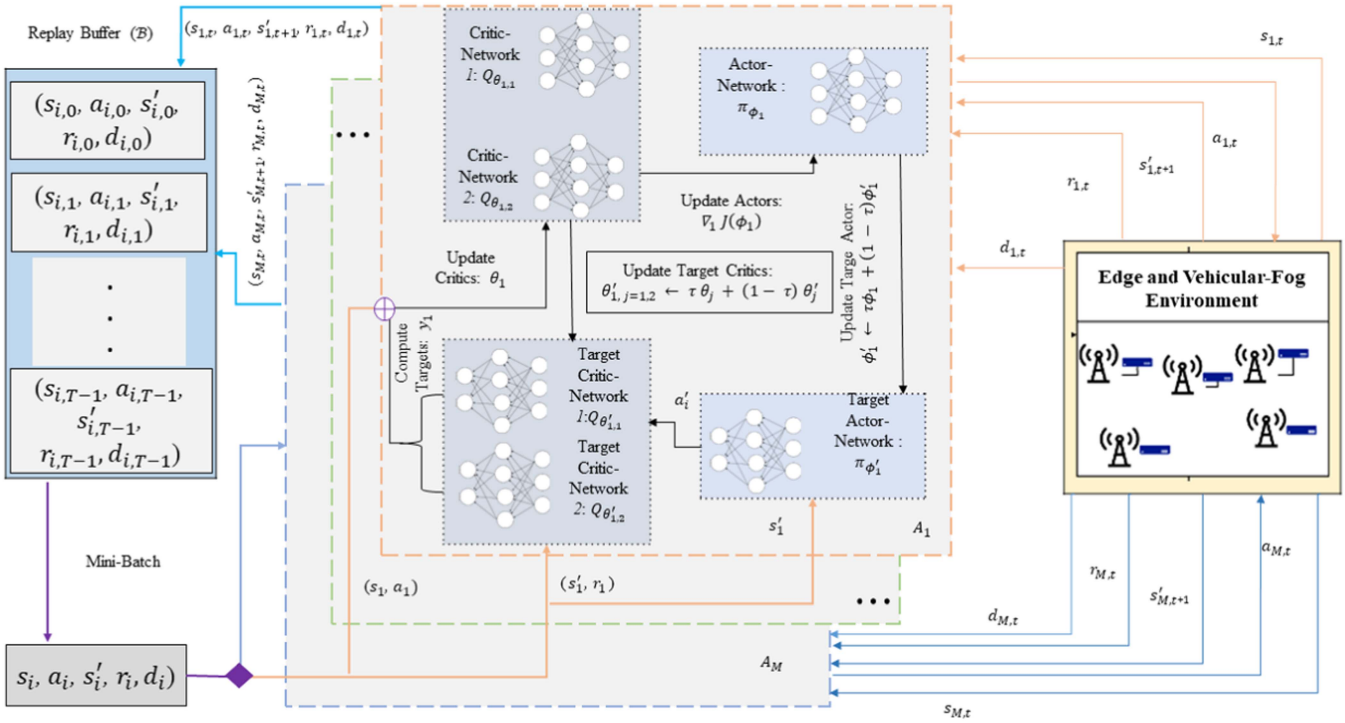



Fig. 4. Proposed multi-agent TD3 model.

destinations formulated as follows: $j^{th}$ AN-MEC ($P_j^A$), $j^{th}$ AN-MEC neighbors ($P_{j\to m}^{\mathrm{AA}}, \forall m \in \mathcal{M}\backslash\{j\}$), and $k^{th}$ vehicular-fog ($P_{j,k}^{\mathrm{F}}, \forall k \in F_j$). The total offloading ratio of each arrival traffic rate at the AN-MEC site is one, i.e., $P_j^{\mathrm{A}} + P_{j\to m}^{\mathrm{AA}} + P_{j,k}^{\mathrm{F}} = 1$, where $\forall m \in \mathcal{M}\backslash\{j\}$ and $\forall k \in F_j$. The action set, which is produced by the actor network using a single-agent TD3, is denoted as

$$a_t = \left\{P_j^{\mathrm{A}}; P_{j\to m}^{\mathrm{AA}}, \forall m \in \mathcal{M}\backslash\{j\}; P_{j,k}^{\mathrm{F}}, \forall k \in F_j\right\}_{j=1}^{M}. \quad (18)$$

In the case of multi-agent TD3, each agent is exclusively responsible for making offloading decisions pertaining to the $j^{th}$ AN-MEC site, and its action set is defined as

$$a_t = P_j^{\mathrm{A}}; P_{j\to m}^{\mathrm{AA}}, \forall m \in \mathcal{M}\backslash\{j\}; P_{j,k}^{\mathrm{F}}, \forall k \in F_j. \quad (19)$$

*3) Reward:* We do not train the agent by creating a predefined best-action direction using labeled data; instead, we generate a reward signal to indicate the effectiveness of the action taken. A positive reward is given for actions that help the system achieve its objective. Otherwise, the agent will be negatively rewarded. The RL's goal is to maximize the cumulative reward. Considering that the objective of offloading optimization is to minimize average latency, the reward for an action taken in $t^{th}$ time step is defined as

$$r_t = (Z_l(t)), \quad (20)$$

where $Z_l(t) = 1/l$, and $l$ is derived from (5).

### B. Proposed Multi-Agent TD3 Algorithm

To address the optimization problem at hand, we adopt a multi-agent RL approach to achieve a viable solution. The multi-agent TD3 (Algorithm 1) is built on TD3 and will be deployed at the control plane. TD3 is one of the RL algorithms that has a continuous action space and reduces overestimation [29]. The agent is responsible for finding an optimal offloading ratio by taking an action that maximizes its expected reward. This offloading ratio contains the destination of the offloaded traffic (vertical and horizontal directions) and how much traffic will be offloaded. The single-agent employs global state information, illustrated in Table II, as input to determine a comprehensive action encompassing offloading ratios for all AN-MEC sites. This requirement leads to a significant input space due to the extensive global observation, coupled with a substantial action space to encompass offloading ratios for all sites. The actor and critic networks of the single-agent undergo training via a replay buffer containing previously recorded data. In the multi-agent TD3 setting, we consider one agent for each AN-MEC site in the edge and vehicular-fog environments. This means we will have $M$ agents denoted as $\{A_1, A_2, \ldots, A_M\}$ to make $M$ different actions (offloading ratio) $a_1, a_2, \ldots, a_M$ which each action sums up to 1. Therefore, we use $M$ actor policy networks $\pi_{\phi_1}, \ldots, \pi_{\phi_M}$ with parameter $\phi_1, \ldots, \phi_M$ and actor-target policy networks $\pi'_{\phi_1}, \ldots, \pi'_{\phi_M}$ with parameter $\phi'_1, \ldots, \phi'_M$ for $M$ agents. We also, use for each agent $i$ two critic networks $Q_i = \{Q_{\theta_{i,1}}, Q_{\theta_{i,2}}\}$ with parameters $\theta_{i,1}, \theta_{i,2}$ and critic-target networks $Q'_i = \{Q_{\theta'_{i,1}}, Q_{\theta'_{i,2}}\}$ with parameters $\theta'_{i,1}, \theta'_{i,2}$ to minimize and correct overestimation. The agents will use the same replay buffer $\mathcal{B}$ to store and sample experience into and from

it to optimize our neural networks. Specifically, at each time $t$, an agent $i$ will observe the $s_{i,t}$ and take an action $a_{i,t}$ using the actor policy networks as in (21).

$$a_i(s_{i,t}) = clip(\pi_{\phi_i}(s_{i,t}) + \epsilon, a_{\min}, a_{\max}), \epsilon \sim N(0, \sigma), \quad (21)$$

where $\epsilon$ is Gaussian noise, and $a_{\min}$ and $a_{\max}$ are the lower and higher values of the action space that an agent can choose from. After the action is executed in edge and vehicular-fog environment, the $i^{th}$ agent will receive the next state $s'_{i,t+1}$, reward $r_{i,t}$, and done flag $d_{i,t}$ alongside the previous state $s_{i,t}$ and action $a_{i,t}$ as a single transition $(s_{i,t}, a_{i,t}, s'_{i,t+1}, r_{i,t}, d_{i,t})$, which is store in replay buffer $\mathcal{B}$.

Once we have enough experience in the buffer, the $i^{th}$ agent will update its neural network by taking a random mini-batch of $\mathcal{N}$ transitions $(s_i, a_i, s'_i, r_i, d_i)$ from the buffer. To compute the target value an agent has to select the action $a'_i$ which can be taken in the next state $s'_i$ using the actor-target policy and apply a clip noise to make sure the policy never gets stuck by selecting an overestimating action (22). Then it computes its target value $y_i$ using (23). The $i^{th}$ agent will use both its critic-target networks to estimate the global view of the next state in action by selecting the one with the minimal estimate.

$$a'_i(s'_i) = clip(\pi_{\phi'_i}(s'_i) + clip(\epsilon, -c, c), a_{\min}, a_{\max}), \quad \epsilon \sim N(0, \sigma), \quad (22)$$

$$y_i = r + \gamma \min_{\theta_{i,j=1,2}} Q_{\theta'_{i,j}}\left(s'_i, a'_{i,t}\right). \quad (23)$$

Once the $i^{th}$ agent has its target value $y_i$, it will update each of its critic networks $\theta_{i,j=1,2}$ by calculating the mean squared error of these batch of observations and applying a step of gradient descent as in (24).

$$\theta_{i,j} \leftarrow \text{argmin}_{\theta_{i,j=1,2}} \frac{1}{\mathcal{N}} \sum \left(y_i - Q_{\theta_{i,j}}(s_i, a_i)\right)^2. \quad (24)$$

The $i^{th}$ agent will update its actor policy $\phi_i$ using one of the critic networks at each $u$ time steps which is an update interval to estimate the Q-value of the state and the action selected by the policy and then we are going to apply a step of gradient ascent to push the parameters of the neural network $\phi_i$ in the direction of maximum growth of the Q-value (25).

$$\nabla_{\phi_i} J(\phi_i) = \frac{1}{\mathcal{N}} \sum \nabla_{a_i} Q_{\theta_{i,1}}(s_i, a_i)|a_i = \pi_{\phi_i}(s_i) \nabla_{\phi_i} \pi_{\phi_i}(s_i). \quad (25)$$

Once it updates the actor networks then it will update its target networks using (26) and (27).

$$\phi'_i \leftarrow \tau\phi_i + (1-\tau)\phi'_i, \quad (26)$$

$$\theta'_{i,j=1,2} \leftarrow \tau\theta_{i,j} + (1-\tau)\theta'_{i,j}. \quad (27)$$

This process will continuously repeat until the training epoch $t$ reaches $T$ and along the way, the agent will able to give the optimal offloading decision that can satisfy the requirement of the objective function in the given edge and vehicular-fog environment.

**Algorithm 1:** Proposed Multi-Agent TD3 Algorithm.

**Input**: Arrival hotspot traffic and No. Agents $M$
**Output**: Offloading ratio
Initialize agents actor networks $\pi_{\phi_1}, \cdots, \pi_{\phi_M}$ with random parameters $\phi_1, \cdots, \phi_M$;
Initialize agents target actor networks $\pi'_{\phi_1}, \cdots, \pi'_{\phi_M}$ with random parameters $\phi'_1, \cdots, \phi'_M$;
Initialize agents critic networks $Q_i, \cdots, Q_M$,where $Q_i = \{Q_{\theta_{i,1}}, Q_{\theta_{i,2}}\}$ with random parameters $\theta_{i,1}, \theta_{i,2}$;
Initialize agents target critic networks $Q_i, \cdots, Q_M$, where $Q'_i = \{Q_{\theta'_{i,1}}, Q_{\theta'_{i,2}}\}$ with random parameters $\theta'_{i,1}, \theta'_{i,2}$;
Initialize replay buffer $\mathcal{B}$;
**for** $i = 1 \cdots M$ *(in parallel)* **do**
  **for** $t = 1$ *do* $T$ **do**
    Observe state $s_{i,t}$;
    Select actions: $a_i(s_{i,t}) = clip(\pi_{\phi_i}(s_{i,t}) + \epsilon, a_{min}, a_{max}), \epsilon \sim N(0, \sigma)$;
    Observe next state $s'_{i,t+1}$ reward $r_{i,t}$, and done flag $d_{i,t}$ to indicate $s'_{i,t}$ is the terminal state save transition $(s_{i,t}, a_{i,t}, s'_{i,t+1}, r_{i,t}, d_{i,t})$ in replay buffer $\mathcal{B}$;
    Sample random mini-batch of $\mathcal{N}$ transitions $(s_i, a_i, s'_i, r_i, d_i)$ from $\mathcal{B}$;
    Select target action: $a'_i(s'_i) = clip(\pi_{\phi'_i}(s'_i) + clip(\epsilon, -c, c), a_{min}, a_{max}), \epsilon \sim N(0, \sigma)$;
    Compute target value: $y_i = r_i + \gamma \min_{\theta_{i,j=1,2}} Q_{\theta'_{i,j}}(s'_i, a'_{i,t})$;
    Update critic networks
    $\theta_{i,j} \leftarrow argmin_{\theta_{i,j=1,2}} \frac{1}{N} \sum \left(y_i - Q_{\theta_{i,j}}(s_i, a_i)\right)^2$;
    **if** $t$ *mod* $u = 0$ **then**
      Update actor networks using deterministic policy gradient:
      $\nabla_{\phi_i} J(\phi_i) = \frac{1}{N} \sum \nabla_{a_i} Q_{\theta_{i,1}}(s_i, a_i) \mid a_i$
      $= \pi_{\phi_i}(s_i) \nabla_{\phi_i} \pi_{\phi_i}(s_i)$;
      Update target actor networks:
      $\phi'_i \leftarrow \tau\phi_i + (1-\tau)\phi'_i$;
      Update target critic networks:
      $\theta'_{i,j=1,2} \leftarrow \tau\theta_{i,j} + (1-\tau)\theta'_{i,j}$;
    **end**
  **end**
**end**

## V. Simulation and Results

Edge and vehicular-fog systems are distributed architectures that encompass tens to hundreds of computing resources. To evaluate the performance of the proposed multi-agent TD3 algorithm, we conducted a comparative analysis involving two other algorithms: the single-agent TD3 and the Simulated Annealing (SA) algorithms [30], as an integral part of our evaluation process. Furthermore, the network abstractions used in simulations closely resemble those employed in the actual network control plane.

### A. Parameter Settings

The edge and vehicular-fog environment were developed in Python, following the standards of the OpenAI Gym framework [31]. We rigorously assessed its performance and suitability using the Stable-Baselines3 framework [32]. The simulations were executed on a desktop system featuring a tenth-generation Core i7 processor, 16 GB of RAM, and an RTX 2060 Super

TABLE III
PARAMETER SETTINGS

| Parameters | Notations | Values |
|---|---|---|
| **Topology:** | | |
| Number of AN-MEC site | $\mathcal{M}$ | 2, 3, 4, 5, 6 |
| Number of vehicular-fog sites connected to an AN-MEC site | $F_j$ | 8 |
| Total number of sites | $\mathcal{M} * (F_j + 1)$ | 18, 27, 36, 45, 54 |
| Distance between ANs | $d^{\mathrm{AA}}_{i,j \leftrightarrow m}$ | $\approx$ 1km – 10km |
| **Capacity:** | | |
| Downlink, Uplink | $B^{\mathrm{AF}}_{j,k}$, $B^{\mathrm{FA}}_{j,k}$ | 1 Gbps |
| AN-MEC site | $\mu^{\mathrm{A}}_{j}$ | $30 \times 10^3$ MIPS |
| vehicular-fog site | $\mu^{\mathrm{F}}_{j,k}$ | $15 \times 10^3$ MIPS |
| **Traffic:** | | |
| Normal traffic rate | $\lambda_j$ | 10K, 20K, 30K, 40K packet/s |
| Hotspot traffic rate | $\lambda_j$ | 20K, 40K, 60K, 80K packet/s |
| Packet size | $\vartheta$ | 10 Kb |
| **Agent Training Parameters:** | | |
| Batch Size | $(\mathcal{N})$ | 100 |
| Learning rate | $(\alpha)$ | 0.001 |
| Discount factor | $(\gamma)$ | 0.99 |
| Update rate | $(\tau)$ | 0.005 |
| Number of hidden layers | NHL | 2 |
| Optimizer | Opt. | ADAM |

graphics processing unit. The simulation settings, encompassing topology, capacity, traffic, and agent training parameters, are outlined in Table III. The selection of topology parameters draws upon insights from the paper [4]. Optimal hyperparameters for training the agent were determined through the utilization of Optuna, an automatic hyperparameter optimization software framework [33]. By utilizing AN-MEC sites with varying quantities, we investigated the impact of system size on the algorithms and the number of agents. AN-MEC sites were evenly distributed across a region using a geographic coordinate sampling application [34]. The AN-MEC site capacity is ten times greater than of the uplink and downlink capacity. Additionally, we evaluated the proposed method under various arrival traffic rates, as listed in Table III.

### B. Result Analysis

The experimental results were organized to address specific research questions: decision time, convergence time, decision quality (or service performance), and decision with Unpredictable Traffic in the context of RL vs. SA, and a comparison between single-agent and multi-agent offloading.

*1) Decision time—RL vs. SA:* In the edge and vehicular-fog environment, the network control plane, located within the AN-MEC, is responsible for making ratio-based offloading decisions. We assumed that idle vehicles form a pool of computing resources within the vehicular-fog environment, and they have no tasks to offload. Offloading a task from a vehicle indicates a scarcity of computing resources. The control plane's algorithm determines the offloading ratio while taking environmental conditions into account. Decision time refers to the time required

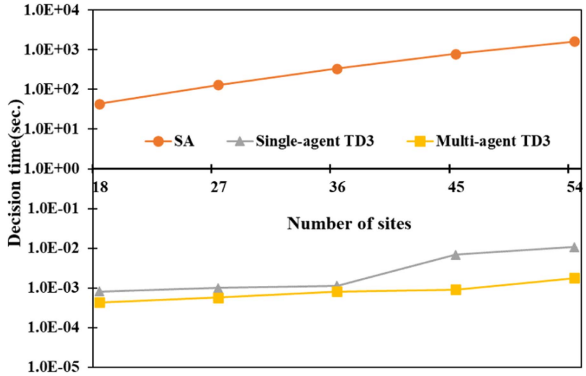



Fig. 5. Decision time.

by the control plane to calculate the offloading ratio, measured in seconds to accommodate fluctuating traffic rates.

Fig. 5 depicts the comparison of decision time between TD3-based offloading and SA offloading. The SA took 365 to 4,909 seconds to determine the offloading ratio for systems with 18 and 54 sites, respectively. Offloading based on TD3 can simultaneously determine the offloading ratio and train the offloading model, resulting in quicker decisions compared to SA, which must complete calculations before establishing the offloading ratio. In TD3-based offloading, a neural network (NN) was employed to determine the offloading ratio. The complexity of this NN is influenced by the number of layers and neurons within the hidden units, as discussed in [35]. The multi-agent TD3 approach has a shorter decision time compared to the single-agent TD3, as shown in Fig. 5. This reduction in decision time can be attributed to the smaller size of the NN in multi-agent TD3, a consequence of multi-agent RL making offloading ratio decisions for the incoming traffic of a single AN-MEC site.

The structure of the NN is influenced by the input state size and action size, as shown in the comparison presented in Fig. 6(a), and (b), respectively, where single-agent TD3 and SA possess larger state and action spaces than multi-agent TD3. In systems with 18 and 54 sites, multi-agent TD3 made decisions in 0.4 and 1 milliseconds, respectively, while single-agent TD3 decisions took 0.8 and 10 milliseconds for the same system configurations. The decision time of TD3 agents is five orders of magnitude faster than SA. For control plane decisions, the TD3-based approach is recommended due to its ability to make decisions at the millisecond or second scale, effectively handling fluctuating traffic rates. TD3 models need to be pre-trained, as they rely on trial-and-error; otherwise, they may take longer to converge towards optimal decisions. In contrast, SA decisions, which take tens of minutes, may become outdated given current traffic rates.

*2) Convergence Time—RL vs. SA:* Convergence time refers to the duration needed for an algorithm to reach its optimal solution. SA employs exhaustive searching to determine the best offloading decision, as illustrated in Fig. 7. It has the longest convergence times of 365 and 4,909 seconds in the environments with 18 and 54 sites, respectively. In this study, instead of employing random actions as seen in TD3's initial training rounds, a weighted-based offloading technique was utilized to gather

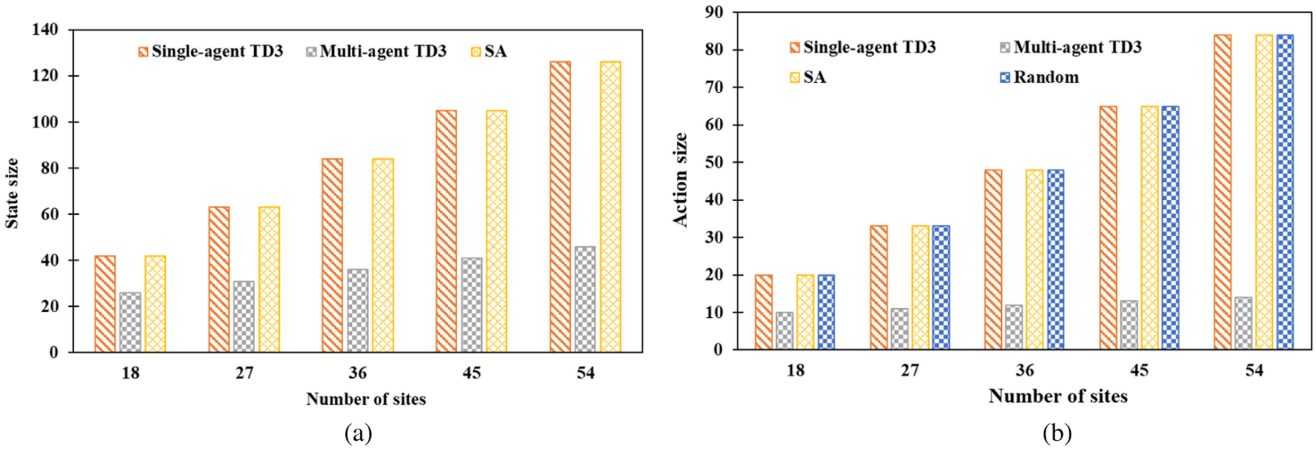



Fig. 6. State and action space comparison. (a) State space. (b) Action space.

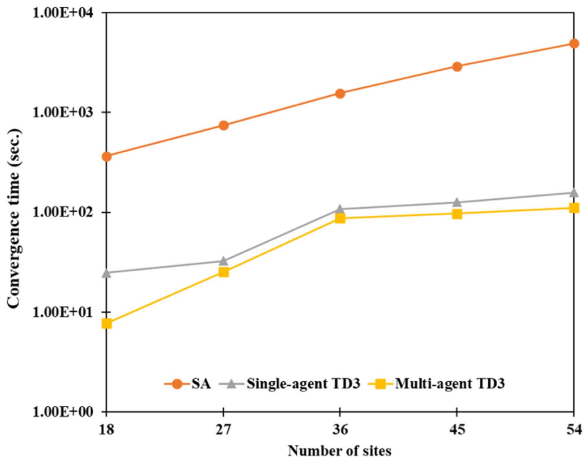



Fig. 7. Convergence time.

experience data for training. This approach led to quicker convergence times for TD3 models compared to the SA method. For the 18-site environment, single-agent TD3 and multi-agent TD3 approaches required 24 and 7 seconds, respectively. Similarly, in the 54-site environment, single-agent TD3 and multi-agent TD3 approaches took 157 and 111 seconds, respectively.

The SA algorithm required exhaustive searching with time complexity of $O(n^{k\times l})$, where $n$ represents the number of inputs, which in this context is the offloading ratio array; $k$ denotes the decreased temperature, and $l$ signifies the number of neighbors visited for each temperature. The convergence time of the SA algorithm increased significantly as the number of sites grew. Its time complexity exhibits exponential growth. The offloading strategy of TD3-based models utilizes NN architecture, the complexity of which depends on the number of inputs $n$, epochs $t$, and the number of hidden layers $k$. The layers of the NN involve matrix multiplication with a complexity of $m^3$. The overall complexity of NN training is $O(nt(km^3))$ or $O(nt(m^3))$ if the constant $k$ is omitted. The number of inputs includes both state and action sizes, which are influenced by the number of sites. As shown in Figs. 6(a) and (b), an increase in the number of sites leads to a simultaneous increase in both state and action sizes. Consequently, as the number of inputs rises, the computational operation of the NN also increases. It is important to note that the NN operation shows a linear relationship with the number of inputs. Since the number of agents remains constant and does not affect either the number of inputs or the operations of the neural network (NN), we omitted it from our analysis. Additionally, we executed the multi-agent algorithm in parallel, resulting in a time complexity of $O(nt(m^3))$. Fig. 5 demonstrates a slight increase in the decision time of the TD3-based offloading agent as the number of sites grows. This trend results from ratio-based offloading, which involves determining the percentage of traffic allocated to each site. As the number of sites increases, the search space also expands asymptotically. Within this expanding search space, TD3-based models and SA are tasked with finding the optimal offloading ratio. However, as the search space enlarges, this endeavor demands progressively more computational effort and time, leading to an asymptotic increase in convergence time.

The SA process involves the computation of the current solution as well as possible solutions generated through a random function. It subsequently compares these solutions and calculates a probability function to determine the feasibility of accepting a suboptimal solution. These computations contribute to SA's longer processing time compared to the TD3-based approach, as shown in Fig. 8. Transition data is stored in a single replay buffer to train both single-agent and multi-agent TD3-based offloading. The primary distinction lies in the state information provided to these agents. In a multi-agent TD3 setting, the stored state of an agent does not encompass details about vehicular-fog of other AN-MEC sites, whereas it does in a single-agent TD3 context due to the need to determine the offloading ratios for all AN-MEC sites. This discrepancy in state information affects the size (number of hidden units) and structure of the NNs, resulting in the multi-agent TD3 having a smaller NN size and structure compared to the single-agent TD3. As a result, the step time of the multi-agent TD3 is shorter than that of the single-agent TD3.

The algorithm's response time to a new traffic pattern was also evaluated. TD3 required 2,000 to 7,000 steps to approximate the optimal reward achieved by SA, which reached 130,000 steps, as illustrated in Fig. 9. Since SA needs to visit each AN-MEC

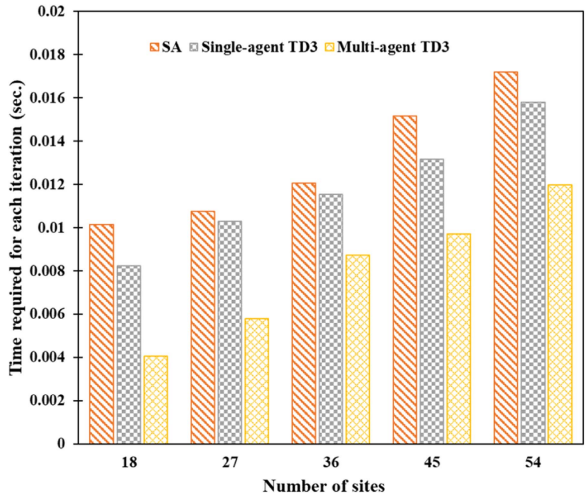



Fig. 8. Processing time of each step.

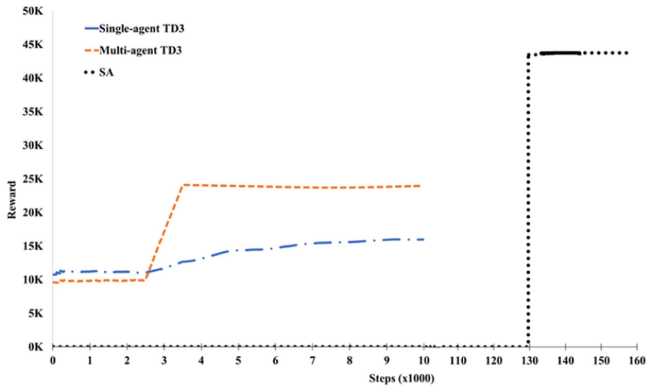



Fig. 9. Reward regret.

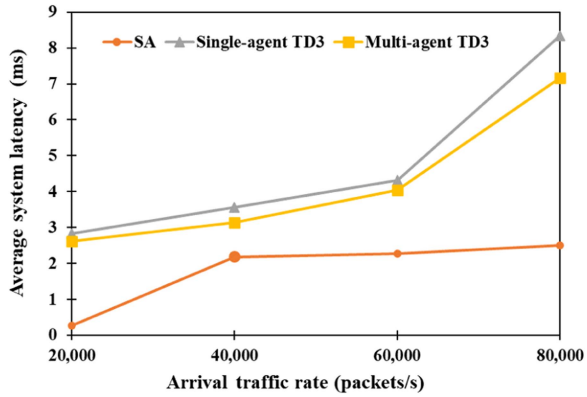



Fig. 10. Offloading performance.

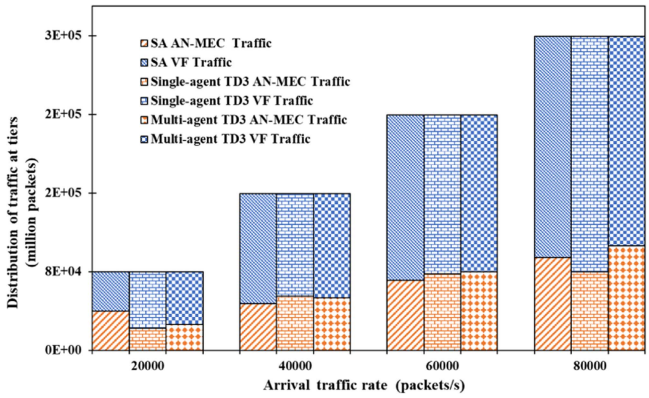



Fig. 11. Traffic distribution.

to optimize its offloading ratio, an unvisited site experiencing traffic exceeding its capacity leads to significant latency and yields a nearly negligible reward.

*3) Decision Quality—RL vs. SA:* This study aims to determine the optimal offloading ratio that results in a low average arrival traffic latency. We compare the traditional optimization method, SA, with the TD3-based approach. The evaluation includes four traffic rates representing scenarios ranging from low to heavy traffic: 20K, 30K, 60K, and 80K packets/s. In each scenario, normal traffic is combined with hotspot traffic, generated at twice the rate of normal traffic, to certain AN-MEC sites.

Fig. 10 shows that the average latency for SA with exhaustive searching is the lowest. At 0.2 and 2.5 milliseconds, SA efficiently served traffic rates of 20K and 80K packets per second, respectively. On the other hand, the TD3-based method, trained using experience data, exhibited an average latency of 2.8 and 2.6 milliseconds when serving 20K packets/s with a single-agent and with multiple agents, respectively. For serving 80K packets/s, single and multiple agents required 8.3 and 7.2 milliseconds, respectively. While SA-based offloading boasts the lowest average latency, its decision-making process takes minutes or even hours. This extended timeframe falls short of accommodating the rapid fluctuations in traffic rates that occur within seconds. In contrast, The TD3-based algorithm demonstrates similar order-of-magnitude average latency as the SA-based algorithm but achieves offloading ratio decisions at a rate five orders of magnitude faster, seamlessly adapting to swiftly changing traffic rates. Among the TD3 variants, the multi-agent TD3 approach that leverages local data proves more efficient than the single-agent TD3 counterpart, which relies on global data.

A base station or an AN-MEC could be connected to various vehicular-fog sites. Given that the number of vehicular-fog sites surpasses the count of AN-MEC sites, the combined capacity of vehicular-fog sites exceeds that of AN-MEC sites. Since both AN-MEC and vehicular-fog sites share proximity with a radio access network, their implementation occurs on a smaller scale, resulting in similar capacities. As shown in Fig. 11, vehicular-fog sites, possessing higher overall capacity, successfully handled a greater volume of traffic compared to the total capacity of AN-MEC sites.

Preferring a large capacity site over scattering small capacity sites across a geographical area is evident. As illustrated in Fig. 12, the AN-MEC site exhibits greater usage in comparison to the vehicular-fog site. Despite AN-MEC's higher utilization relative to vehicular-fog, it doesn't encounter elevated computing latency due to its superior capacity. The AN-MEC site's average utilization across all traffic scenarios ranged from 0.18

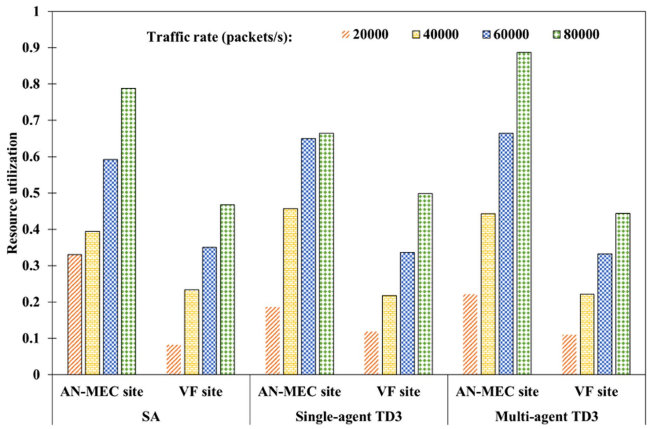



Fig. 12. Site utilization average.

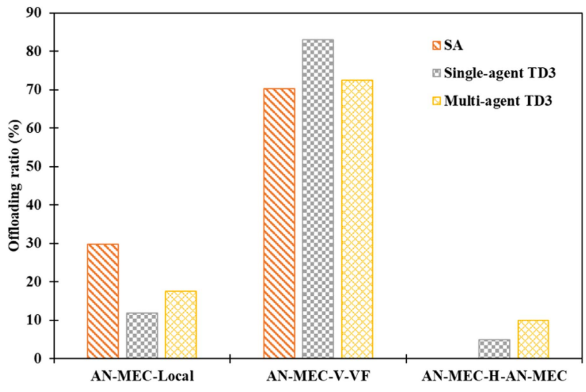
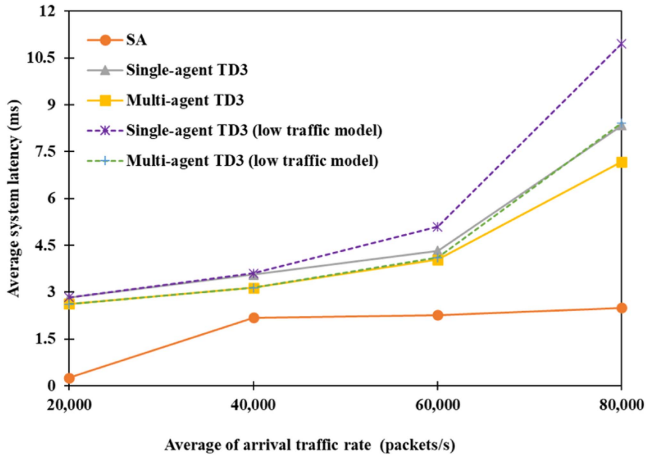
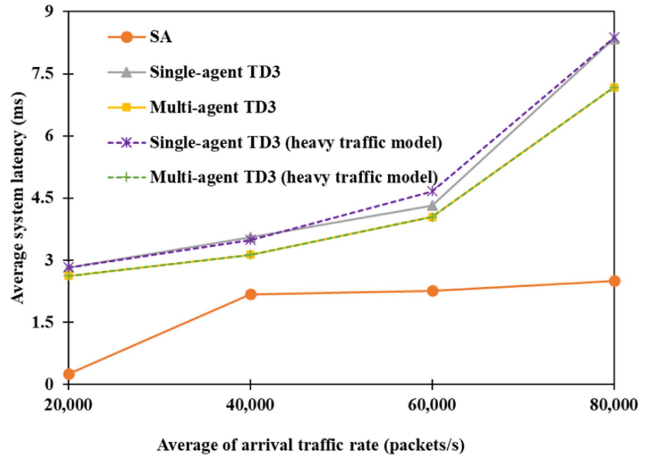



Fig. 13. Offloading ratio.




(a)




(b)

Fig. 14. Effect of fluctuating traffic to the agent model. (a) Low traffic model performance. (b) Heavy traffic model performance.

to 0.89, while the vehicular-fog site's average utilization ranged from 0.08 to 0.499.

The offloading ratios depicted in Fig. 13 indicate that all algorithms exhibit a stronger preference for vertical downward offloading from AN-MEC to vehicular-fog (AN-MEC-V-VF) as opposed to horizontal offloading to other AN-MEC sites (AN-MEC-H-AN-MEC). The term "AN-MEC-Local" represents the proportion of incoming traffic serviced by the arriving AN-MEC sites. In the case of SA, vertical offloading to vehicular-fog occurred primarily after addressing the local AN-MEC, since other AN-MEC sites might already be handling incoming traffic directed at them. On the other hand, the single and multi-agent TD3 approaches implemented horizontal offloading to different AN-MEC sites, incurring additional propagation latency and consequently resulting in a higher average latency compared to SA.

*4) Decision With Unpredictable Traffic– - RL vs. SA:* Additionally, we simulate transitions from light to heavy and heavy to light traffic during the evaluation. When traffic rates increase from low to heavy, the TD3 models, which have been trained on low traffic rate data, are utilized to handle unfamiliar heavy traffic rates. Conversely, when traffic decreases from heavy to low rates, the TD3 models are unaccustomed to the low traffic rates. This condition impacts the algorithm's performance, as illustrated in Fig. 14. In Figs. 14(a) and (b), we illustrate the responsiveness to traffic fluctuations. The model trained under light traffic conditions successfully managed fluctuating heavy traffic in Fig. 14(a), whereas the model trained under heavy traffic conditions effectively handled incoming light traffic in Fig. 14(b).

The model trained under heavy traffic demonstrated greater transferability in handling light traffic compared to the model trained under light traffic for managing heavy traffic. Fig. 14(a) illustrates the struggle of the lightweight traffic model of the single-agent TD3 in managing heavy incoming traffic. Deciding the global offloading ratio proved more challenging than determining the local offloading ratio, as observed in the multi-agent TD3 scenario. This complexity arises because the agent's NN needs to map a large input space to a corresponding desired output space, leading to challenges in maintaining offloading precision. The offloading ratio precision of the light traffic model in the single-agent TD3 framework couldn't be effectively applied to heavy traffic scenarios. Conversely, models trained under heavy traffic conditions exhibit precise direction of incoming traffic to available sites, resulting in low average latency. Applying such a model to light traffic, as depicted in Fig. 14(b), yields satisfactory results due to the comparative

simplicity of the task compared to managing heavy incoming traffic.

## VI. Conclusion and Future Work

In this paper, we address the computational offloading of arrival hotspot traffic in edge and vehicular-fog networks. We formulate an optimization problem with the objective of minimizing the average system latency. The ratio-based offloading optimization of the edge and vehicular-fog system employs a continuous action search to estimate the total offloaded traffic. Traditional strategies like SA's exhaustive search require 365 to 4,909 seconds for decisions, conflicting with the control plane's need for rapid offloading ratio determination due to fluctuating traffic. In contrast, TD3-based offloading's actor model rapidly maps the environment to action, achieving ratios in tens to hundreds of milliseconds, five orders of magnitude faster than SA. However, multi-agent TD3 provides more accurate and quicker decisions than single-agent TD3. Both single and multi-agent TD3 models approximate SA's decision quality with slightly higher average latencies of 2 to 6 milliseconds. Most traffic is offloaded to vehicular-fog sites despite their capacity being one order of magnitude lower than AN-MEC sites because vertical offloading is more effective than horizontal offloading. However, the average utilization of AN-MEC sites in serving heavy arrival traffic ranges from 66% to 89%, whereas for the vehicular-fog sites, it is between 44% and 50%. Both single-agent and multi-agent TD3 offloading demonstrate adaptable transferability, even with sub-optimal solutions. Models trained with high arrival traffic patterns prove to be adaptable to diverse scenarios and perform comparably at lower rates, considering system capacity limits.

For future work, there is a need to explore energy-efficient offloading strategies that balance computational power consumption with reduced latency. Additionally, moving beyond theoretical research and focusing on practical implementation and deployment of offloading solutions in real-world testbeds is essential. Evaluating the performance, scalability, and reliability of these systems in practical settings is a step we believe will yield invaluable insights and practical outcomes.

## References


[1] Y. Liu, Y. Chen, T. Deng, and F. Lin, "A data offloading strategy based on UE movement prediction in MEC," in *2022 IEEE 10th Int. Conf. Inf., Commun. Netw.*, 2022, pp. 229–235.

[2] F. Giust et al., "MEC deployments in 4G and evolution towards 5G," *ETSI White Paper*, vol. 24, no. 2018, pp. 1–24, 2018.

[3] C. Wan, S. Guo, and Y. Yang, "Deep reinforcement learning based computation offloading in swipt-assisted MEC networks," in *2022 Int. Conf. Comput. Commun. Netw.*, 2022, pp. 1–10.

[4] W. Yahya, E. Oki, Y.-D. Lin, and Y.-C. Lai, "Scaling and offloading optimization in pre-CORD and post-CORD multi-access edge computing," *IEEE Trans. Netw. Serv. Manage.*, vol. 18, no. 4, pp. 4503–4516, Dec. 2021.

[5] J. Feng and Z. Feng, "A vehicle-assisted offloading scheme for hotspot base stations on metropolitan streets," in *2017 IEEE 28th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, 2017, pp. 1–6.

[6] B. Kar, W. Yahya, Y.-D. Lin, and A. Ali, "Offloading using traditional optimization and machine learning in federated cloud-edge-fog systems: A survey," *IEEE Commun. Surv. Tut.*, vol. 25, no. 2, pp. 1199–1226, Second quarter 2023.

[7] Z. Wu and D. Yan, "Deep reinforcement learning-based computation offloading for 5G vehicle-aware multi-access edge computing network," *China Commun.*, vol. 18, no. 11, pp. 26–41, 2021.

[8] L.-H. Yen, J.-C. Hu, Y.-D. Lin, and B. Kar, "Decentralized configuration protocols for low-cost offloading from multiple edges to multiple vehicular fogs," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 872–885, Jan. 2021.

[9] Y. Zhou, M. Shi, Y. Tian, Y. Li, Q. Ye, and J. Lv, "Federated CINN clustering for accurate clustered federated learning," in *Proc. ICASSP 2024-2024 IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2024, pp. 5590–5594.

[10] Y.-D. Lin, J.-C. Hu, B. Kar, and L.-H. Yen, "Cost minimization with offloading to vehicles in two-tier federated edge and vehicular-fog systems," in *2019 IEEE 90th Veh. Technol. Conf.*, 2019, pp. 1–6.

[11] B. Kar, K.-M. Shieh, Y.-C. Lai, Y.-D. Lin, and H.-W. Ferng, "QoS violation probability minimization in federating vehicular-fogs with cloud and edge systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13270–13280, Dec. 2021.

[12] Y. Li, "Optimization of task offloading problem based on simulated annealing algorithm in MEC," in *2021 9th Int. Conf. Intell. Comput. Wireless Opt. Commun.*, 2021, pp. 47–52.

[13] A. Shakarami, M. Ghobaei-Arani, M. Masdari, and M. Hosseinzadeh, "A survey on the computation offloading approaches in mobile edge/cloud computing environment: A stochastic-based perspective," *J. Grid Comput.*, vol. 18, pp. 639–671, 2020.

[14] D. Saguil and A. Azim, "Time-efficient offloading for machine learning tasks between embedded systems and fog nodes," in *2019 IEEE 22nd Int. Symp. Real-Time Distrib. Comput.*, 2019, pp. 79–82.

[15] A. Shakarami, M. Ghobaei-Arani, and A. Shahidinejad, "A survey on the computation offloading approaches in mobile edge computing: A machine learning-based perspective," *Comput. Netw.*, vol. 182, 2020, Art. no. 107496.

[16] M. Guo, M. Mukherjee, G. Liang, and J. Zhang, "Computation offloading for machine learning in industrial environments," in *Proc. IECON 2020 46th Annu. Conf. IEEE Ind. Electron. Soc.*, 2020, pp. 4465–4470.

[17] C. Sonmez, C. Tunca, A. Ozgovde, and C. Ersoy, "Machine learning-based workload orchestrator for vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2239–2251, Apr. 2021.

[18] J. Shi, J. Du, J. Wang, and J. Yuan, "Deep reinforcement learning-based V2V partial computation offloading in vehicular fog computing," in *2021 IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–6.

[19] C. Pan, Z. Wang, Z. Zhou, and X. Ren, "Deep reinforcement learning-based URLLC-aware task offloading in collaborative vehicular networks," *China Commun.*, vol. 18, no. 7, pp. 134–146, 2021.

[20] M. Khayyat, I. A. Elgendy, A. Muthanna, A. S. Alshahrani, S. Alharbi, and A. Koucheryavy, "Advanced deep learning-based computational offloading for multilevel vehicular edge-cloud computing networks," *IEEE Access*, vol. 8, pp. 137052–137062, 2020.

[21] D. Lan, A. Taherkordi, F. Eliassen, and L. Liu, "Deep reinforcement learning for computation offloading and caching in fog-based vehicular networks," in *2020 IEEE 17th Int. Conf. Mobile Ad Hoc Sensor Syst.*, 2020, pp. 622–630.

[22] U. Maan and Y. Chaba, "Deep Q-network based fog node offloading strategy for 5G vehicular ad-hoc network," *Ad Hoc Netw.*, vol. 120, 2021, Art. no. 102565.

[23] S. Vemireddy and R. R. Rout, "Fuzzy reinforcement learning for energy efficient task offloading in vehicular fog computing," *Comput. Netw.*, vol. 199, 2021, Art. no. 108463.

[24] J. Xie, Q. Si, and Y. Tang, "Multi-armed bandit based task offloading by probabilistic V2X communication in vehicle edge cloud system," in *2020 IEEE 91st Veh. Technol. Conf.*, 2020, pp. 1–5.

[25] X. Zhang, W. Wu, Z. Zhao, J. Wang, and S. Liu, "RMDDQN-learning: Computation offloading algorithm based on dynamic adaptive multi-objective reinforcement learning in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 11374–11388, Sep. 2023.

[26] Y. Cui, H. Li, D. Zhang, A. Zhu, Y. Li, and H. Qiang, "Multi-agent reinforcement learning based cooperative multitype task offloading strategy for Internet of Vehicles in B5G/6G network," *IEEE Internet Things J.*, vol. 10, no. 14, pp. 12248–12260, Jul. 2023.

[27] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[29] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.

[30] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[31] G. Brockman et al., "OpenAI gym," 2016, *arXiv:1606.01540*.
[32] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html
[33] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 2623–2631.
[34] E. Sergeant, "Epitools epidemiological calculators," 2018.
[35] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 8, pp. 1553–1565, Aug. 2014.

**Frezer Guteta Wakgra** received the B.Sc. and M.Sc. degrees in software engineering from Adama Science and Technology University, Ethiopia, in 2013 and 2017, respectively. He is currently working toward the Ph.D. degree with the National Taiwan University of Science and Technology, Taipei, Taiwan. His research interests include cloud computing, multi-access edge computing (MEC), fog computing, optimization, and deep reinforcement learning.

**Widhi Yahya** received the Ph.D. degree in electrical engineering and computer sciencefrom the International Graduate Program of National Yang Ming Chiao Tung University, Hsinchu, Taiwan, in 2023. He is currently a Lecturer and a Researcher in computer science with the University of Brawijaya, Malang, Indonesia. His research interests include network programming, software-defined networking, and multi-access edge computing (MEC) optimization.

**Binayak Kar** (Member, IEEE) received the Ph.D. degree in computer science and information engineering from National Central University (NCU), Taoyuan City, Taiwan, in 2018. He is currently an Assistant Professor of computer science and information engineering with the National Taiwan University of Science and Technology, Taipei, Taiwan. He was a Postdoctoral Research Fellow in computer science with National Chiao Tung University, Hsinchu, Taiwan, from 2018 to 2019. His research interests include edge computing, optimization, machine learning, cybersecurity, and quantum computing.

**Ying-Dar Lin** (Fellow, IEEE) received the Ph.D. degree in computer science from the University of California at Los Angeles, Los Angeles, CA, USA, in 1993. He is currently a Chair Professor of computer science with National Yang Ming Chiao Tung University, Hsinchu, Taiwan. His research interests include network softwarization, cybersecurity, and wireless communications. His work on multi-hop cellular was the first along this line, and has been cited over 1000 times. He is an IEEE Distinguished Lecturer. He has served or is serving on the editorial boards of several IEEE journals and magazines, and was the Editor-in-Chief ofIEEE COMMUNICATIONS SURVEYS AND TUTORIALS(COMST), during 2016–2020.

**Yuan-Cheng Lai** received the Ph.D. degree from the Department of Computer and Information Science from National Chiao Tung University, Hsinchu, Taiwan, in 1997. In 2001, he joined the Faculty of the Department of Information Management at National Taiwan University of Science and Technology, Taipei, Taiwan. He has been a Professor with the Department of Information Management at National Taiwan University of Science and Technology, since 2008. His research interests include performance analysis, wireless networks, network security, and machine learning.

**Seifu Birhanu Tadele** received the B.Sc. degree in computer science from Jimma University, Jimma, Ethiopia, in 2012, and the M.S. degree Software Engineering from Adama Science and Technology University, Adama, Ethiopia, in 2017. He is currently working toward the Ph.D. degree with the National Taiwan University of Science and Technology, Taipei, Taiwan. His research interests include cloud computing, multi-access edge computing, fog computing, optimization, and deep reinforcement learning.