IEEE ComSoc IEEE Communications Society | IEEE COMPUTER SOCIETY | IEEE Signal Processing Society | VTS | IEEE Transactions on Machine Learning in Communications and Networking

Received 18 December 2023; revised 16 May 2024 and 19 July 2024; accepted 22 July 2024.
Date of publication 25 July 2024; date of current version 31 July 2024.

The associate editor coordinating the review of this article and approving it for publication was H. Ko.

Digital Object Identifier 10.1109/TMLCN.2024.3433620

Hierarchical Reinforcement Learning for Multi-Layer Multi-Service Non-Terrestrial Vehicular Edge Computing

SWAPNIL SADASHIV SHINDE [1,2] (Student Member, IEEE), AND DANIELE TARCHI [2] (Senior Member, IEEE)

[1]Consorzio Nazionale Interuniversitario delle Telecomunicazioni (CNIT), University of Bologna Research Unit, 40136 Bologna, Italy
[2]Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi," University of Bologna, 40136 Bologna, Italy

CORRESPONDING AUTHOR: D. TARCHI (daniele.tarchi@unibo.it)

This work was supported in part by European Commission under the "5G-STARDUST" Project, which received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe Research and Innovation Programme under Grant 101096479; in part by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future under Grant PE00000001-program "RESTART"; and in part by the Swiss State Secretariat for Education, Research and Innovation (SERI). The views expressed are those of the authors and do not necessarily represent the project. The Commission is not liable for any use that may be made of any of the information contained therein.

ABSTRACT Vehicular Edge Computing (VEC) represents a novel advancement within the Internet of Vehicles (IoV). Despite its implementation through Road Side Units (RSUs), VEC frequently falls short of satisfying the escalating demands of Vehicle Users (VUs) for new services, necessitating supplementary computational and communication resources. Non-Terrestrial Networks (NTN) with onboard Edge Computing (EC) facilities are gaining a central place in the 6G vision, allowing one to extend future services also to uncovered areas. This scenario, composed of a multitude of VUs, terrestrial and non-terrestrial nodes, and characterized by mobility and stringent requirements, brings in a very high complexity. Machine Learning (ML) represents a perfect tool for solving these types of problems. Integrated Terrestrial and Non-terrestrial (T-NT) EC, supported by innovative intelligent solutions enabled through ML technology, can boost the VEC capacity, coverage range, and resource utilization. Therefore, by exploring the integrated T-NT EC platforms, we design a multi-EC-enabled vehicular networking platform with a heterogeneous set of services. Next, we model the latency and energy requirements for processing the VU tasks through partial computation offloading operations. We aim to optimize the overall latency and energy requirements for processing the VU data by selecting the appropriate edge nodes and the offloading amount. The problem is defined as a multi-layer sequential decision-making problem through the Markov Decision Processes (MDP). The Hierarchical Reinforcement Learning (HRL) method, implemented through a Deep Q network, is used to optimize the network selection and offloading policies. Simulation results are compared with different benchmark methods to show performance gains in terms of overall cost requirements and reliability.

INDEX TERMS Vehicular networks, edge computing, non-terrestrial networks, computation offloading, reinforcement learning.

I. INTRODUCTION

WITH the integration Edge Computing (EC), Vehicular Networks (VNs) are rapidly converging into a highly reliable, intelligent, and complex network system that serves vehicular users (VUs) with new services and applications [1]. However, Vehicular Edge Computing (VEC), enabled by the deployment of roadside units (RSUs) along road networks, started to suffer, mainly due to the increasing demand for high-quality services with stringent latency and data-rate requirements [2]. Furthermore, limited coverage, fixed positions with higher deployment costs, vulnerability to ground-based natural disasters, and data security threats are the main concerns of current terrestrial VEC systems [3].

Non-Terrestrial Networks (NTN) are considered a key enabler for the upcoming 6G systems and are expected to play an important role in improving the capacity and coverage of

traditional terrestrial networks [4]. In recent times, several networking platforms are populating space; based on their distance from the Earth's surface, these platforms can be classified into aerial- and space-based networking technologies. Aerial platforms include Low Altitude Platforms (LAPs), such as unmanned aerial vehicles (UAVs), air taxis, and helicopters, and High Altitude Platforms (HAPs), including airships, balloons, aircraft, etc. On the other hand, various satellite constellations can compose the space network, such as Low Earth Orbit (LEO), Medium Earth Orbit (MEO), and Geostationary Orbit (GEO) satellites. Aerial and space onboard computation and communication technologies can be exploited for potential EC services, creating a multi-layer EC environment. Additionally, NTN platforms can also act as relay nodes, to route the VUs computation load toward cloud facilities at ground to serve the users having limited or zero connectivity towards terrestrial networks. Therefore, the integration of NTN platforms into current VEC systems may be useful to serve VUs and their growing needs for new services [2], [5].

In multi-service vehicular settings, VUs have the capability to request various services from EC nodes and delegate part of their tasks to the chosen nodes via partial offloading procedures. The EC facilities mentioned above can have different characteristics in terms of mobility, node density, size, distance from VUs, etc. Therefore, it is important to select a suitable EC layer. Furthermore, size and energy restrictions often limit the ability of EC nodes placed in different networking layers. Due to these limitations, each node can provide only a limited set of services that can be exploited by VUs. Therefore, selecting a particular EC node to offload VU service data is an important problem to solve. Furthermore, due to the limited computing capabilities of each node, offloading a proper amount towards the selected Edge Node (EN) can improve performance and service quality [6]. This can be defined as a joint network selection and computation offloading problem, and solving such complex problems over a multi-EC, multi-service VN can be extremely challenging.

In recent years, Machine Learning (ML) has been introduced to solve complex wireless networking problems [7], [8]. Various intelligent solutions are available in the literature for different problems, such as routing, user assignments, resource allocations, etc. Innovative training methods, such as edge intelligence, have made the use of ML solutions in edge computing facilities increasingly common for effective user service. The considered joint network selection and computation offloading problem over a multi-layer, multi-service VN can be extremely challenging. Therefore, using an ML-based approach can be effective and futuristic given the demands of intelligent solutions in vehicular systems.

Among others, Reinforcement Learning (RL) has shown great potential to solve complex problems effectively, in particular in the case of terrestrial systems [9], and is a perfect candidate to solve the problem considered. Recently, RL has been further specialized in different methods, e.g., Multi-agent RL [10], hierarchical RL (HRL) [11], and distributed RL [12]. As discussed before, the considered network selection and offloading problem can effectively be solved through a multi-layer sequential decision-making process in terms of network layer selection, EC node selection, and offloading portion. Such multi-layer hierarchical processes, where a decision made at different layers can impact each other's performance, can be solved through HRL methods.

By taking this into account, we first present the Terrestrial and Non-Terrestrial (T-NT) EC-enabled vehicular network, serving the VUs with a set of services. With this, VUs can request services from different EC nodes from the T-NT layers. Next, we design a partial computation offloading process for processing the VU service requests by allowing them to offload the task data over the selected ENs. Since processing operations involve various communication and computation steps, proper latency and energy models are designed. Next, our objective is to minimize overall latency and energy costs by selecting the appropriate ENs and the amount of data that will be offloaded by mobile VUs in multiple EC-enabled VNs. We define the joint latency and energy minimization problem as a constrained optimization problem. The problem is modeled as a multi-layer sequential decision-making process through a Markov Decision Process (MDP). Next, a Deep Q Network (DQN) method is adapted to find appropriate layer selection, node selection, and offloading policies in HRL environments. Therefore, in this work, we demonstrate the use of HRL-based ML solutions over a complex multidimensional edge computing environment to satisfy VU demands. The results are compared with benchmark solutions to show the effectiveness of the proposed approach.

A. RELATED WORKS

VEC, which stands for EC-enabled VNs, has been thoroughly researched in the literature to deliver services that are both latency-sensitive and data-intensive to end users [9]. From a networking point of view, joint T-NT networks have gained huge popularity for serving VUs with improved capacity, coverage, and sustainability [13], [14]. In [15], the authors consider a space-air-ground integrated network for secure transmission from the perspective of the physical layer. A multi-point symbiotic security scheme through a digital twin-assisted multi-dimensional domain synergy precoding is proposed to ensure secure transmissions of multi-tier heterogeneous downlink communications in the considered 3D network architecture. Similarly, in [16], the authors further investigate the secure transmission scheme for cybertwin-enabled integrated satellite-terrestrial networks with the support of digital twin technology. The heterogeneous resources of terrestrial and non-terrestrial entities along with the link characteristics were the center of the performed study. Semi-definite relaxation and semi-definite programming are adopted for proposed beamforming optimization approaches to maximize the secrecy rate of satellite-to-vehicle links and the terrestrial BS-to-vehicle links. Also in [17], satellite beamforming and UAV power allocation are

jointly optimized to maximize the secrecy rate of the legitimate user within a target beam while guaranteeing the quality of service of users within other beams for the considered satellite-enabled vehicle communications. The UAV is used as a relay to improve the secure satellite-to-vehicle link and also to serve as a jammer by deliberately generating artificial noise to confuse intruders. Also, the use of Machine Learning (ML) to solve complex vehicular problems has gained noticeable popularity in recent years [1]. Among others, RL-based implementations are widely used to solve complex VN problems such as resource allocation, network selection, and service placement [9], [18].

The majority of VEC research can be categorized based on the problems they address. Often, network selection and computation offloading problems are solved independently. In some cases, they are considered jointly through some simplifications. Such studies are usually performed with the aim of optimizing latency, energy, reliability, or, in some cases, their combination. In the following, we survey some of the important studies related to VEC and joint optimization-related solutions from the recent past.

a: NETWORK SELECTION PROBLEM

In [19], the authors have proposed an on-line and off-policy learning algorithm based on multi-armed bandit theory to address the network selection problem in VEC environments. Also in [20], the authors have proposed a multi-hop mobility-aware task offloading mechanism with optimal EN selection for autonomous driving scenarios in VEC. In [21], the authors have proposed a multi-agent RL-based computation offloading strategy for vehicular scenarios. The latency performance of the offloading process is optimized with the binary computation offloading strategy. In such network selection optimization studies, offloading process decisions are limited to node selection only, without considering the optimization of partial offloading amounts. Offloading an imperfect amount of data to the selected optimal EN can still have impacts.

b: COMPUTATION OFFLOADING PROBLEM

In [22], the authors have proposed a federated learning-based framework to optimize the offloading parameter assuming the nearest node selection strategy. In [23], the authors have considered partial offloading in EC environments assisted by parked vehicles. Furthermore, in [24], the authors have proposed deep learning-based solutions for the Vehicle-to-Vehicle-assisted partial offloading problem in vehicular fog computing environments. However, such studies, where network selection decisions are based on a static/heuristic approach, cannot guarantee optimal performance.

c: JOINT SOLUTIONS

In [9], the authors have proposed collaborative RL-based strategies to solve the problem of joint network selection and computation offloading in a multi-service VEC environment. However, the study is limited to terrestrial VEC facilities. In [25], the authors have proposed a joint strategy for network selection and computation offloading in VEC environments with a single service. These joint studies are often limited to a single EC layer. In [26], the authors have considered a multi-tier EC scenario to solve the problem of resource allocation and offloading. However, the study is limited to terrestrial facilities.

d: HRL-BASED SOLUTIONS

In [11], authors have considered an HRL-based solution approach to reduce processing operations latency through joint optimization of the movements of mobile devices and the offloading parameters. The study is limited to minimizing latency with the binary offloading process. In [18], the authors have developed a dynamic resource allocation strategy to allocate edge computing resources in vehicular networks. The authors have combined HRL and meta-learning strategies to provide a solution that can adapt to a dynamically changing environment effectively. However, the study does not take into account the possibility of using multi-layered edge computing facilities with heterogeneous nodes along with the multi-service case. Recently in [27], the authors have proposed a novel strategy for joint service caching and offloading decisions in the edge computing scenario. The study is limited to the latency minimization problem without taking into account the energy cost. In addition, only terrestrial edge computing facilities are considered.

B. MOTIVATIONS

As mentioned above, several of the previous studies have independently considered network selection and computation offloading problems. In some cases, a joint network selection and offloading problem is considered with a single EC layer and/or a single service scenario. Furthermore, most studies have prioritized latency and energy cost on the device side only. With the presence of NTN platforms, the energy costs of EC layers become important. This shows that there are still some gaps in the VEC research, in particular for the case of network selection and offloading related studies, and further investigation is required. Additionally, with 6G on the horizon, multi-EC, multi-service vehicular scenarios require special attention. More advanced intelligent solutions are needed to enable intelligent vehicular networks. This motivates us to perform this study, in which we aim to solve a joint network selection and offloading problem over a multi-EC, multi-service enable T-NT vehicular scenario. The main contributions of this work are as follows:

- **Multi-layer Multi-service joint T-NTN:** We propose a multi-layer EC-enabled T-NTN for promoting VUs with various services. Ground-based VUs can request different services offered by RSUs, LAPs, HAPs, and LEO satellites through onboard EC facilities (Section II). The considered system model can have an advantage over terrestrial solutions in terms of coverage and capacity

boosts. Furthermore, compared to the NTN case, VUs can exploit nearby RSU resources. Such integrated T-NT architectures are expected to play a key role in the future generation of wireless technologies.

- **Joint latency and energy minimization problem:** A proper mathematical model is proposed that includes the latency and energy requirements of the different steps involved during the VU task processing operation. Both the VU and EN side latency/energy costs are considered when modeling the offloading process. In the end, an optimization problem is developed to minimize the overall latency and energy consumption by selecting a proper EC facility and the amount to be offloaded (Section II).
- **Multi-layer sequential decision-making process:** The joint network selection and computation offloading problem is modeled as a multi-layer sequential decision-making process through MDP, and an advanced DQN approach is considered to find optimal policies (Section III).
- **Performance Analysis:** In addition, a set of benchmark methods are used to analyze the results, showing the improved latency and energy performance of a proposed scheme (Section IV).

This work, in particular, advances the current literature by addressing the joint network selection and offloading problem over a multi-EC, multi-service vehicular scenario while taking into account several missing elements. In particular, the system model considered includes the joint energy costs from both the user and EN side, which is often neglected in the current literature. Additionally, we aim to minimize the latency and energy costs of vehicular data processing through the proper network selection and offloading over a multi-EC, multi-service vehicular scenario, which is in line with upcoming 6G technology where such multi-layer network architectures with abundant services will play a key role. Next, we propose the HRL-based solution for solving the problem considered using deep learning techniques. Such intelligent solutions can envision the future of intelligent transportation systems.

II. SYSTEM MODEL AND PROBLEM FORMULATION

The system is modeled as a multi-tiered EC facility to serve VUs with a set of services. Table 1 provides the important notations used in the following. We consider an integrated T-NTN, composed of a constellation of LEO satellites $\mathcal{S} = \{s_1, \ldots, s_q, \ldots, s_Q\}$ with Q satellites, a set $\mathcal{H} = \{h_1, \ldots, h_p, \ldots, h_P\}$ of P HAPs, a set $\mathcal{L} = \{l_1, \ldots, l_u, \ldots, l_U\}$ of U LAPs, a set $\mathcal{R} = \{r_1, \ldots, r_n, \ldots, r_N\}$ of N RSUs and a set of $\mathcal{V} = \{v_1, \ldots, v_k, \ldots, v_K\}$ of K VUs, randomly located in the area, allowing us to model a multi-layer EC-enabled vehicular scenario. Additionally, one Cloud Computing facility $\mathcal{C}$ is considered, located at the ground level. VUs can explore the EC facilities provided by different layers to enable various applications and services. The VN is modeled as a discrete time system in which the network parameters are supposed to be constant in each time interval τ, where τ_i identifies the ith time interval, i.e., $\tau_i = \{\forall t | t \in [i\tau, (i+1)\tau]\}$. To avoid additional complexity, we have considered that in each time instance, VUs can access services on the LEO satellite under visibility, while the whole constellation can be reached through proper inter-satellite links. Additionally, each EC layer can have access to the cloud facility through backhaul links. Figure 1 shows the various elements of the vehicular network scenario considered, consisting of one reference LEO satellite, HAPs, LAPs, RSUs, the cloud computing facility, and VUs.

The generic kth VU is characterized by a processing capability equal to $c_{v,k}$ Floating Point Operations per Second (FLOPS) per CPU cycle, while its CPU frequency is $f_{v,k}$. Each VU is supposed to be able to communicate using a bandwidth $B^{\mathrm{e}}_{v,k}$ with a reference EN $e \in (\mathcal{S} \cup \mathcal{H} \cup \mathcal{L} \cup \mathcal{R})$.

TABLE 1. List of main notations.

Notation	Description
$\mathcal{S}$, Q, s_q	LEO satellite constellation, No. of LEO Satellites, qth satellite
$\mathcal{H}$,P, h_p	HAP set, No. of HAPs, pth HAP
$\mathcal{L}$, U, l_u	LAP set, No. of UAVs, uth UAV
$\mathcal{R}$, N, r_n	RSU set, No. of RSUs, nth RSU
$\mathcal{V}$, K, v_k	VUs set, No. of VUs, kth VU
$\mathcal{C}$	Cloud Facility
Ξ, Z, ξ_z	Set of possible services, No. of services, zth service
τ_i	ith time interval
$c_{v,k}$, $f_{v,k}$	VUs FLOPs per sec. and CPU frequency
$B^{\mathrm{e}}_{v,k}$	VUs bandwidth wrt EN e
x_k	VUs task request
D_{x_k}	task size in Byte
Ω_{x_k}	CPU execution cycles requested by VU
$\bar{T}_{x_k}$	maximum latency of the requested service
$\bar{\Xi}_{x_k}$	type of service requested by the VU
c_e, f_e	ENs FLOPs per sec. and CPU frequency
B_e	ENs bandwidth
R_e	Coverage Radius of e
h_l, h_h and h_s	Altitude of UAV, HAP and Satellite nodes
$\vec{v}_{v,k}(\tau_i)$	VUs speed
$\vec{v}_{\min}$, $\vec{v}_{\max}$	Min. and Max. speed
μ, σ	mean and standard deviation of the VUs speed
$\mathrm{erf}(x)$	Gauss error function on x
$(x_{v_k}(\tau_i), y_{v_k}(\tau_i))$	kth VU location
(x_e, y_e)	eth EN projection on ground
R_e	Earths radius
$0 \le \alpha_{x_k}(\tau_i) \le 1$	offloading index
$a_{(k,e)}(\tau_i)$	VU-EN assignment index
$P_{c,\hat{l}}$	power used by the $\hat{l}$th device for the computation
$r_{\hat{k}\hat{l}}(\tau_i)$	data rate between $\hat{k}$ and $\hat{l}$
$\mathbf{A} = \{\alpha_{x_k}\}^M$	computation offloading matrix
γ_1, γ_2	weighting coeff. for balancing latency/energy cost
$\langle \mathcal{S}^p, \mathcal{A}^p \rangle$	State and action space
$\mathcal{R}^p$, $\hat{\mathcal{P}}^p$	Reward and state transition dynamics

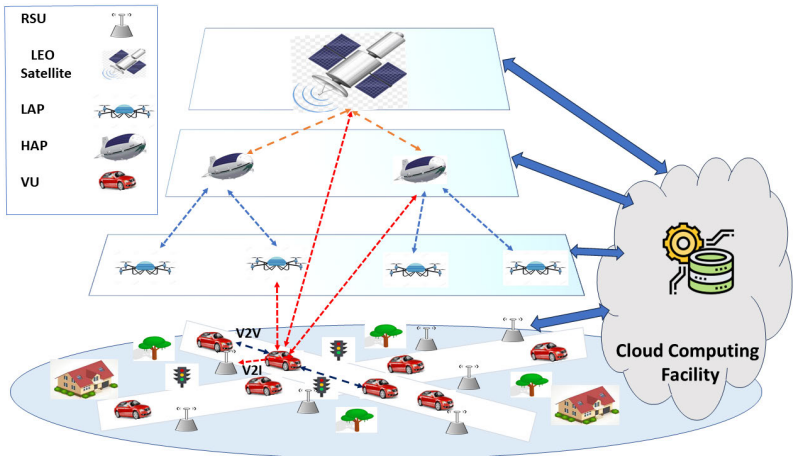

FIGURE 1. The reference Multi-Layer NTN edge computing scenario.

At each interval, the kth VU is supposed to generate a task request x_k to be processed, where the task x_k is identified through the tuple $\langle D_{x_k}, \Omega_{x_k}, \bar{T}_{x_k}, \bar{\Xi}_{x_k} \rangle$, and D_{x_k} is the task size in Byte, Ω_{x_k} corresponds to the CPU execution cycles requested, $\bar{T}_{x_k}$ is the maximum latency of the requested service, and $\bar{\Xi}_{x_k}$ is the type of service requested by the VU.

The eth EN (i.e., one node among any RSU, LAP, HAP, LEO or Cloud)[1] is characterized by a processing capability equal to c_e FLOPS per CPU cycle, with CPU frequency f_e, and communication capabilities, supposed to be identified through a communication technology able to work on a bandwidth B_e and covering an area with radius R_e. The LAP, HAP, and satellite nodes are located at heights h_l, h_h, and h_s from the ground level, respectively. Each EN provides computation offloading services to VUs within its coverage area. VU tasks are characterized by the type of service, supposing that the system is capable of providing different services, where $\Xi = \{\xi_1, \ldots, \xi_z, \ldots, \xi_Z\}$ is the set of possible services and Z the maximum number of services. With limited storage capabilities, RSUs, LAPs, HAPs, and LEO satellites can provide a limited number of services. We consider that the eth EN can provide the service set $\Xi^e = \{\xi_1, \cdots \xi_z, \cdots, \xi_e^{max}\}$. Furthermore, the cloud facility $\mathcal{C}$ is capable of providing all the possible services requested by VUs. Figure 2 shows a possible multi-service vehicular scenario where RSUs, LAPs, HAPs, and LEO satellites provide a subset of services (i.e., 2, 2, 3, 4, respectively) while the cloud facility is able to provide all possible services (i.e., 6) requested by the VUs. The T-NTN architecture includes the possibility of having inter-layer communication through radio frequency channels [28]. In every scenario, the upper-layer nodes distribute channel resources to the lower-layer platforms within their coverage area to facilitate communication between them, such as UAV to HAP, HAP to LEO, etc. It should be noted that other communication modes, e.g., optical connectivity, could be possibles; however, such analysis is beyond the scope of this work. Furthermore, the coverage characteristics of each edge node, i.e., eth, are based on the coverage radius R_e defined before. We also assume that the UAV nodes hover over the service area with a relatively low speed to serve VUs [29], [30]. Based on requests from VUs, UAVs can move in different directions with optimal path planning, whose management is beyond the scope of this work.

[1] In the following, $e = \{r_n\}$, $e = \{l_u\}$, $e = \{h_p\}$, $e = \{s_q\}$ and $e = \mathcal{C}$ stands for the rth RSU, uth LAP, pth HAP, qth LEO satellite and the cloud facility.

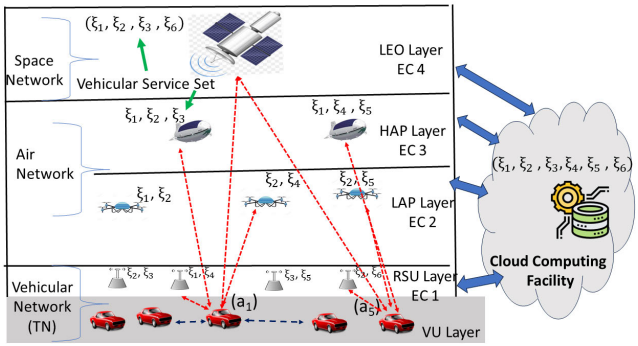

FIGURE 2. Multi-service vehicular network with non-terrestrial EC layers.

A. VUs MOBILITY AND DISTANCE MEASURES

In this work, we consider that VUs are moving with a variable speed $\vec{v}_{v,k}(\tau_i)$, whose value is bounded within $\vec{v}_{\min}$ and $\vec{v}_{\max}$ [31], while the kth VU instantaneous speed is modeled through a truncated normal distribution density function:

$$f\left(\vec{v}_{v,k}(\tau_i)\right) = \begin{cases} \dfrac{2e^{\frac{-\left(\vec{v}_{v,k}(\tau_i)-\mu\right)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}\left(\operatorname{erf}\left(\frac{\vec{v}_{\max}-\mu}{\sigma\sqrt{2}}\right)-\operatorname{erf}\left(\frac{\vec{v}_{\min}-\mu}{\sigma\sqrt{2}}\right)\right)}, \\ \qquad \vec{v}_{\min} \le \vec{v}_{v,k}(\tau_i) \le \vec{v}_{\max} \\ 0, \quad \text{else} \end{cases}$$

where μ and σ are the mean and standard deviation of the vehicles speed, and $\operatorname{erf}(x)$ is the Gauss error function over x. The path length within which the kth VU remains under the coverage of any terrestrial and aerial eth node

(i.e., RSU, UAV, HAP) is $D_{v_k,e}(\tau_i)$ and can be given by:

$$D_{v_k,e}(\tau_i) = \sqrt{R_e^2 - \left(y_e - y_{v_k}(\tau_i)\right)^2} \pm \left(x_e - x_{v_k}(\tau_i)\right)$$

where, $\left(x_{v_k}(\tau_i), y_{v_k}(\tau_i)\right)$ is the location of the kth VU at τ_i and (x_e, y_e) is the projection over the ground of a generic eth EC node, which can be RSU, UAV, HAP. Also, R_e, is the coverage radius of the edge node considered, and $\pm$ identifies the two possible directions taken by the mth VU. For simplicity, we have assumed that the VUs can travel on the road network along the coordinate x in either direction. For more details, the interested reader can look at [22]. The available sojourn time for the kth VU with respect to a generic eth node (i.e., RSU, LAP, or HAP) can be written as:

$$T^{\text{soj}}_{v_k,e}(\tau_i) = \frac{D_{v_k,e}(\tau_i)}{|\vec{v}_{v,k}(\tau_i)|} \qquad \forall i, v_k \tag{1}$$

It should be noted that despite in the literature there are several mobility models, the one considered here is often utilized in the vehicular edge computing context [32]. In fact, the main purpose of considering the mobility characteristics of VUs is to induce mobility constraints during the offloading process. Therefore, the following analysis can also be performed with other mobility models.

B. LEO SATELLITE MOBILITY AND DISTANCE MEASURES

In general, VUs can communicate with satellite nodes within a specific time interval, depending on the locations and mobility patterns of the LEO satellites. LEO satellites can move at very high speeds (i.e., a few kilometers per second) compared to VUs that move at a few meters per second. Here, we consider a well-known coverage model to find the arc length over which ground-based VUs can communicate [33]. Figure 3 visualizes the mobility of the LEO satellite with respect to the position of VUs on the Earth surface.

Consider a LEO satellite s_q located at height h_q with respect to the ground, moving at constant speed $\vec{v}_{s,q}$ km/s. Let us consider that at a certain instant τ_i its elevation angle with respect to the kth VU is $\theta_k(\tau_i)$, corresponding to a geocentric angle for the kth VU as,

$$\delta_{q,k}(\tau_i) = \arccos\left(\frac{R_e}{R_e + h_q} \cdot \cos\left(\theta_k(\tau_i)\right)\right) - \theta_k(\tau_i)$$

where R_e is the radius of the Earth. The total arc length over which the VU can communicate with the considered LEO satellite is defined as,

$$L_{q,k}(\tau_i) = 2(R_e + h_q) \cdot \delta_{q,k}(\tau_i)$$

With this, the total time within which the kth VU can be in the coverage range of the LEO is given as

$$T^{\text{soj}}_{v_k,q}(\tau_i) = \frac{L_{q,k}(\tau_i)}{\vec{v}_{s,q}} \tag{2}$$

To avoid incurring extra costs related to LEO satellite handovers, the kth VU must finish the offloading process within the limited coverage period. Given that satellite nodes move at much higher speeds than VUs, the VUs' mobility is disregarded when determining the satellite's coverage area.

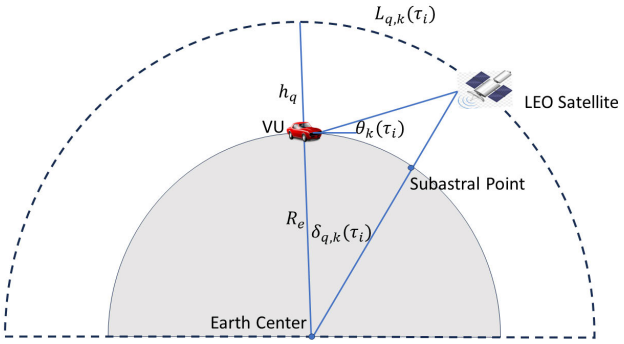

FIGURE 3. LEO satellite mobility model.

Eqs. (1) and (2) define the mobility constraint for the considered multi-EC vehicular networking system.

C. NETWORK SELECTION AND TASK OFFLOADING PROCESS

For each interval ith, the kth VU task is supposed to be managed through a partial computation offloading process, allowing a portion of the task to be offloaded toward the selected EN while the rest can be processed locally, thus allowing reducing the overall processing time and energy cost [22]. Here, we consider $0 \leq \alpha_{x_k}(\tau_i) \leq 1$ as an offloading index associated with the kth VU representing the portion of the task offloaded towards the selected EN, where $\alpha_{x_k}(\tau_i) = 1$ corresponds to a complete offloading, while $\alpha_{x_k}(\tau_i) = 0$ corresponds to perform only local processing. To avoid additional complexity, we assume that each VU can offload only towards one EN. We define a network selection variable, $a_{(k,e)}(\tau_i)$, indicating that the kth VU has selected the eth EN at τ_i for task offloading, where

$$\sum\nolimits_e a_{(k,e)}(\tau_i) \leq 1, \quad \forall i \tag{3}$$

corresponding to say that the kth VU can select at most one of the possible EN under visibility. The vehicular task processing operation with partial computation offloading involves several steps, such as the transmission of a selected task portion towards a selected EN, task computation operation at EN, reception of task back at VU, local computation of remaining task. Here, we present the generic task computation and communication models in the vehicular scenario, which are later used to model the overall task computation latency and energy.

D. TASK COMPUTATION MODEL

The time and energy required for task computation on any $\hat{l}$th device for a generic task $x_{\hat{k}}$ can be written as:

$$T^{x_{\hat{k}}}_{c,\hat{l}} = \frac{\Omega_{x_{\hat{k}}}}{c^{x_{\hat{k}}}_{\hat{l}} f^{x_{\hat{k}}}_{\hat{l}}}, \quad E^{x_{\hat{k}}}_{c,\hat{l}} = T^{x_{\hat{k}}}_{c,\hat{l}} P_{c,\hat{l}} \tag{4}$$

where $c_{\hat{l}}^{x_{\hat{k}}}$ and $f_{\hat{l}}^{x_{\hat{k}}}$ are the number of FLOPS per CPU-cycle and CPU-frequency allocated to the task $x_{\hat{k}}$, respectively, whether $\hat{l}$ identifies a VU (v_k), a RSU (r_n), a LAP (l_u), a HAP (h_p) or a LEO satellite (s_q). We have considered that the node capacity will be equally shared by all the tasks assigned to it. Also, in (4), $P_{c,\hat{l}}$ is the power used by the generic $\hat{l}$th device for the task computation phase.

E. TASK COMMUNICATION MODEL

The partial computation offloading process considers the possibility to split the tasks into sub-portions and offload only part of the task, while the rest is locally processed at the originating device; this means that both transmission of the VUs data portion towards the selected EN and the reception back of the results from EN should be considered. The total time and energy consumed during the transmission of data from a generic node $\hat{k}$ to another node $\hat{l}$ for task $x_{\hat{k}}$ are given by[2]:

$$T_{tx,\hat{k}\hat{l}}^{x_{\hat{k}}}(\tau_i) = \frac{D_{x_{\hat{k}}}}{r_{\hat{k}\hat{l}}(\tau_i)}, \qquad E_{tx,\hat{k}\hat{l}}^{x_{\hat{k}}}(\tau_i) = T_{tx,\hat{k}\hat{l}}^{x_{\hat{k}}}(\tau_i) Pt_{\hat{k}}$$

where $r_{\hat{k}\hat{l}}(\tau_i)$ is data-rate of the link between the two nodes Similarly, the time and energy required to receive the task of size $D_{x_{\hat{k}}}^r$ from $\hat{l}$th EN to $\hat{k}$ are:

$$T_{rx,\hat{l}\hat{k}}^{x_{\hat{k}}}(\tau_i) = \frac{D_{x_{\hat{k}}}^r}{r_{\hat{k}\hat{l}}(\tau_i)}, \qquad E_{rx,\hat{l}\hat{k}}^{x_{\hat{k}}}(\tau_i) = T_{rx,\hat{l}\hat{k}}^{x_{\hat{k}}}(\tau_i) Pr_{\hat{k}}$$

where $Pr_{\hat{k}}$ is the power consumed for receiving data. In addition, a symmetric channel model is assumed between $\hat{k}$ and $\hat{l}$. The channel between $\hat{k}$ and $\hat{l}$ in the i th interval is characterized by the link gain, modeled as [34]:

$$h_{\hat{k},\hat{l}}(\tau_i) = \beta_0 \cdot d_{\hat{k},\hat{l}}^{\theta}(\tau_i)$$

where, $d_{\hat{k},\hat{l}}(\tau_i)$ is the distance between node $\hat{k}$ and $\hat{l}$ in the ith interval, β_0 is the channel power gain at the reference distance of 1 m, while θ is the path loss coefficient over different communication links. The expression for the channel transmission rate is based on the Shannon capacity formula and can be written as:

$$r_{\hat{k}\hat{l}}(\tau_i) = b_{\hat{l}}^{x_{\hat{k}}}(\tau_i) \log_2 \left(1 + \frac{Pt_{\hat{k}} \cdot h_{\hat{k},\hat{l}}(\tau_i)}{N_0}\right) \quad \forall \hat{k}, \hat{l}$$

where $Pt_{\hat{k}}$ is the transmission power of a node $\hat{k}$, $b_{\hat{l}}^{x_{\hat{k}}}(\tau_i)$ is the communication bandwidth, and $N_0 = N_T b_{\hat{l}}^{x_{\hat{k}}}(\tau_i)$ is the thermal noise power with the noise spectral density defined as N_T.

F. VUs TASK PROCESSING

The VUs task processing operation through the use of edge facilities includes the task offloading phase and the local computing process. In the following, we detail this process.

[2]Here $\hat{l}$ and $\hat{k}$ are the indexes of any generic node among v_k, r_n, l_u, h_p, and s_q.

1) OFFLOADING PROCESS

During the partial computation offloading process, if the kth VU selects the eth EN, the time and energy required to offload the task to e and to get back the result at v_k in the ith interval are:

$$\begin{aligned}
T_{k,e}^{\text{off}}(\alpha_{x_k}(\tau_i)) &= \alpha_{x_k}(\tau_i) \Big(T_{tx,ke}^{x_k} + T_{w,e}^{x_k} + (1 - b_{\bar{\Xi}_{x_k}}^{e}) T_{c,e}^{x_k} \\
&\quad + T_{rx,ek}^{x_k} + b_{\bar{\Xi}_{x_k}}^{e} \left(T_{tx,e\mathcal{C}}^{x_k} + T_{w,\mathcal{C}}^{x_k} + T_{c,\mathcal{C}}^{x_k} + T_{rx,\mathcal{C}k}^{x_k}\right)\Big) \\
E_{k}^{\text{off}}(\alpha_{x_k}(\tau_i)) &= \alpha_{x_k}(\tau_i) \left(E_{tx,ke}^{x_k} + E_{rx,ek}^{x_k}\right) \\
E_{e}^{\text{off}}(\alpha_{x_k}(\tau_i)) &= \alpha_{x_k}(\tau_i) \Big(E_{w,e}^{x_k} + (1 - b_{\bar{\Xi}_{x_k}}^{e}) E_{c,e}^{x_k} \\
&\quad + b_{\bar{\Xi}_{x_k}}^{e} \left(E_{tx,e\mathcal{C}}^{x_k} + E_{rx,\mathcal{C}e}^{x_k}\right)\Big)
\end{aligned}$$

where $T_{tx,ke}^{x_k}$, $T_{w,e}^{x_k}$ $T_{c,e}^{x_k}$, and $T_{rx,ek}^{x_k}$ are the transmission time, waiting time at e to receive task data, computation time at e, and the receiving time for the task x_k generated by v_k during the offloading phase, and $E_{tx,ke}^{x_k}$ and $E_{rx,ek}^{x_k}$ are the energy consumed by the VU during the task transmission and result collection phases on the device, while $E_{c,e}^{x_k}$ and $E_{w,e}^{x_k}$ are the energy consumed by EN e during task computation and waiting phases, respectively. A binary variable $b_{\bar{\Xi}_{x_k}}^{e}$ is also considered, having value 1 if EN e cannot provide a requested service $\bar{\Xi}_{x_k}$, otherwise 0. In such a case, EN e transmits the user data to the cloud facility $\mathcal{C}$ for further computation. Therefore, additional latency and energy components are considered accordingly. From a latency perspective, the time required to transmit the data to cloud facilities, waiting time at $\mathcal{C}$, computation time, and the additional time required to receive back the computation results at e are considered. The energy cost is limited to the EN side costs only assuming that the cloud facilities have a stable energy supply from the electric grid. The energy required to transmit and receive back the data from the cloud facility is considered for cloud-based computations.

2) LOCAL COMPUTATION

The time and energy needed to perform the computation locally during the ith interval are:

$$\begin{aligned}
T_{k}^{\text{loc}}(\alpha_{x_k}(\tau_i)) &= \left(1 - \alpha_{x_k}(\tau_i)\right) T_{c,k}^{x_k} \\
E_{k}^{\text{loc}}(\alpha_{x_k}(\tau_i)) &= \left(1 - \alpha_{x_k}(\tau_i)\right) E_{c,k}^{x_k}
\end{aligned}$$

where $T_{c,k}^{x_k}$ and $E_{c,k}^{x_k}$ are the time and energy spent for the whole task x_k local processing, while $\alpha_{x_k}(\tau_i)$ is the portion of the task locally processed at the time interval τ_i.

3) PARTIAL OFFLOADING COMPUTATION

The delay and the energy consumed during the task processing phases, when partial offloading is performed in the ith interval, can be written as:

$$\begin{aligned}
&T^{x_k}(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)) \\
&\quad = \max\left\{T_{k,e}^{\text{off}}(\alpha_{x_k}(\tau_i)), T_{k}^{\text{loc}}(\alpha_{x_k}(\tau_i))\right\} \\
&E^{x_k}(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)) \\
&\quad = w_k \left(E_{k}^{\text{off}}(\alpha_{x_k}(\tau_i)) + E_{k}^{\text{loc}}(\alpha_{x_k}(\tau_i))\right) + w_e E_{e}^{\text{off}}(\alpha_{x_k}(\tau_i))
\end{aligned}$$

where w_k and w_e are the weighting coefficients associated with the energy costs of VUs and ENs, respectively. In addition, each vehicle should finish the offloading process and receive the results back within the sojourn time, hence:

$$T^{\text{off}}_{k,e}(\alpha_{x_k}(\tau_i)) \leq T^{\text{soj}}_{v_k,e}(\tau_i) \quad \forall i \tag{5}$$

G. PROBLEM FORMULATION

Main aim of this work is to optimize the network-wide performance of the multi-layer EC-enabled vehicular network. We aim to optimize the performance in terms of overall latency and energy consumed during the offloading process towards edge servers by selecting appropriate EN and offloading amounts. For this, we formulate the joint latency and energy minimization problem as follows:

$$\mathbf{P1}: \min_{\mathcal{A},\mathbf{A}} \left\{ \frac{1}{K} \sum_{k=1}^{K} \left[\gamma_1 T^{x_k}\left(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)\right) + \gamma_2 E^{x_k}\left(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)\right) \right] \right\} \forall i \tag{6}$$

$$s.t.\ \mathbf{C1}: \text{Eq. (3)} \tag{7a}$$

$$\mathbf{C2}: T^{x_k}\left(\alpha_{x_k}(\tau_i)\right) \leq T_{x_k} \quad \forall v_k \in \mathcal{V},\ \forall i \tag{7b}$$

$$\mathbf{C3}: \text{Eq. (5)} \tag{7c}$$

$$\mathbf{C4}: E^{x_k}(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)) \leq w_k E^{x_k}_{c,k} \tag{7d}$$

$$\mathbf{C5}: 0 \leq \gamma_1, \gamma_2 \leq 1;\ \gamma_1 + \gamma_2 = 1 \tag{7e}$$

where $\mathbf{A} = [\alpha_{x_k}]_{1\times M}$ is the computation offloading vector with the set of offloading parameters selected by M users, $\mathcal{A} = [a_{k,e}]_{(M\times|\mathcal{S}\cup\mathcal{H}\cup\mathcal{L}\cup\mathcal{R}|)}$ is the network selection matrix including the decisions made by M VUs to select the available EN and γ_1, γ_2 are two weighting coefficients for balancing latency and energy consumption. These values can be defined on the basis of the service latency demands and availability of energy resources. **C1** stands that each VU can select at most one EN for the computation offloading. **C2** puts a limit on the maximum processing time as one of the task requirements. According to **C3**, to avoid handover phenomena and related latency, each VN should complete the offloading process before it passes through the selected EN coverage. According to **C4**, the total energy cost of task processing during the partial computation offloading process should be bounded by the cost of energy needed to process the entire task locally. **C5** stands that the two weighting coefficients (γ_1, γ_2) should be between 0 and 1 with their sum equal to 1.

III. HIERARCHICAL REINFORCEMENT LEARNING SOLUTION

In this work, by finding the proper ENs and offloading portions, our goal is to minimize the overall latency and energy cost during the vehicular task processing operation over the multi-service multi-layer EC vehicular network. Given the complex nature of the considered problem, traditional heuristic and meta-heuristic approaches can have limited impacts, and more advanced solutions are required.

The formulated problem can be modeled as a sequential decision-making process through a proper MDP, and RL-based solution methods can be adapted to solve it. With the presence of multiple heterogeneous EC platforms with different properties, i.e., speed, location, services, resources, a traditional single agent-based RL solution can be computationally expensive and might not even be feasible. In recent times, several advanced RL methods have been introduced, especially to solve challenging problems in dynamic scenarios.

The considered problem can effectively be decomposed into multiple sub-problems impacting each other's performance and enabling a multi-layer decision-making process with reduced complexity. The HRL method can be adapted for such multi-layer decision-making process. Therefore, we have considered an HRL-based solution method for solving the network selection and offloading problem over multi-service multi-layer EC facilities.

P1 can be decomposed into three hierarchical learning processes, i.e., $\{\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3\}$. In the first process, $\mathcal{P}^1$, RL agent aims to select a proper EC layer j to perform the task computation. Based on the selected layer $j \in \mathcal{J}$, with $J = |\mathcal{J}|$ being a maximum number of edge layers available for offloading in an edge layer set $\mathcal{J}$, $\mathcal{P}^2$ aims to select a proper EC node e belonging to the jth layer for processing the kth VU data. Next, through $\mathcal{P}^3$, VU aims to select the appropriate amount of data α_{x_k} to be offloaded towards e to minimize the overall cost.

A. MDP MODELS

In general, the MDP for any problem p can be defined as a tuple $\langle \mathcal{S}^p, \mathcal{A}^p, \mathcal{R}^p, \hat{\mathcal{P}}^p, \gamma^p \rangle$, with state space $\mathcal{S}^p$, action space $\mathcal{A}^p$, reward $\mathcal{R}^p$, state transition probabilities $\hat{\mathcal{P}}^p$ and discount factor γ^p. By solving $\mathcal{P}^1$, we aim to find a proper EC layer to compute the user task. We define $\mathcal{S}^1 = \{s^1_1, \cdots, s^h_1, \cdots, s^H_1\}$ as a discrete state space associated with $\mathcal{P}^1$ with H states. Here, s^h_1 is the hth state function of the properties of the kth VU and jth EC layer, since the hth state of $\mathcal{P}^1$, based on VU k and layer $j \in J$ at time instance i is defined as,

$$s^h_1(\tau_i) = \{\bar{h}_{kj}(\tau_i), \bar{N}_j(\tau_i), \bar{s}_j(\tau_i), Pr_j^{\bar{\Xi}_{x_k}}(\tau_i)\}$$

where $\bar{h}_{kj}(\tau_i)$ is the distance measure, modeling the relative distance between layer j and VU k, $\bar{N}_j(\tau_i)$ is the average number of VUs selecting the layer j for processing their data, $\bar{s}_j(\tau_i)$ is the mobility state of ENs defined as the average speed of all ENs belonging to a particular layer j. In addition, $Pr_j^{\bar{\Xi}_{x_k}}(\tau_i)$ measures the probability that a service $\bar{\Xi}_{x_k}$ is present in the selected jth layer. Then, $\mathcal{A}^1 = \{a^1_1, \cdots, a^{\hat{h}}_1, \cdots, a^{\hat{H}}_1\}$ is the discrete action space for $\mathcal{P}^1$ with $\hat{H}$ actions, where the $\hat{h}$th action is defined as,

$$a^{\hat{h}}_1 = \{0, 1\}_{1\times J} \quad \text{with,} \quad \sum a^{\hat{h}}_1 \leq 1$$

where $a^{\hat{h}}_1$ is a binary vector modeling the VU layer selection decision where it can select at most one edge layer for

processing the data. The performance of $\mathcal{P}^1$ can be affected by the node selection and offloading policies of $\mathcal{P}^2$ and $\mathcal{P}^3$. Therefore, the reward received will be based on the policies adopted by these MDPs.

The subproblem $\mathcal{P}^2$ receives the information from the higher layer $\mathcal{P}^1$ in terms of the selected layer j and other state parameters. The aim is to select a suitable EN from the layer j to serve the VU. The node selection operation can be based on several parameters such as VU location, service demand, speed, layer properties, etc. Here, we introduce $\mathcal{S}^2 = \{s_2^1, \cdots, s_2^m, \cdots, s_2^M\}$ as a discrete-state space associated with $\mathcal{P}^2$ with M states, where s_2^m is the mth state based on the kth VU data and the properties of the jth EC layer selected in $\mathcal{P}^1$. The mth state associated with the VU k and jth layer node e in τ_i is defined as,

$$s_2^m(\tau_i) = \{\hat{N}_e, \hat{s}_{kj_e}, \hat{T}_{kj_e}^{soj}\}$$

where $\hat{N}_e$ is the resource state of eth node modeled as an average number of VUs requesting services from e, $\hat{s}_{kj_e}$ is a binary service state that takes the value 1 if the service requested by kth VU is available at e, otherwise 0. Then, $\hat{T}_{kj_e}^{soj}$ is the sojourn-time state, modeled as a binary variable that becomes 1 if the eth EN is able to cover the VU k with more than ζ of its coverage space, otherwise 0. Furthermore, $\mathcal{A}^2 = \{a_2^1, \cdots, a_2^{\hat{m}}, \cdots, a_2^{\hat{M}}\}$ is the discrete action space for $\mathcal{P}^2$ with $\hat{M}$ actions, where the $\hat{m}$th action is defined as,

$$a_2^{\hat{m}} = \{0, 1\}_{1\times \bar{E}_j} \quad \text{with} \quad \sum a_2^{\hat{m}} \leq 1,$$

where $\bar{E}_j$ is the maximum number of egde nodes from layer j that can cover any VU. Here, $a_2^{\hat{m}}$ is a binary vector that models the VUs node selection decision, where it can select at most one EN from a selected layer to process the data. The performance of $\mathcal{P}^2$ can be affected by the offloading policies of $\mathcal{P}^3$.

The subproblem $\mathcal{P}^3$ receives information from the higher layers on the selected layer j, the node e, and other state parameters. It aims to select a suitable offloading parameter α_{x_k} for offloading kth VU data. Here, we introduce $\mathcal{S}^3 = \{s_3^1, \cdots, s_3^l, \cdots, s_3^L\}$ as a discrete-state space associated with $\mathcal{P}^3$ with L states, where s_3^l is the lth state based on the VU data and the properties of the selected node e from the jth EC layer. The lth state at time instance τ_i is defined as

$$s_3^l = \left\{F_{k,e}^1(\tau_i), F_{k,e}^2(\tau_i), F_{k,e}^3(\tau_i)\right\}$$

where, $F_{k,e}^1(\tau_i)$, $F_{k,e}^2(\tau_i)$, and $F_{k,e}^3(\tau_i)$ are three binary functions, defined as:

$$F_{k,e}^1(\tau_i) = \begin{cases} 0 & T_{k,e}^{off}(\alpha_{x_k}(\tau_i)) \leq T_{v_k,e}^{soj}(\tau_i) \\ 1 & \text{else} \end{cases} \tag{7}$$

$$F_{k,e}^2(\tau_i) = \begin{cases} 0 & T^{x_k}\left(\alpha_{x_k}(\tau_i)\right) \leq T_{x_k} \\ 1 & \text{else} \end{cases} \tag{8}$$

$$F_{k,e}^3(\tau_i) = \begin{cases} 0 & E^{x_k}(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)) \leq w_k E_{c,k}^{x_k} \\ 1 & \text{else} \end{cases} \tag{9}$$

In addition, $\mathcal{A}^3 = \{a_3^1, \cdots, a_3^{\hat{l}}, \cdots, a_3^{\hat{L}}\}$ is the discrete action space for $\mathcal{P}^3$ with $\hat{L}$ actions, where the $\hat{l}$th action, $a_2^{\hat{l}} \in \{\alpha_{x_k}(\tau_i), \alpha_{x_k}(\tau_i) \pm \Lambda\}$ where $0 < \Lambda < 1$, is a step change in the amount to offload. For discretizing the action space we have considered a step change to the amount to be offloaded. This can be a consistent assumption while taking into account the vehicular applications constituted by several subtasks of a certain size. The partial offloading decision of each user allows one to process the tasks in parallel, i.e., through local and edge computing facilities, reducing the overall latency and energy requirements. However, the proper offloading portion should be selected based on the communication environment for such benefits.

We define a reward $\mathcal{R}^3$ for measuring the overall performance of network selection and offloading decisions given by:

$$\begin{aligned}\mathcal{R}^3(s_1^h, a_1^{\hat{h}}, s_2^m, a_2^{\hat{m}}, s_3^l, a_3^{\hat{l}}) \\ &= \gamma_1 T^{x_k}\left(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)\right) \\ &\quad + \gamma_2 E^{x_k}\left(\alpha_{x_k}(\tau_i), a_{(k,e)}(\tau_i)\right) \\ &\quad + w_1 F_{k,e}^1(\tau_i) + w_2 F_{k,e}^2(\tau_i) + w_3 F_{k,e}^3(\tau_i)\end{aligned}$$

where the first two elements measure the latency and energy performance, and the next three measure the performance in terms of service time, sojourn time, and energy constraints, respectively. If node selection policies violate the constraints, an additional weighted penalty value is added with positive weights w_1, w_2, and w_3.

B. DEEP Q NETWORK BASED SOLUTION

Given the complex and dynamic nature of the scenario considered, we consider Q learning as a model-free RL to find optimal policies. It is one of the highly explored model-free strategies for determining the optimal policy in unknown environments. The main objective is to demonstrate the effectiveness of the proposed HRL solution.

It should be noted that the proposed HRL framework is quite flexible and easily adapted to introduce new solution methods, such as Advantage Actor Critic (A3C), whose selection may affect performance. For example, Asynchronous A3C can allow asynchronous updates to neural network parameters through the different agents working independently; however, it can also introduce new complexities in terms of an asynchronous architecture and parallelized learning process compared to the simple DQN solution. The proposed DQN solution is better suited to a discretized action space. Therefore, we have considered the discritized action space with a step change in offloading values. Other DRL solutions, such as Deep Deterministic Policy Gradient (DDPG) and Twin-Delayed DDPG (TD3), may be considered to solve problems with a continuous action space. Such solutions can further improve offloading performance. However, they are complex to implement and may require a large number of evaluations for the convergence. Other RL solutions, such as the model-based method, can be more sample-efficient. However, their performance can be highly

dependent on the modeling errors. Given the complex and dynamic nature of the scenario considered, such solutions can be challenging to implement. Other RL solutions, such as policy gradient methods, can also be suited to continuous action spaces and provide direct policy optimization. However, they often require many training iterations with large sample sizes for convergence and can have unstable training. Therefore, although there are other advanced RL solution methods, to limit the complexity of the solution process, we consider a DQN-based approach. The proposed solution can be easily extended to more complex models in the future. In this case, the policy π_p for the problem p maps all states $s \in \mathcal{S}^p$ to action $a \in \mathcal{A}^p$. The Q-learning strategy is based on a state-action function, i.e., the Q function, defined as

$$Q^{\pi_p}(s', a') = R^p(s', a') + \gamma \sum_{\hat{s} \in \mathcal{S}^p} \hat{\mathcal{P}}^p_{s'\hat{s}}(a') V^{\pi_p}(\hat{s})$$

where $Q^{\pi_p}(s', a')$ is the state-action value function for policy π, representing a discounted cumulative reward from state s' when action a' is taken before following the policy π_p. $R^p(s', a')$ is the immediate reward received from the state action pair (s', a') for problem p. Also, the optimal Q-value can be represented as

$$Q^{\pi^*_p}(s', a') = R^p(s', a') + \gamma \sum_{\hat{s} \in \mathcal{S}^p} \hat{\mathcal{P}}^p_{s'\hat{s}}(a') V^{\pi^*_p}(\hat{s})$$

where $V^{\pi^*_p}(\hat{s}) = \min_{\hat{a} \in \mathcal{A}^p} Q^{\pi^*_p}(s', \hat{a})$ with optimal policy π^*_p. The Q values can be estimated through a recursive approach, where

$$Q_{t+1}(s', a') = Q_t(s', a') + \epsilon \cdot \left(r + \gamma \max_{\hat{a}} Q_t(s', \hat{a}) - Q_t(s', a') \right)$$

and ϵ is a learning rate. The Q function can be estimated through various traditional methods, such as the temporal difference approach, tabular methods, etc. In the case of complex RL problems with high-dimensional state/action spaces, the use of novel methods, such as function approximation, can be beneficial in terms of overall training complexity and generalization. The Q function can be estimated using a neural network-based function approximation technique with $Q(s', a'; \theta) \approx Q(s', a')$, where θ represents the weights of the neural network. During the training process, the values of θ can be adjusted to reduce the mean square error values.

Among several deep learning solutions based on neural networks, DQN is one of the most explored methods for estimating the RL agent policy in an unstable environment, mainly due to its simplicity. The DQN solution considered involves two networks (i.e., primary and target Q networks) for each layer to estimate the values of the Q function effectively. The primary network estimates the real/primary Q-value, whereas the target Q-values are estimated through the target network. The RL agent uses the backpropagation and gradient descent processes with a loss function based on the mean square error (MSE) to reduce the gap between the primary and target Q values, where the loss function is defined as:

$$L(\theta) = \mathbb{E}\left[\left(r + \gamma \max_{\hat{a}} Q_t(s', \hat{a}, \theta') - Q(s, a, \theta) \right)^2 \right] \quad (10)$$

and the primary values $Q(s, a, \theta)$ are based upon primary network parameters θ, and $r + \gamma \max_{\hat{a}} Q_t(s', \hat{a}, \theta')$ is the target Q value based upon the target network parameters θ'.

In the proposed HRL framework, three RL agents that exploit DQN are considered to find a proper coverage node selection policy for $\mathcal{P}^1$, $\mathcal{P}^2$, and $\mathcal{P}^3$ to determine the EC layer, the EC node, and the offloading amounts. The agent associated with $\mathcal{P}^1$ senses the state of the environment by processing the data associated with the demands of the users, the properties of the layers, etc., and selects the appropriate EC layer. The agent associated with $\mathcal{P}^2$ receives this information along with the current state information. The agent $\mathcal{P}^2$ then uses this information to select a suitable EN for data processing. The intrinsic reward $\mathcal{R}^2$ based on the feedback signal is used to update the EN selection policies, while the global reward $\mathcal{R}^1$ is considered while updating the $\mathcal{P}^1$ policies. The next agent for $\mathcal{P}^3$ receives information from $\mathcal{P}^2$ on the selected EN and its properties, which is used to define the offloading amount. The intrinsic reward $\mathcal{R}^3$ is used to update the policy of the $\mathcal{P}^3$ agent. Figure 4 details the interaction between different elements of the proposed HRL framework for network selection and offloading process.

Algorithm 1 details the DQN-based HRL process for the proposed network selection and offloading problem. The process begins with the initialization of the primary/target neural networks (w^P_H, w^T_H), (w^P_M, w^T_M), (w^P_L, w^T_L) associated with $\mathcal{P}^1$, $\mathcal{P}^2$ and $\mathcal{P}^3$ (lines 1-2). The neural networks associated with $\mathcal{P}^1$, $\mathcal{P}^2$, and $\mathcal{P}^3$ have, respectively, $\bar{L}^1$, $\bar{L}^2$, and $\bar{L}^3$ fully connected layers with $n_l, \forall l \in \bar{L}^1/\bar{L}^2/\bar{L}^3$, neurons. The training process lasts for $\bar{N}$ episodes (ε) with a maximum of $\mathcal{I}$ epochs per episode. Each training episode begins with the random initial state s_0 (lines 4-5). In each iteration ι, the DQNs are trained using the batch gradient descent approach described in Algorithm 2. Iteration begins by selecting a higher layer action $a^1_{\hat{h}}$ through the Epsilon Greedy Policy (EGP) with parameter e^1 (line 8). The information related to the selected edge layer and its properties is then passed to the next-layer DQN, i.e., node selection. Based on the selected layer and the state s^2_m action $a^2_{\hat{m}}$ is selected through EGP with the parameter e^2 (lines 10-11). The information associated with the selected EN is then exchanged with the DQN of the lower layer corresponding to $\mathcal{P}^3$ to define the offloading policy. Next, from state s^3_l, the action $a^3_{\hat{l}}$ is selected through EGP with parameter e^3 and the corresponding intrinsic reward $\mathcal{R}^3$ is determined (lines 12-15). The tuple $\langle s^3_l, a^3_{\hat{l}}, \mathcal{R}^3, s^3_{l,\text{new}} \rangle$ is then saved in replay memory $\mathcal{D}^3$ (line 16). The DQN function is then called to update the parameters of w^P_L, w^T_L (line 17), which is then used to determine the intrinsic reward $\mathcal{R}^2$ (line 18). After that, the DQN for $\mathcal{P}^2$ is updated using a gradient descent approach (lines 19-20). Next, a global

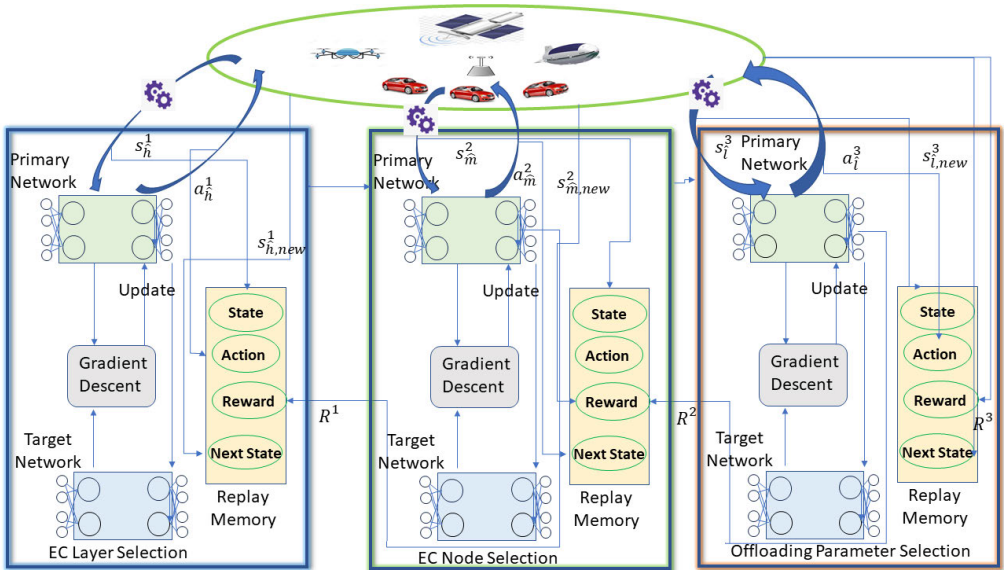

FIGURE 4. **Functional scheme for the proposed HRL solution.**

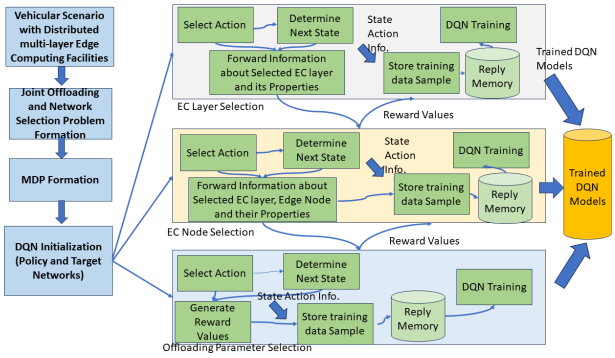

FIGURE 5. **DQN training process.**

reward $\mathcal{R}^1$ is defined (line 21). After that, the DQN for $\mathcal{P}^1$ is updated using a gradient descent approach (lines 22-23).

The DQN function defined in Algorithm 2 takes input as replay memory $\mathcal{D}^1/\mathcal{D}^2/\mathcal{D}^3$, batch size $\hat{k}^1/\hat{k}^2/\hat{k}^3$, learning rate $\epsilon^1/\epsilon^2/\epsilon^3$, and discount factors $\gamma^1/\gamma^2/\gamma^3$. The steps include random batch selection (lines 1-2), loss value generation (line 3), primary network parameter update through gradient descent step (line 4), and target network parameter update step (line 6).

Figure 5 includes the key steps used while defining the DQN models and their training process. The steps include the definition of joint network selection and the offloading problem over vehicular EC scenario, the definition of MDP models and their parameters, the initialization of the DQN process through the initialization of policy and target networks, and their training process. The interactive training process lasts for $\mathcal{I}$ iterations where each DQN model is updated according to the process defined in Algorithms 1 and 2.

The computation complexity of a single agent DQN solution can be defined as $\mathcal{O}(\mathcal{I} \cdot k \cdot (n_0 n_l + \sum_{l=1}^{L-1} n_l n_{l+1})/\bar{\mathcal{I}})$

Algorithm 1 HRL for Network Selection and Offloading

Input: $\mathcal{S}^1, \mathcal{A}^1, \mathcal{S}^2, \mathcal{A}^2, \mathcal{S}^3, \mathcal{A}^3, \mathcal{I}, \bar{N}, e^1, e^2, e^3, |\mathcal{D}^1|, |\mathcal{D}^2|, |\mathcal{D}^3|, \epsilon^1, \epsilon^2, \epsilon^3, \gamma^1, \gamma^2, \gamma^3, \bar{\iota}$

Output: w_H^P, w_M^P, w_L^P

1: Initialize w_H^P, w_M^P, w_L^P
2: Duplicate policy networks to Target Networks, i.e., $w_H^T = w_H^P$, $w_M^T = w_M^P$, $w_L^T = w_L^P$
3: **for all** $\varepsilon = 1, \ldots, N$ **do**
4: Select Random s_0
5: $s_h^1 \leftarrow s_0, \iota = 0$
6: **while** $\iota \neq \mathcal{I}$ **do**
7: $\iota = \iota + 1$
8: Select action $a_{\hat{h}}^1 \in \mathcal{A}^1$ with probability e^1
9: Determine next state $(s_{h,\text{new}}^1)$
10: Select s_m^2 based upon selected EC layer and local properties.
11: Select action $a_{\hat{m}}^2 \in \mathcal{A}^2$ with probability e^2
12: Determine next state $(s_{m,\text{new}}^2)$
13: Select s_l^3 based upon selected coverage node.
14: Select action $a_{\hat{l}}^3 \in \mathcal{A}^3$ with probability e^3
15: Find next state $s_{l,\text{new}}^3$ and reward R^3
16: Store $\mathcal{D}^3 \leftarrow \langle s_l^3, a_{\hat{l}}^3, R^3, s_{l,\text{new}}^3 \rangle$
17: w_L^P, w_L^T =DQN($\mathcal{D}^3, \hat{k}^3, w_L^P, w_L^T, \iota, \bar{\iota}, \epsilon^3, \gamma^3$)
18: Find Intrinsic Reward $\mathcal{R}^2$
19: Store $\mathcal{D}^2 \leftarrow (s_m^2, a_{\hat{m}}^2, R^2, s_{m,\text{new}}^2)$
20: w_M^P, w_M^T =DQN($\mathcal{D}^2, \hat{k}^2, w_M^P, w_M^T, \iota, \bar{\iota}, \epsilon^2, \gamma^2$)
21: Find Global Reward $\mathcal{R}^1$
22: Store $\mathcal{D}^1 \leftarrow (s_h^1, a_{\hat{h}}^1, R^1, s_{h,\text{new}}^1)$
23: w_H^P, w_H^T =DQN($\mathcal{D}^1, \hat{k}^1, w_H^P, w_H^T, \iota, \bar{\iota}, \epsilon^1, \gamma^1$)
24: **end while**
25: **end for**
26: **return** w_H^P, w_M^P, w_L^P

Algorithm 2 DQN Function

Input: $\mathcal{D}, k, w^p, w^T, i, \bar{i}, \epsilon, \gamma$
Output: $\{w^p, w^T\}$
function DQN($\mathcal{D}, k, w^p, w^T, i, \bar{i}, \epsilon, \gamma$)
  Select Random batch of k samples from $\mathcal{D}$
  Preprocess and pass the batch to w^p
  Find Loss between primary and Target Q values using (10)
  With gradient descent step update w^p
  Update w^T if $rem(i, \bar{i}) = 0$
end function
return $\{w^p, w^T\}$

where $\mathcal{I}$ is the maximum number of training iterations per episode, $\bar{\mathcal{I}}$ is the number of steps to update the parameters of the target network, k is the batch size, L is the maximum number of layers, and n_l for $l \in L$ are the number of neurons per layer [9]. Based on this, the complexity of the three-level HRL solution proposed in Algorithm 1 can be defined as $\mathcal{O}((k_1 \cdot (n_0 n_l + \sum_{l=1}^{\bar{L}_1 - 1} n_l n_{l+1}) + k_2 \cdot (n_0 n_l + \sum_{l=1}^{\bar{L}_2 - 1} n_l n_{l+1}) + k_3 \cdot (n_0 n_l + \sum_{l=1}^{\bar{L}_3 - 1} n_l n_{l+1})) \cdot \mathcal{I}/\bar{\imath})$. Based on the availability of computational resources and performance demands, the DQN parameters can be selected to have adequate complexity.

In Table 2, we have compared the proposed solutions with the recent HRL solutions considered in EC research. In [11], authors have considered a two-stage HRL approach for solving the UAV scheduling problem for providing EC facilities to mobile devices. The work aims to minimize the latency while considering the UAV energy constraints. Next, in [18], authors have considered HRL solutions for optimizing the resource allocation policies over dynamic vehicular networks. The study is limited to the single-edge computing facility. Also, it does not address the problem of energy minimization from the edge nodes side. More recently, in [27] authors have proposed HRL solutions for solving the Service Caching and Offloading problem over joint edge-cloud facilities. The study is limited to the single-edge computing layer and only addresses the latency minimization problem. Also, static node selection policies are adopted. Compared to these studies, our approach advances the literature by solving the joint network selection and offloading problem over multi-layer EC facilities to minimize the latency and energy costs from both the user and server sides.

C. BENCHMARK SOLUTIONS

Three different benchmark solutions are considered for comparison purposes.

a: RANDOM METHOD (RM): PROBABILISTIC VU-EN ASSIGNMENT

In this Random Method (RM) $\forall v_k \in \mathcal{V}$ we randomly select the node e. The probability of v_k selecting EN e is given by:

$$Pr\{a_{k,e}(\tau_i) = 1\} = \frac{1}{\mathcal{E}_k(\tau_i)} \tag{11}$$

Such random EN selection policies can have a limited impact in terms of task processing latency due to improper offloading policies. Such random solutions can be easily implemented without guaranteeing optimal load distribution and utilization of edge resources. $\mathcal{E}_k(\tau_i) = \{e | D_{v_k,e}(\tau_i) > 0, \quad \forall e\}$ is the set of ENs available for the kth VU to perform the offloading operation.

b: DISTANCE-BASED METHOD (DM): POSITION-BASED VU-EN ASSIGNMENTS

In this Distance-Based Method (DM), each nearby competing VU is assigned to the ENs based on the distance available before it passes through the ENs coverage range and the distance between VU and EN. Thus, $\forall v_k \in \mathcal{V}$:

$$a_{k,e}(\tau_i) = 1 \Leftrightarrow \frac{D_{v_k,e}(\tau_i)}{d_{k,e}(\tau_i)} = \max_{e \in \mathcal{E}_k(\tau_i)} \left\{ \frac{D_{v_k,e}(\tau_i)}{d_{k,e}(\tau_i)} \right\} \tag{12}$$

By using a static method, the computational complexity of the offloading process can be minimized with simpler techniques. Nonetheless, with these static policies, the use of distributed edge resources might be constrained, causing multiple VUs to opt for the same edge facilities while disregarding the local environmental parameters. This may lead to high computational and communication costs during the offloading process, particularly when there is a substantial number of VUs.

c: LOCAL COMPUTATION (LC)

In this case, VUs perform the task computation by themselves without employing any EN. Such an approach can reduce the latency requirements in terms of data transmission toward ENs. However, without the parallel computation process between the EC nodes and resource-limited VUs, the computation latency can grow significantly and cannot satisfy the service latency demands.

IV. NUMERICAL RESULTS

The proposed HRL-based solution and benchmark approaches are simulated in a Python environment with ML-related libraries, including NumPy, Pandas, Math, and Matplotlib.[3] A service area composed of a 5-km road network is considered, with randomly located VUs. A set of users between 200 and 2000 is considered with a probability that each VU is active equal to $P_a = 0.1$. VU can request up to six different services. The Cloud Computing facility is able to provide all six services, while LEO satellites are able to host four services. HAP nodes are equipped with three services, while RSUs and UAVs can provide two services each. Services are deployed randomly and in advance.

The number of ENs (i.e., RSUs, UAVs and HAP) is based on the total service area and the coverage radius of the ENs. We assume that the number of ENs can be estimated as $(\kappa \times A_d)/R_e$, where κ is a parameter that allows the ENs to have an overlap of their coverage radius, allowing users to potentially

[3]The simulation code is publicly available at https://github.com/swapnilshinde15/EC_HRL.git

TABLE 2. Comparison with recent literature.

HRL Solution	Edge Computing	Service Type	Device Mobility	Problem	Objective	HRL Solution
[11]	Single-layer UAV-assisted MEC	Single service	Can move away from its position	Edge Node Scheduling	Latency minimization with energy constraints	Two-stage HRL approach with DDPG and DQN
[18]	Generic Single layer MEC Facility	Multiple contents	Vehicular Scenario with Mobility	Resource Allocation in a dynamic vehicular scenario	Maximize the net revenue of the learning agent	N-stage HRL approach with Actor-Critic solutions
[28]	Generic Single layer MEC Facility with integrated cloud	Multi-service Case	Not Considered	Service Caching and Offloading with static network selection	Latency Minimization	2-stage HRL approach with Actor-Critic and DQN solutions
Our Solution	Multi-layer MEC with Joint T-NTN	Multi-service Case	Vehicular Scenario with Mobility	Joint Network Selection and Offloading	Joint Latency and Energy (both user and EC node side) minimization	3-stage HRL approach with DQN solutions

TABLE 3. Simulation parameters.

Coverage $(R_{r_n}, R_{l_u}, R_{h_p})$	(50, 200, 1000) m
VU Computation Cap. $(c_{v,k} \cdot f_{v,k})$	8 GFLOPS
RSU Computation Cap. $(c_{r_n} \cdot f_{r_n})$	20 GFLOPS
LAP Computation Cap. $(c_{l_u} \cdot f_{l_u})$	20 GFLOPS
HAP Computation Cap. $(c_{h_p} \cdot f_{h_p})$	40 GFLOPS
LEO Computation Cap. $(c_{s_1} \cdot f_{s_1})$	60 GFLOPS
Altitude (h_l, h_h, h_s)	(1, 100, 2000) Km
Bandwidth $(B_{r_n}, B_{l_u}, B_{h_p}, B_{s_1})$	(20, 20, 50, 100) MHz
VU Speed Range $(\vec{v}_{\min}, \vec{v}_{\max})$.	(8 m/s, 14 m/s)
VU Power $(P_{c,v_k}, Pt_{v_k}, Pr_{v_k})$	(1.3, 1.5, 1.3) W
Task Size (D_{x_k})	5 MB
Task Latency Req. $(\bar{T}_{x_k})$	4 Sec
Task Computation Req. (Ω_{x_k})	$(10^3 \cdot D_{x_k})$ Flops
Elevation angle (θ_m)	$\mathcal{U}(20^0, (\pi/2+20)^o)$
Weighting Coefficients (η_1, η_2, w_k)	(0.5, 0.5, 0.5)

connect to multiple ENs. We have set $\kappa = 1.5$. A_d is the length of the road considered and R_e is the coverage radius of the ENs. The position of each eth EN (i.e., RSU, UAV, and HAP) along the road is randomly determined as $x_e = x_{e-1} + 0.5 \times R_e + (1.5 \times R_e - 0.5 \times R_e + 1)U(0, 1)$. With this, the ENs are placed in an incremental order with random distance between them, and $U(0, 1)$ is a random value between 0 and 1.

For DQN simulation, primary and target networks with layers $(\bar{L}^1, \bar{L}^2, \bar{L}^3) = 4$ are considered with learning parameters $(e^1, e^2, e^3) = 0.7, (\mathcal{D}^1, \mathcal{D}^2, \mathcal{D}^3) = 4000, (\epsilon^1, \epsilon^2, \epsilon^3) = 0.05$, $(\gamma^1, \gamma^2) = 0.98$. Furthermore, the learning process includes $\bar{N} = 50$ with $\mathcal{I} = 10^3$ and $\bar{\iota} = 50$. The main simulation parameters are provided in Table 3. Performance results rely on the mean of 20 iterations to minimize random fluctuations.

A. LATENCY AND ENERGY COST

1) JOINT LATENCY AND ENERGY COST

In this work, our objective is to minimize the overall latency and energy cost associated with vehicle task processing operations at EC facilities. Figure 6 presents the average cost required for different methods. Cost values can be affected by layer/node selection and offloading decisions. Regarding the considered benchmark methods, RM randomly selects the layer/node for processing the VUs' complete task. With such a suboptimal approach, RM requires a much higher amount of latency and energy costs for processing the VUs data. The distance-based benchmark method, DM, chooses EN based on distance metrics. Nevertheless, this static method can restrict the flexibility of the offloading process in terms of utilizing edge resources. Moreover, given the multiple VUs, services, and edge facilities, it is necessary to adapt the VUs offloading policies. Due to these limitations, the DM approach incurs higher processing costs. The local computation case has a static cost since it is unable to take advantage of distributed computing environments, resulting in much higher latency costs. The proposed HRL method can reduce the overall cost through a proper layer/node selection and offloading process. With the help of local environment data, HRL policies can be adapted to enable efficient edge processing operations.

DQN-based HRL agents can properly explore the simulation environment and define optimal policies based on resource availability, location parameters, mobility, and task requirements. The HRL policies that we considered address different objectives by considering the combined costs of latency, energy, and weighting coefficients. In particular, γ_1, γ_2 values are updated to prioritize latency and energy

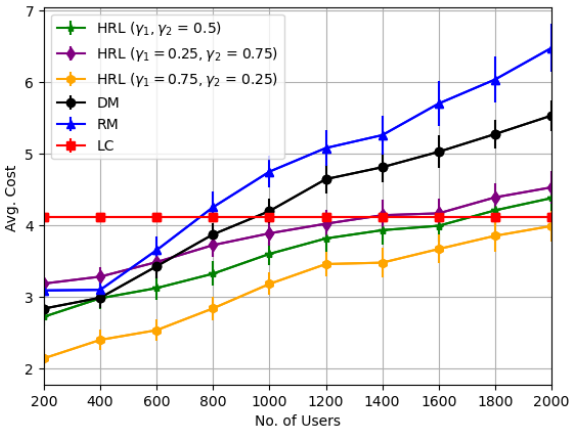

FIGURE 6. Joint latency and energy cost.

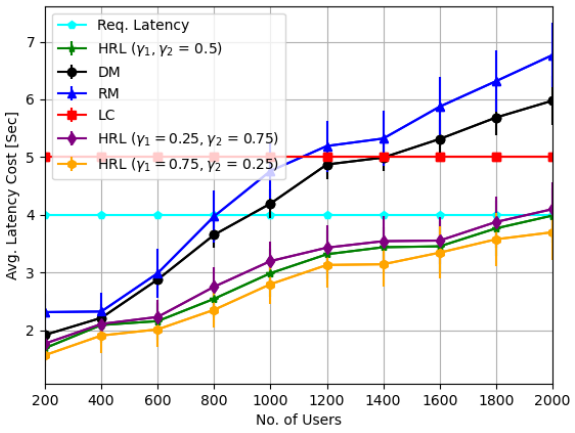

FIGURE 7. Total task processing latency.

costs during the optimization phases. It can be seen that the energy cost on a higher scale dominates the optimization process, where if the weight associated with the energy, i.e., γ_2 is lower, the overall cost is reduced. However, this does not imply that the energy cost is optimized. The overall energy cost is slightly higher because of the reduced weights associated with the energy values. By adjusting the weight coefficients, different types of systems such as latency-critical, energy-efficient, balanced, and others can be developed. Moreover, optimizing the values of γ_1 and γ_2 to achieve the optimal system is a task that goes beyond the scope of this study and could be addressed in future research. Additionally, the performance outcomes include error bars that represent the standard deviation values for each scenario.

2) AVERAGE LATENCY COST

Given the latency-constraint nature of vehicular scenarios, it is important to analyze the performance of the proposed HRL approach in terms of latency requirements. In Figure 7, we present the latency performance for different solution methods, highlighting the performance gain for the proposed HRL case. It is evident that using suboptimal node/layer selection policies, RM and DM methods incur higher latency costs. As the number of VUs increases, the performance of both DM and RM deteriorates, becoming worse than that of the LC approach. The LC method does not meet the service latency requirements for VUs. In contrast, the proposed HRL approach substantially reduces latency cost by employing an efficient offloading process.

By selecting the proper edge computing layer, corresponding node, and offloading portions, the proposed HRL solution can optimize latency costs by reducing data communication and computation costs. The performance of HRL solutions is also compared with different weight coefficients to enable latency-critical/energy-efficient solutions. It can be seen that for the case of latency-critical applications, with increasing γ_1 values, the latency cost can be reduced further by prioritizing it. Furthermore, the standard deviation of the data collected in 20 iterations is displayed using error bars, which illustrate the variations in the cost values from the mean.

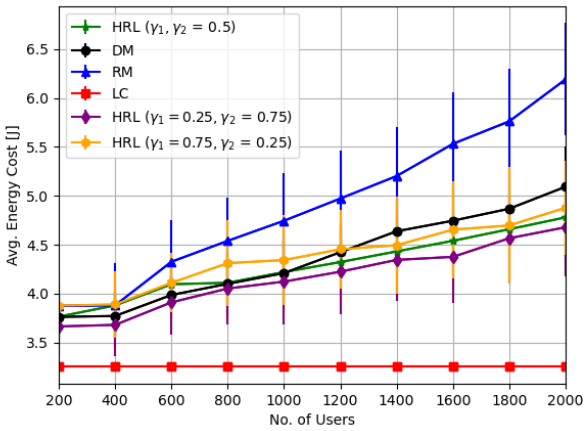

FIGURE 8. Energy requirements.

3) AVERAGE ENERGY COST

With the presence of dynamic VUs and different non-terrestrial platforms, analyzing the energy cost becomes important. In Figure 8, we present the energy requirements for different methods as a weighted average of the energy elements on the VU side and on the energy side. For the LC approach, the overall energy requirements can be reduced because of the local computation process. However, as previously presented, it suffers from unfeasible computation latency due to reduced computation resources. With long-distance transmissions and improper offloading parameters, RM and DM approaches suffer from higher energy requirements, especially with the growing number of users. The HRL approach requires higher energy costs compared to the LC method with additional data transmission/reception and waiting steps. However, it can gain an advantage in terms of latency and handover requirements. Furthermore, by increasing γ_2 (i.e., the weight coefficient associated with the energy cost), further reduction of energy costs can be achieved in expenses of slightly higher latencies. Such an adaptability of solutions can be useful for future generations of wireless systems to satisfy diverse user requirements.

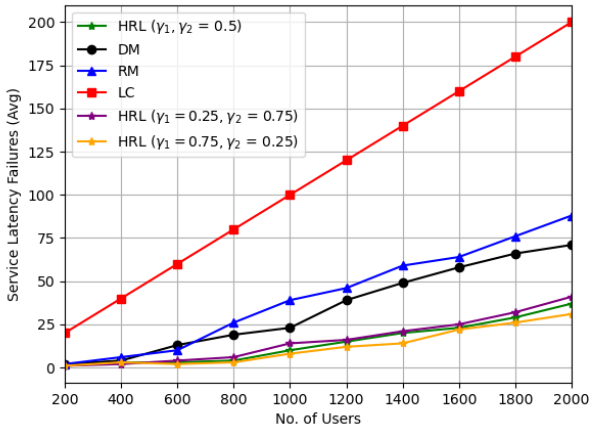

FIGURE 9. **Service time failures.**

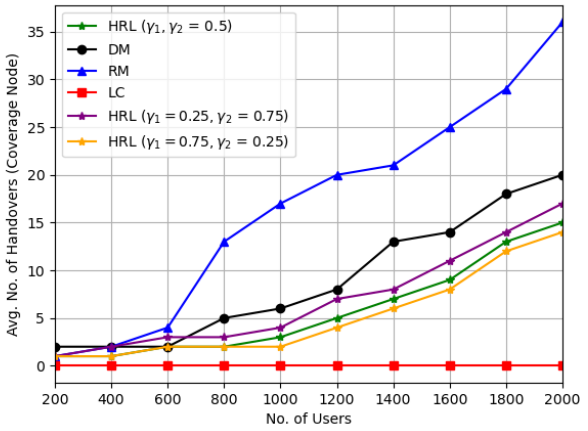

FIGURE 10. **Sojourn time failures.**

B. RELIABILITY ANALYSIS

After measuring the performance in terms of different cost elements, in the following, we measure the reliability of the proposed solution methods. In particular, given the crucial nature of the reliability aspects of the vehicular scenario, the reliability performance of the solutions often becomes a critical benchmark. In the following three subsections, we measure the reliability performance in terms of the ability of the proposed solutions to follow the constraints in terms of service time, vehicular mobility, and service placements.

1) AVERAGE NUMBER OF SERVICE TIME FAILURES

Apart from the latency and energy costs associated with the offloading process, it is also important to analyze the performance of the proposed solution in terms of service latency requirements. In Figure 9, we present the performance of different methods in terms of satisfaction of the service latency constraint defined in (7b). With the complete local process, the LC approach cannot satisfy the service latency requirements. Similarly, the other two benchmark solutions (RM, DM) with static offloading processes have a large number of service latency failures. By including distance measures when selecting ENs, the DM solution can reduce the number of failures compared to the RM approach. However, with static offloading solutions and a growing number of VUs, overall failures can still be much higher. The proposed HRL solution, on the other hand, can enable efficient task processing based on VU's local environments and task requirements, thus reducing the number of service latency failures. This shows the importance of the proposed HRL solution in enabling latency-critical vehicular services.

2) AVERAGE NUMBER OF SOJOURN TIME FAILURES

With the presence of dynamic VUs and non-terrestrial nodes, it is important to analyze the performance of the proposed solution in terms of a mobility constraint defined in (5). With the mobility constraint, the offloading process should be completed before the VUs pass through the coverage area of a selected EN. Thus, performance can be affected by the selected layer/node and the offloading portions. As shown in Figure 10, with the random node selection approach, the RM method suffers from many failures due to mobility constraints. With a static distance-based approach, the DM method can have a reduced number of failures; however, the performance can be suboptimal. The proposed HRL solution can significantly reduce mobility constraint failures with proper offloading decisions in terms of offloading node and amount selections.

3) AVERAGE NUMBER OF SERVICE HANDOVERS REQUIRED

For the considered multi-service vehicular scenario, it is important to analyze the performance in terms of a number of service handover requirements. If the selected EN cannot provide the requested service, it needs to take additional measures to satisfy the demands of the users. In the considered simulation, ENs that are unable to provide the demanded service relay the user data to the cloud facility, having all the services pre-installed. In Figure 11, we present the performance in terms of such service handover requirements, for different methods. The proposed HRL methods with different weight coefficients can have a superior performance compared to the other solutions with static or random node selection strategies.

Here, we have analyzed the service handover in which the selected edge node is unable to provide the service requested by the user. In this case, several possibilities can arise that are beyond the scope of this work. For example, user data should be routed to the other nearby edge node for processing, which introduces additional handover costs. Otherwise, the centralized service manager could update the service deployments to allow the edge node to provide the requested service, introducing additional service placement costs. Otherwise, the edge node can drop a user, causing poor reliability of the solutions. Exploring such options for optimal behavior can be one possible future direction for this work.

C. JOINT COST Vs TRAINING EPISODES

HRL training performance can be measured in terms of various parameters to analyze the complexity of the training

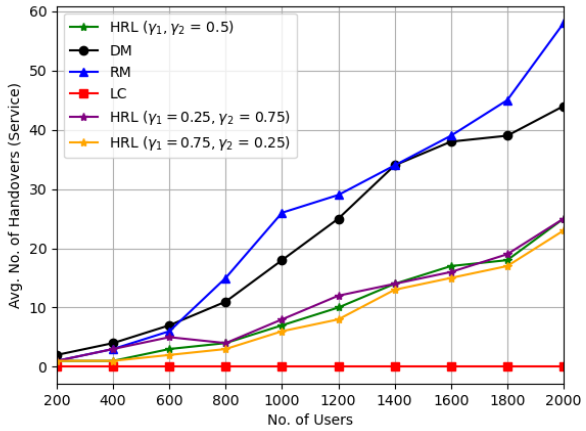

FIGURE 11. Service handover requirements.

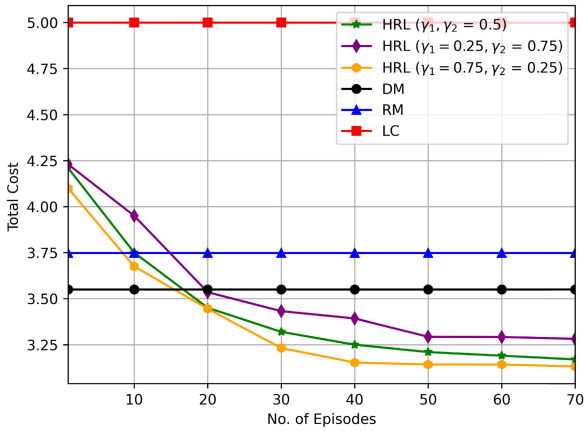

FIGURE 12. HRL training perfromance.

process and the convergence to stable models. In Figure 12, we present HRL training performance in training episodes. Performance is measured in terms of total cost and can be seen improving for increasing number of training iterations. Over time, with a sufficient number of training episodes (i.e., > 50), the performance becomes more stable by converging to a cost value that is much lower compared to the other benchmark solutions. For different weighting coefficients (i.e., γ), HRL training performance slightly differs. In particular, the HRL solution with ($\gamma_1 = 0.75, \gamma_2 = 0.25$) induces a lower weight for energy costs, resulting in slightly improved performance. On the other hand, the solution ($\gamma_1 = 0.25, \gamma_2 = 0.75$) has slightly increased cost values mainly due to the dominant energy factor in the cost function.

V. CONCLUSION

In this work, we have considered a joint network selection and computation offloading problem in a joint T-NTN environment to process vehicular data. A multi-service vehicular scenario is considered that allows VUs to request different services through multiple EC environments. A constrained optimization problem is formed to minimize overall latency and energy costs through proper network selection and offloading decisions. The problem considered is modeled as a multi-layer sequential decision process, and the HRL approach is used to solve it. In particular, deep learning-based strategies are applied to find optimal network selection and offloading policies. The performance of a proposed method is analyzed using Python-based solutions and compared with several other benchmark solutions. The current study includes the possibility of the multi-service vehicular scenario with different edge computing facilities. The simulation is limited to the single LEO satellite-based scenario and can be further extended towards the full LEO constellation with a possible exploration of satellite resources through inter-satellite links. In this work, we have proposed a DQN-based solution method. There are several other RL solution methods available with the capability to solve problems with continuous action space, direct policy optimization, model-based solutions, etc. However, such solutions require more complex implementations, a large number of training samples, and iterations for convergence, and can have an unstable training process. In the future, we plan to extend the proposed DQN framework with more advanced RL solutions by properly investigating complexity issues, training process requirements, and the corresponding performance boost. Finally, the considered approach can be extended in the future to jointly solve the problem of service placement, network selection, and offloading effectively.

REFERENCES

[1] F. Tang, B. Mao, N. Kato, and G. Gui, "Comprehensive survey on machine learning in vehicular network: Technology, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 2027–2057, 3rd Quart., 2021.

[2] S. S. Shinde and D. Tarchi, "Towards a novel air–ground intelligent platform for vehicular networks: Technologies, scenarios, and challenges," *Smart Cities*, vol. 4, no. 4, pp. 1469–1495, Dec. 2021.

[3] F. Zhou, R. Q. Hu, Z. Li, and Y. Wang, "Mobile edge computing in unmanned aerial vehicle networks," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 140–146, Feb. 2020.

[4] M. Giordani and M. Zorzi, "Non-terrestrial networks in the 6G era: Challenges and opportunities," *IEEE Netw.*, vol. 35, no. 2, pp. 244–251, Mar. 2021.

[5] A. Traspadini, M. Giordani, and M. Zorzi, "UAV/HAP-assisted vehicular edge computing in 6G: Where and what to offload?" in *Proc. Joint Eur. Conf. Netw. Commun. 6G Summit (EuCNC/6G Summit)*, Grenoble, France, Jun. 2022, pp. 178–183.

[6] J. Sun, Q. Gu, T. Zheng, P. Dong, A. Valera, and Y. Qin, "Joint optimization of computation offloading and task scheduling in vehicular edge computing networks," *IEEE Access*, vol. 8, pp. 10466–10477, 2020.

[7] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of machine learning in wireless networks: Key techniques and open issues," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3072–3108, 4th Quart., 2019.

[8] Y. Shi et al., "Machine learning for large-scale optimization in 6G wireless networks," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2088–2132, 4th Quart., 2023.

[9] S. S. Shinde and D. Tarchi, "Collaborative reinforcement learning for multi-service Internet of Vehicles," *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2589–2602, Feb. 2023.

[10] S. Hwang, H. Kim, H. Lee, and I. Lee, "Multi-agent deep reinforcement learning for distributed resource management in wirelessly powered communication networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 14055–14060, Nov. 2020.

[11] T. Ren et al., "Enabling efficient scheduling in large-scale UAV-assisted mobile-edge computing via hierarchical reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7095–7109, May 2022.
[12] S. Liu, C. Zheng, Y. Huang, and T. Q. S. Quek, "Distributed reinforcement learning for privacy-preserving dynamic edge caching," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 749–760, Mar. 2022.
[13] D. Han et al., "Two-timescale learning-based task offloading for remote IoT in integrated satellite-terrestrial networks," *IEEE Internet Things J.*, vol. 10, no. 12, pp. 10131–10145, Jun. 2023.
[14] S. S. Shinde and D. Tarchi, "Network selection and computation offloading in non-terrestrial network edge computing environments for vehicular applications," in *Proc. 11th Adv. Satell. Multimedia Syst. Conf. 17th Signal Process. Space Commun. Workshop (ASMS/SPSC)*, Graz, Austria, Sep. 2022, pp. 1–8.
[15] Z. Yin, N. Cheng, T. H. Luan, Y. Song, and W. Wang, "DT-assisted multi-point symbiotic security in space-air-ground integrated networks," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 5721–5734, 2023.
[16] Z. Yin, N. Cheng, T. H. Luan, and P. Wang, "Physical layer security in cybertwin-enabled integrated satellite-terrestrial vehicle networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 4561–4572, May 2022.
[17] Z. Yin et al., "UAV-assisted physical layer security in multi-beam satellite-enabled vehicle communications," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2739–2751, Mar. 2022.
[18] Y. He, Y. Wang, Q. Lin, and J. Li, "Meta-hierarchical reinforcement learning (MHRL)-based dynamic resource allocation for dynamic vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 3495–3506, Apr. 2022.
[19] A. Bozorgchenani, S. Maghsudi, D. Tarchi, and E. Hossain, "Computation offloading in heterogeneous vehicular edge networks: On-line and off-policy bandit solutions," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4233–4248, Dec. 2022.
[20] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2169–2182, Feb. 2023.
[21] X. Zhu, Y. Luo, A. Liu, M. Z. A. Bhuiyan, and S. Zhang, "Multiagent deep reinforcement learning for vehicular computation offloading in IoT," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9763–9773, Jun. 2021.
[22] S. S. Shinde, A. Bozorgchenani, D. Tarchi, and Q. Ni, "On the design of federated learning in latency and energy constrained computation offloading operations in vehicular edge computing systems," *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 2041–2057, Feb. 2022.
[23] X.-Q. Pham, T. Huynh-The, E.-N. Huh, and D.-S. Kim, "Partial computation offloading in parked vehicle-assisted multi-access edge computing: A game-theoretic approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 10220–10225, Sep. 2022.
[24] J. Shi, J. Du, J. Wang, and J. Yuan, "Deep reinforcement learning-based V2V partial computation offloading in vehicular fog computing," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Nanjing, China, Mar. 2021, pp. 1–6.
[25] S. S. Shinde and D. Tarchi, "A Markov decision process solution for energy-saving network selection and computation offloading in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 12031–12046, Sep. 2023.
[26] W. Feng, S. Lin, N. Zhang, G. Wang, B. Ai, and L. Cai, "Joint C-V2X based offloading and resource allocation in multi-tier vehicular edge computing system," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 432–445, Feb. 2023.
[27] C. Sun, X. Li, C. Wang, Q. He, X. Wang, and V. C. M. Leung, "Hierarchical deep reinforcement learning for joint service caching and computation offloading in mobile edge-cloud computing," *IEEE Trans. Services Comput.*, early access, Jan. 19, 2024, doi: 10.1109/TSC.2024.3355937.
[28] H. Yu, T. Taleb, K. Samdanis, and J. Song, "Toward supporting holographic services over deterministic 6G integrated terrestrial and non-terrestrial networks," *IEEE Netw.*, vol. 38, no. 1, pp. 262–271, Jan. 2024.
[29] Z. Wu, Z. Yang, C. Yang, J. Lin, Y. Liu, and X. Chen, "Joint deployment and trajectory optimization in UAV-assisted vehicular edge computing networks," *J. Commun. Netw.*, vol. 24, no. 1, pp. 47–58, Feb. 2022.
[30] S. S. Shinde and D. Tarchi, "Joint air-ground distributed federated learning for intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 9996–10011, Sep. 2023.
[31] H. Xiao, J. Zhao, Q. Pei, J. Feng, L. Liu, and W. Shi, "Vehicle selection and resource optimization for federated learning in vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11073–11087, Aug. 2022.
[32] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: A survey and taxonomy," *IEEE Commun. Surveys Tuts.*, vol. 11, no. 4, pp. 19–41, 4th Quart., 2009.
[33] Q. Tang, Z. Fei, B. Li, and Z. Han, "Computation offloading in LEO satellite networks with hybrid cloud and edge computing," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 9164–9176, Jun. 2021.
[34] X. Gu and G. Zhang, "Energy-efficient computation offloading for vehicular edge computing networks," *Comput. Commun.*, vol. 166, pp. 244–253, Jan. 2021.

SWAPNIL SADASHIV SHINDE (Student Member, IEEE) was born in Pune, India, in 1991. He received the B.Eng. degree in electronics and telecommunication engineering from Pune University, India, in 2013, the M.Des. degree in communication systems from Indian Institute of Information Technology, Design and Manufacturing, Kancheepuram, India, in 2015, and the M.S. degree in telecommunication engineering and the Ph.D. degree in automotive engineering for intelligent mobility from the University of Bologna, Italy, in 2020 and 2024, respectively, with a focus on connected vehicles for beyond 5G scenarios.

From 2015 to 2017, he was a Project Engineer with Indian Institute of Technology, Kanpur, India. Since November 2023, he has been a Researcher with the Consorzio Nazionale Interuniversitario delle Telecomunicazioni (CNIT), Italy. His research interests include edge computing, reinforcement learning, distributed machine learning, and non-terrestrial networks.

DANIELE TARCHI (Senior Member, IEEE) was born in Florence, Italy, in 1975. He received the M.Sc. degree in telecommunications engineering and the Ph.D. degree in informatics and telecommunications engineering from the University of Florence, Florence, in 2000 and 2004, respectively.

From 2004 to 2010, he was a Research Associate with the University of Florence. From 2010 to 2019, he was an Assistant Professor with the University of Bologna, Bologna, Italy, where he is currently an Associate Professor. He is the author of around 160 published papers in international journals and conference proceedings. He has been involved in several national and international research projects and collaborates with several foreign research institutes. His research interests include wireless communications and networks, satellite communications and networks, edge computing, distributed learning, smart cities, and optimization techniques.

Prof. Tarchi is an Editorial Board Member of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE OPEN JOURNAL OF THE COMMUNICATION SOCIETY, and *IET Communications*. He was the Symposium Co-Chair of IEEE WCNC 2011, IEEE Globecom 2014, IEEE Globecom 2018, and IEEE ICC 2020, and the Workshop Co-Chair of IEEE ICC 2015.

Open Access funding provided by 'Alma Mater Studiorum - Università di Bologna' within the CRUI CARE Agreement