This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

Deep Reinforcement Learning-based Task Scheduling and Resource Allocation for Vehicular Edge Computing: A Survey

Peisong Li, *Member, IEEE,* Xinheng Wang, *Senior Member, IEEE,* Changle Li, *Senior Member, IEEE,* Muddesar Iqbal, *Member, IEEE,* Anwer Al-Dulaimi, *Senior Member, IEEE,* Chih-Lin I, *Fellow, IEEE,* Pablo Casaseca-de-la-Higuera

***Abstract*—With the development of intelligent transportation systems, vehicular edge computing (VEC) has played a pivotal role by integrating computation, storage, and analytics closer to the vehicles. VEC represents a paradigm shift towards real-time data processing and intelligent decision-making, overcoming challenges associated with latency and resource constraints. In VEC scenarios, the efficient scheduling and allocation of computing resources are fundamental research areas, enabling real-time processing of vehicular tasks and intelligent decision-making. This paper provides a comprehensive review of the latest research in Deep Reinforcement Learning (DRL)-based task scheduling and resource allocation in VEC environments. Firstly, the paper outlines the development of VEC and introduces the core concepts of DRL, shedding light on their growing importance in the dynamic VEC landscape. Secondly, the state-of-the-art research in DRL-based task scheduling and resource allocation is categorized, reviewed, and discussed. Finally, the paper discusses current challenges in the field, offering insights into the promising future of VEC applications within the realm of intelligent transportation systems.**

***Index Terms*—Vehicular edge computing; Deep reinforcement learning; task scheduling; resource allocation.**

This work received partial support from the National Natural Science Foundation of China (NSFC) under grant 52175030, PSDSRC Funded Projects ID PID-000085_01_04 and PID-000085_01_03, and the EU Horizon 2020 Research and Innovation Programme under the Marie Sklodowska-Curie grant agreement No. 101008297. The authors would like to thank Prince Sultan University for their Support. (*Corresponding author: Xinheng Wang*)

Peisong Li is with the Hangzhou Institute of Technology, Xidian University, Hangzhou 311200, China, E-mail: lipeisong@xidian.edu.cn.

Xinheng Wang is with the School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China, (E-mail: xinheng.wang@xjtlu.edu.cn).

Changle Li is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China, E-mail: clli@mail.xidian.edu.cn.

Muddesar Iqbal is with the Renewable Energy Laboratory, Communications and Networks Engineering Department, College of Engineering, Prince Sultan University, Riyadh 11586, Saudi Arabia, (E-mail: miqbal@psu.edu.sa).

Anwer Al-Dulaimi is with the College of Technical Innovations, Zayed University, Dubai, United Arab Emirates (Email: anwer@veltris.com).

Chih-Lin I is with the China Mobile Research Institute, Beijing 100053, China, (Email: icl@chinamobile.com).

Pablo Casaseca-de-la-Higuera is with the Laboratorio de Procesado de Imagen (LPI), Universidad de Valladolid, Valladolid, Spain, (E-mail: jcasasec@tel.uva.es).

I. Introduction

Nowadays, Intelligent Transportation System (ITS) is evolving with advancements in connectivity, data analytics, and AI, paving the way for smarter, safer, and more efficient transportation systems [1]. ITS refers to the integration of advanced technologies and communication systems into transportation infrastructure and vehicles to improve safety, efficiency, and sustainability in transportation, which leverages various technologies such as sensors, communication networks, data analytics, and automation to enable real-time monitoring, control, and management of transportation systems [2]. The development of ITS has evolved significantly, starting from basic traffic management systems and progressing towards more advanced applications, including traffic signal optimization, congestion management, incident detection and response, intelligent routing and navigation, connected and autonomous vehicles, and multi-modal transportation integration.

In recent years, vehicular networks have served as a crucial foundation for enabling and supporting various ITS applications. Vehicular networks refer to the early stages of communication infrastructure designed specifically for vehicles [3]. These networks aim to facilitate communications between vehicles and vehicles as well as between vehicles and roadside units. Vehicular networks rely on technologies like Dedicated Short Range Communications (DSRC) and Wireless Access in Vehicular Environments (WAVE) to enable vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications [4].

Vehicular ad hoc networks (VANETs) represent a subset of vehicular networks where vehicles form a self-organizing and decentralized network. VANETs utilize wireless communications to establish ad hoc connections between vehicles in close proximity [5]. These networks leverage V2V and V2I communications to exchange safety-related information, such as collision warnings, traffic updates, and road conditions. VANETs aimed to enhance road safety, traffic management, and cooperative driving.

VANETs initially focused on enabling communication and cooperation between vehicles and infrastructure. With the advent of Vehicular Cloud Computing (VCC), cloud computing technologies were integrated into the vehicular

domain, allowing for resource-intensive applications to be offloaded to the cloud for processing and storage [6].

However, VCC faced challenges related to latency, bandwidth, and reliance on remote cloud infrastructure. In addition, the vehicular services are becoming latency-sensitive and resource-intensive. In order to address these limitations, vehicular edge computing (VEC) emerged as a paradigm that leverages edge computing resources deployed at the network edge, closer to the vehicles, enabling real-time data processing, analytics, and intelligent decision-making [7]. VEC brings computation, storage, and analytics capabilities to the proximity of vehicles, to reduce latency, bandwidth consumption, and enhance overall system performance [8]. While VANETs provide the underlying communication framework for vehicular networks, VEC introduces an additional layer of computation capabilities at the network edge. Unlike traditional VANETs, which focus solely on reliable data exchange, VEC necessitates joint optimization of communication and computation resources to meet latency and energy constraints. For example, task offloading decisions must account for both wireless channel conditions (communication) and edge server workload (computation). This dual focus distinguishes VEC from VANETs and emphasizes the need for cross-disciplinary approaches integrating networking and edge computing. This development represents a shift towards localized, faster, and more efficient processing within the vehicular environment, enabling innovative applications and advancing the field of intelligent transportation [9].

In VEC scenarios, task scheduling/offloading and resource allocation are two main research topics that ensure efficient utilization of edge computing resources and enable real-time processing of vehicular tasks. Task scheduling in VEC involves determining which tasks or computations should be executed locally within vehicles and which should be offloaded to the edge computing infrastructure [10]. Resource allocation in VEC focuses on efficiently allocating computing, storage, and communication resources among vehicles and edge servers [11]. These techniques enable intelligent decision-making, real-time analytics, and efficient allocation of resources within the vehicular environment, contributing to the success of VEC applications and the overall advancement of intelligent transportation systems. The high-level overview of the vehicular edge computing scenarios is elucidated by Fig. 1. As shown in the figure, in VEC scenarios, vehicles on the road can offload tasks to edge servers and the cloud server. The cloud server and edge servers provide computation and communication resources to vehicles, enabling efficient task processing and real-time decision-making. In addition, vehicles can communicate with roadside infrastructure, such as traffic lights and roadside units, to facilitate vehicle-road collaboration. This interaction enhances the overall performance of intelligent transportation systems by supporting applications like emergency services, traffic management, and autonomous driving.

Over the past years, several surveys have studied different aspects of VEC systems. In [12], a comprehensive survey of VEC and its implications for smart vehicles and vehicular networks is presented. It illustrates the VEC architecture, discusses technical issues and solutions, and highlights future research challenges. In [13], a comprehensive survey of VEC is presented, which describes key research topics, conducts a literature review, and identifies open research issues and future directions in the field. In [14], the VEC is introduced, including describing VEC concepts, technologies, and architectures, discussing resource allocation mechanisms, reviewing security approaches, and highlighting the main challenges. However, to the best of the authors' knowledge, There are very few surveys focused on summarizing the latest research work on task scheduling and resource allocation in VEC scenarios. In addition, Deep Reinforcement Learning (DRL) algorithms have gained significant attention and usage in the domain of task scheduling and resource allocation due to their ability to make intelligent decisions in complex, dynamic environments [15]. Reinforcement learning (RL) holds distinct advantages over conventional optimization algorithms in dynamic decision-making scenarios. RL's adaptability to changing environments, ability to balance exploration and exploitation, model-free learning, online learning capabilities, and suitability for handling partial observability make it particularly effective in real-world problems. RL excels in scenarios involving complex and high-dimensional decision spaces, sequential decision-making, and robustness to uncertainty, addressing challenges that conventional optimization algorithms may struggle with due to their static nature, reliance on accurate models, and limited capacity for handling dynamic and uncertain environments. The inherent flexibility and learning capacity of RL position it as a powerful approach for dynamic decision-making problems.

In this context, the latest studies on DRL-based task scheduling and resource allocation in VEC are reviewed. Firstly, the development of VEC and the basic concepts of DRL are introduced. Secondly, the state-of-the-art research is reviewed, classified, and then discussed, respectively. Finally, the evaluation metrics of the research are presented and current challenges are discussed. The scope of this survey is more focused on DRL applications within VEC, which has not been the central theme of earlier surveys. In addition, this survey is up-to-date, incorporating studies and developments up to the present year, which is beyond the coverage of previous surveys.

The main contributions are summarised as follows:

(1) This work is the first comprehensive survey specifically focused on DRL-based task scheduling and resource allocation in VEC scenarios. This is significant as it fills a critical gap in the literature, providing a foundation upon which future research can build.

(2) A detailed introduction of VEC is provided, where the architecture, application scenarios, task scheduling and resource allocation problems are elaborated in detail. Then, the deep reinforcement learning method is introduced. After

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

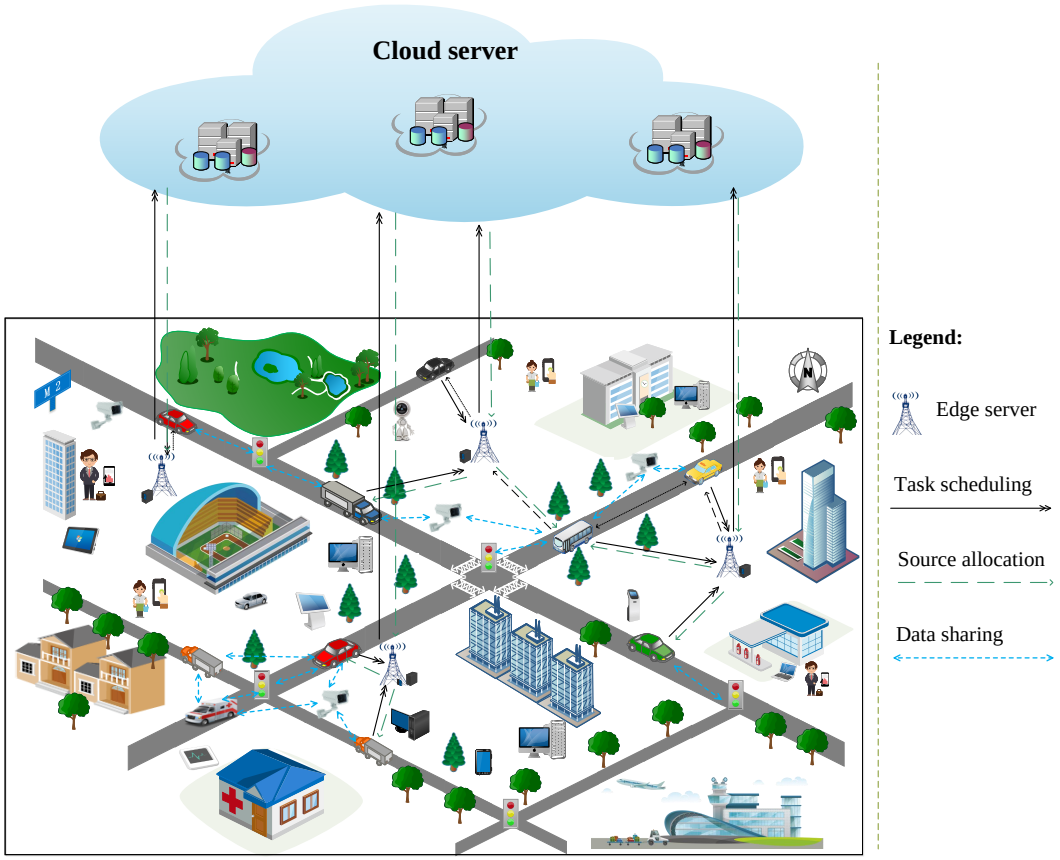

Fig. 1: Overview of the vehicular edge computing scenarios.

this, a comprehensive review of recent advancements in DRL-based task scheduling and resource allocation within VEC environments is presented. The studies are categorized according to the utilized DRL algorithms, the DRL decisions, and the optimization targets. This contribution is vital for keeping both academic researchers and industry practitioners updated with the cutting-edge developments in this fast-evolving field.

(3) In addition to summarizing existing studies, this work delves into a deep analysis of these works, identifying strengths and limitations. Based on the analysis, we offer insights into potential future research directions and emerging industrial applications, aiming to guide and inspire subsequent investigations and implementations.

The rest of the paper is organized as follows: In Section II, the architecture of the VEC system is introduced. It also presents the key components of VEC. In Section III, the concepts of DRL and some DRL algorithms are introduced. The latest studies on task scheduling and resource allocation using DRL algorithms are introduced and compared in Section IV. The real-world applications of the VEC are introduced in Section V. The typical evaluation metrics are introduced in Section VI. The future research directions are presented in Section VII. Finally, this paper is concluded in Section VIII. The framework of the survey is shown in Fig. 2. The key abbreviations are shown in Table I.

TABLE I: List of main abbreviations

Abbrevation	Description
ITS	Intelligent Transportation System
VEC	Vehicular Edge Computing
MEC	Mobile Edge Computing
V2V	Vehicle to Vehicle
V2I	Vehicle to Infrastructure
VANETs	Vehicular ad hoc networks
VCC	Vehicular Cloud Computing
DRL	Deep Reinforcement Learning
IoV	Internet of Vehicles
RSUs	Roadside Units
PPO	Proximal Policy Optimization
SAC	Soft Actor Critic
DQN	Deep Q-Network
DDPG	Deep Deterministic Policy Gradient
A3C	Asynchronous Advantage Actor Critic

II. Vehicular Edge Computing

In this section, the system model of VEC will be introduced first. Then, some key factors in VEC are discussed, including task scheduling, resource allocation, communications, and computation.

A. System Model

VEC is a paradigm that integrates edge computing technologies into the vehicular environment [16]. It leverages

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

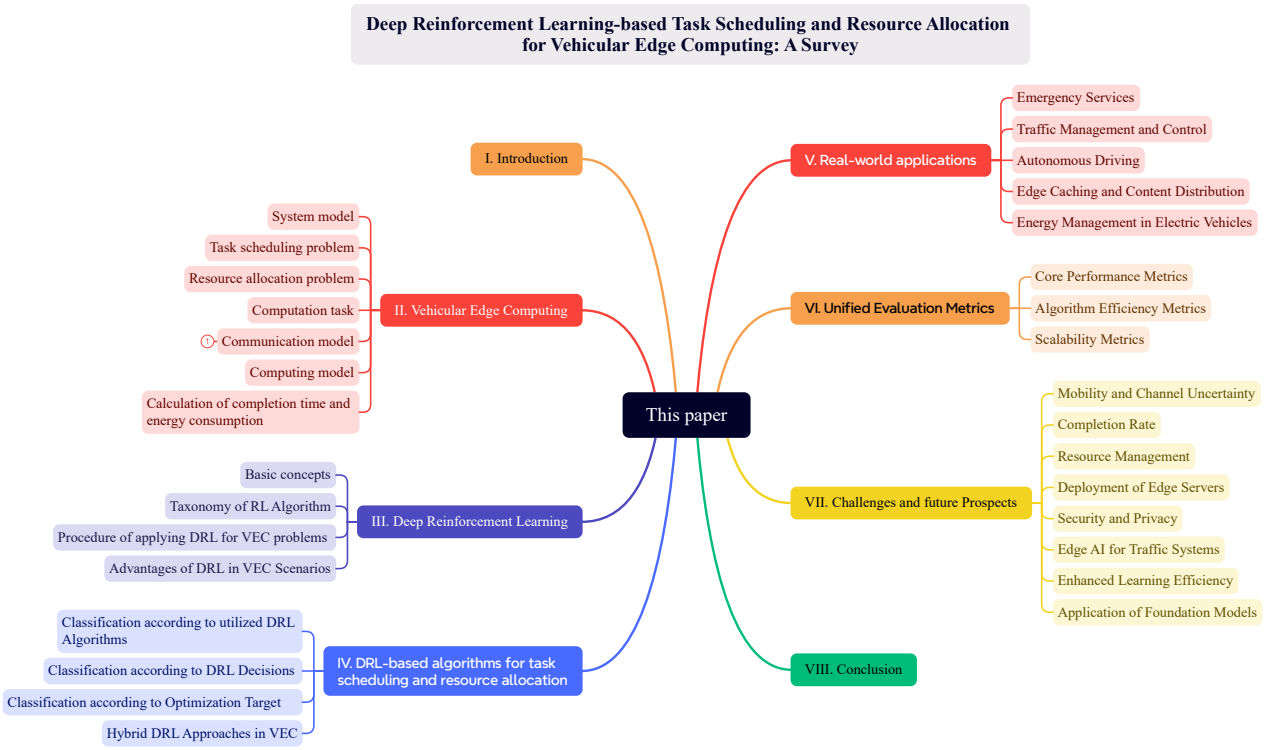

Fig. 2: Organization of the survey paper.

edge servers and fog nodes deployed at the network edge, closer to vehicles, to enable real-time data processing, analytics, and intelligent decision-making. VEC brings computation, storage, and analytics capabilities in close proximity to vehicles, reducing latency, bandwidth consumption, and reliance on remote cloud infrastructure.

While both MEC and VEC involve integration of edge computing into specific environments, VEC is tailored to the unique characteristics of vehicular environments, considering mobility, low latency, and resource constraints specific to vehicles. VEC aims to provide real-time data processing, decision-making, and intelligent services within the vehicular domain, ultimately enhancing safety, efficiency, and overall transportation experience.

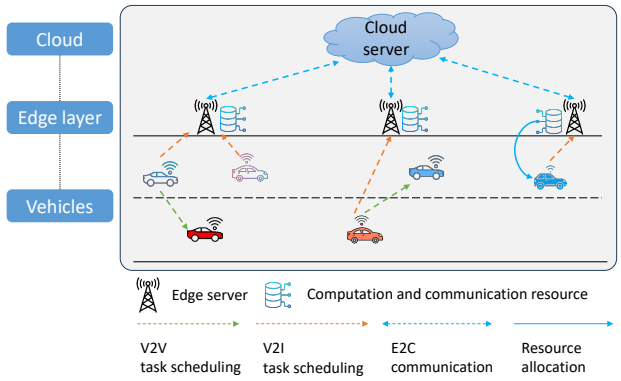

Fig. 3: The architecture of the three-layer VEC.

1) Architecture: A typical VEC system is comprised of three layers, including vehicles, edge layer, and cloud layer [17]. The architecture of the three-layer VEC is shown in Fig. 3.

a) Vehicles: Smart vehicles play a vital role in the VEC ecosystem. These vehicles are equipped with advanced sensors, communication capabilities, and computational resources, enabling them to generate, collect, process, and transmit data within the VEC architecture.

b) Edge layer: The edge layer in VEC refers to the layer that comprises edge servers, RSUs, and other computing resources deployed at the network edge, closer to the vehicles. This layer brings storage and computation capabilities in proximity to the vehicles, enabling real-time data processing and low-latency interactions. The edge layer performs localized data processing and analytics, leveraging the computational resources available at the edge. It reduces latency, network congestion, and dependency on remote cloud infrastructure. The edge layer enables fast response time, supports time-sensitive applications, and enhances the overall performance of VEC.

c) Cloud layer: The cloud layer in VEC represents the layer that consists of remote cloud infrastructure, such as data centers, servers, and storage facilities. This layer provides large-scale computing and storage resources, which are traditionally used in cloud computing. The cloud layer can be leveraged for tasks that do not require real-time processing or are more resource-intensive, such as long-term storage, batch processing, and complex data analytics. It

offers scalability, flexibility, and expansive computing power for applications that can tolerate higher latency or involve massive data processing.

2) Application scenarios: Currently, the VEC systems have been applied in various scenarios. Some of the key application scenarios are illustrated as follows:

a) Emergency Services: VEC facilitates faster incident detection, emergency response, and coordination among vehicles, infrastructure, and emergency services. It enables real-time sharing of information, location tracking, and efficient resource allocation during emergency situations [18].

b) Vehicular safety: VEC supports cooperative collision warning systems by enabling fast processing of sensor data and real-time communications between vehicles. It enhances safety by providing timely alerts, collision prediction, and cooperative manoeuvres to prevent accidents [19].

c) Traffic Management and Optimization: VEC supports real-time traffic management and optimization by leveraging edge computing capabilities. Through V2I and V2V communications, vehicles share data about traffic conditions, congestion, and incidents. Edge servers analyze these data to dynamically adjust traffic signal timings, optimize signal phasing, and provide real-time route guidance to vehicles [20].

d) Edge-Assisted Autonomous Driving: VEC plays a crucial role in the development of autonomous driving by providing edge computing resources. In edge-assisted autonomous driving scenarios, vehicles collect sensor data and offload computationally intensive tasks, such as sensor fusion, localization, mapping, and decision-making, to the edge servers. The edge servers process the data in real time, enabling faster perception, analysis, and response, thereby enhancing the safety and reliability of autonomous vehicles [21].

e) Path planning and vehicle navigation: In VEC, path planning and vehicle navigation applications benefit from localized data processing and real-time decision-making. Vehicles can leverage edge servers to analyze traffic data, road conditions, and historical patterns to determine the optimal path for navigation. Edge-based path navigation systems can provide real-time updates, dynamic rerouting, and personalized route suggestions based on the current traffic situation and individual preferences [22].

f) Edge-Based infotainment and sServices: VEC enables edge-based infotainment and personalized services within vehicles. With the help of edge servers, vehicles can access real-time multimedia streaming, personalized advertisements, location-based services, and other infotainment applications. The edge resources provide low-latency delivery of content, ensuring a seamless and enjoyable user experience for vehicle occupants [23].

g) Ultra-low latency services: VEC is well-suited for ultra-low latency services that require immediate response times. For applications such as real-time video analytics, Augmented Reality (AR) guidance, or time-critical vehicle-to-vehicle communicationS, edge servers can process data locally, minimize latency and provides near-instantaneous results [24].

h) Computation-intensive services: VEC offloads computationally intensive tasks from vehicles to the edge servers. Applications such as high-resolution sensor data processing, complex simulations, or AI-powered analytics can benefit from the high computational capabilities at the edge [25].

B. Task Scheduling Problem

Task scheduling in VEC refers to the process of efficiently allocating computation tasks between vehicles and edge servers deployed at the network edge [26]. It involves deciding which computational tasks should be executed locally within vehicles and which tasks should be offloaded to the edge servers for processing. The task scheduling scheme can be operated into two main steps: (1) Task division: The service request generated from one vehicle can be divided into multiple tasks. These tasks can be executed serially or concurrently. (2) Scheduling decision: This step determines ”which task should be offloaded?” and ”where to offload the task?” [27].

For task scheduling, real-time and safety-critical tasks may be executed locally to minimize communication delays, while computationally intensive or less time-sensitive tasks can be offloaded to the edge servers for efficient processing. By strategically managing task allocation, VEC ensures timely and accurate data processing, facilitating intelligent decision-making and improving the overall efficiency of vehicular applications.

C. Resource Allocation Problem

Resource allocation involves efficient distribution and utilization of computational, storage, and communication resources among vehicles and edge servers deployed at the network edge [28]. It encompasses dynamic allocation of resources based on the current demand, system load, and application requirements.

Algorithms and mechanisms for resource allocation consider factors such as task priorities, network conditions, available resources, and real-time data analysis to make informed decisions. By effectively allocating resources, VEC maximizes the performance of applications, reduces latency, and enhances the overall efficiency and reliability of the VEC ecosystem [29].

The overall workflow of task scheduling and resource allocation in a VEC system is presented in Fig. 4. It depicts how computation tasks generated by vehicles can either be executed locally or offloaded to nearby edge servers, based on resource availability and latency requirements. This figure also highlights the interaction between task scheduling decisions and resource allocation mechanisms, providing a visual representation of the system’s coordination process.

D. Computation Task

1) CPU cycles requirement: CPU requirement refers to the amount of computational resources or processing capac-

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

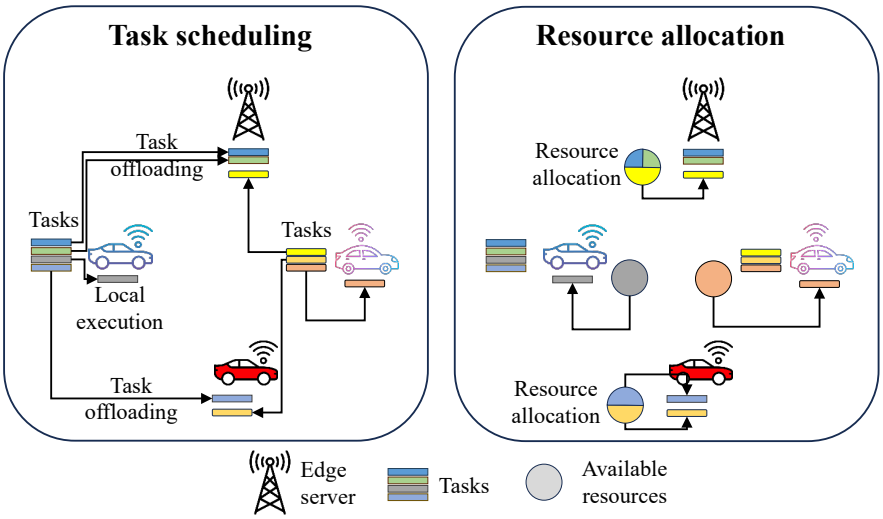

Fig. 4: Illustration of task scheduling and resource allocation.

ity that a specific task requires to execute effectively and efficiently [30].

2) Data Size: Data size relates to the volume of data generated, transmitted, or processed within the VEC systems. This could include the size of data packets, images, videos, or any other information exchanged between vehicles and the edge computing infrastructure [31].

3) Maximum tolerable latency: Tolerable latency refers to the maximum allowable delay or latency that a VEC application can tolerate without affecting its functionality or safety. In other words, it's the threshold for how quickly a task or data transfer needs to be completed to meet the requirements of the application [32].

E. Communication Model

In VEC scenarios, V2V and V2I communications are two fundamental components, enabling vehicles to communicate with each other and with the surrounding infrastructure [33].

The data transmission rate is the rate at which data can be transmitted over the wireless communication link. It depends on the modulation scheme, channel conditions, and available bandwidth. The Shannon-Hartley theorem provides a fundamental formula for calculating the maximum achievable data rate:

$$R = B \cdot \log_2 (1 + SNR) \tag{1}$$

where R represents the data transmission rate, B represents the available bandwidth in hertz (Hz), SNR represents the Signal-to-Noise Ratio.

F. Computing Model

The task completion time depends on the CPU-cycle frequency of the computing entities [34].

1) Local execution: If the task is executed locally on the vehicle, the local completion time T_{loc} can be represented by $T_{loc} = c_i / f_i$, where c_i and f_i represent the computation resource requirement of the task and the CPU-cycle frequency of vehicle i, respectively.

2) Offloading: If the task is offloaded to the edge server, the completion time T_{off} depends on the allocated CPU-cycle frequency from the edge server, calculated by $T_{off} = c_i / f_{i,r}$, where $f_{i,r}$ represent the CPU-cycle frequency of edge server r allocated to the task. The maximum CPU-cycle frequency of edge server can be defined as F_r^{max}.

G. Calculation of Completion Time and Energy Consumption

1) Completion time: For the local computation, the task completion time is equal to the task's local execution time. For the task offloaded to the edge server, the task completion time comprises transmission time and execution time [35]. The task completion time can be calculated according to:

$$T = \begin{cases} T_{loc}, \text{ local} \\ T_{trans} + T_{off}, \text{ offloading} \end{cases} \tag{2}$$

where T_{trans} represents the task transmission time, calculated by $T_{trans} = s_i / R_{i,r}$, in which s_i denotes the data size of the task, $R_{i,r}$ denotes the data transmission rate between the vehicle i and edge server r.

2) Energy consumption: Typically, the energy consumption E can be estimated generally by :

$$E = \begin{cases} \xi \cdot (f_v)^\gamma \cdot c_i, \text{ local} \\ p_v \cdot \frac{s_i}{R_{i,r}} + \xi \cdot (f_r)^\gamma \cdot c_i, \text{ offloading} \end{cases} \tag{3}$$

where f_v and f_r are the computing power of vehicle v and edge server r, ξ and γ are constant and represent the vehicle power consumption coefficients, s_i and c_i represent the data size and CPU requirement of the task i.

3) Weight allocations: In multi-objective optimization problems, reward functions are often designed to combine latency and energy consumption with specific weights. These weights depend on the application requirements. For latency-sensitive applications, higher priority is given to minimizing delay, while for energy-constrained environments, energy efficiency becomes the primary focus. For example, in the application scenario of emergency services, the weight value of completion time is 0.7 while that of energy consumption is 0.3. In contrast, in the scenario of EV charging, the weight value of completion time is 0.2 while that of energy consumption is 0.8.

III. Deep Reinforcement Learning

In this section, the basic concepts of DRL and different DRL algorithms will be introduced. The DRL method is widely used in VEC scenarios for task scheduling and resource allocation [36]. The structure of this section is shown in Fig. 5.

A. Basic Concepts

DRL is a subfield of artificial intelligence and machine learning that combines two powerful techniques: deep learning and reinforcement learning. It aims to train agents to make intelligent decisions in an environment, similar to how humans and other intelligent beings learn from their experiences and interactions with the world.

In conventional reinforcement learning, an agent acquires knowledge through trial and error. It engages with an environment, performs actions, and receives feedback in the form of rewards or penalties corresponding to its actions. The objective for the agent is to acquire a policy, constituting a strategy or mapping from states to actions, with the aim of maximizing the cumulative expected reward over time.

DRL introduces deep neural networks to represent the agent's policy or value functions. These deep neural networks allow the agent to handle high-dimensional and complex environments, making it suitable for tasks such as playing games, controlling robots, and making autonomous driving decisions. The interface between agent and environment is shown in Fig. 6. The following describes the typical components of a DRL system.

1) Agent: The agent serves as the learner or decision-maker involved in interactions with the environment. It represents the entity undergoing training to execute a particular task or attain a goal. The agent observes the prevailing state of the environment and chooses actions to execute in accordance with its existing policy. The primary objective of the agent is to acquire an optimal policy, mapping states to actions, with the aim of maximizing the cumulative reward over the duration of the learning process.

2) Environment: The environment is the external context with which the agent interacts. It could be a simulated environment like a video game or a real-world environment like a robot navigating through its surroundings. The environment offers responses to the agent, delivering rewards or penalties according to the actions undertaken by the agent. The agent's actions affect the environment, and the environment responds by transitioning to a new state.

3) State space: The state serves as a depiction of the current condition or observation of the environment perceived by the agent, aiding in decision-making. The state space encompasses the entirety of conceivable states that the environment can assume.

4) Action space: The action space represents the set of all possible actions that the agent can take in the environment. Action refers to the choices available to the agent that it can take within the environment. The action space can be discrete (e.g., a finite set of actions) or continuous (e.g., a range of real values). In VEC scenarios, the action for task scheduling is discrete while the action for resource allocation is continuous.

5) Reward: The reward serves as immediate feedback from the environment to the agent following its action, signifying the effectiveness of the action and the resulting consequences. The agent's aim is to maximize the overall cumulative reward across time to accomplish the task's objective.

6) Policy: The policy is the strategy or mapping that the agent uses to decide which action to take in a given state. It defines the agent's behavior and governs its decision-making process. In DRL, the policy is often represented using a neural network, where the input is the state, and the output is the action probabilities (in the case of a stochastic policy) or the action itself (in the case of a deterministic policy).

7) State value and state-action value function: In reinforcement learning, state value function and state-action value function are two important concepts used to assess the anticipated cumulative reward from a particular state or state-action pairing, respectively, under a specific policy. They are central to many value-based RL algorithms.

a) State Value Function: The state value function, denoted as $V_\pi(s)$, signifies the expected anticipated reward when starting from a particular state s and following a certain policy thereafter. In other words, $V_\pi(s)$ estimates how good it is for the agent to be in state s and continue following the policy from that point onwards. The state value function is defined as the sum of immediate rewards and the expected future rewards, discounted by a factor (γ) that represents the agent's inclination toward immediate rewards compared to those in the future. Bellman equation is one of the central elements of many Reinforcement Learning algorithms [37]. The Bellman equation for the state value function can be expressed by:

$$V_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^t r_{t+1} | s_t = s \right] \tag{4}$$

where $\mathbb{E}$ represents the expected value, γ is the discount factor, r_{t+1} is the reward received at time step $t+1$, s_t is the state at time step t, and π is the policy.

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

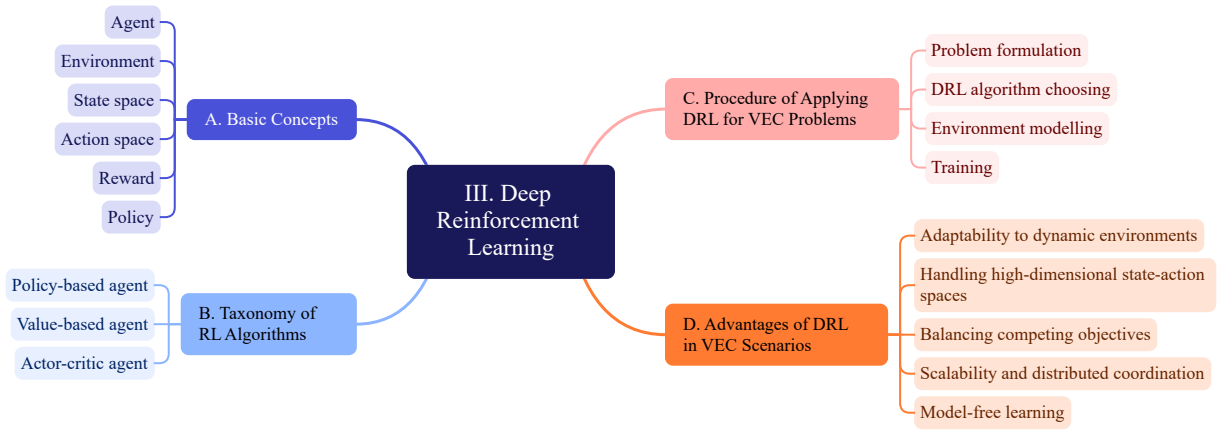

Fig. 5: Organization of Section III.

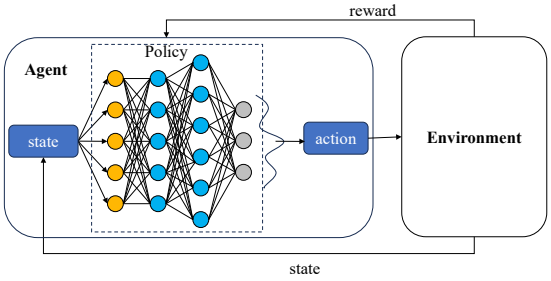

Fig. 6: Agent-environment interface.

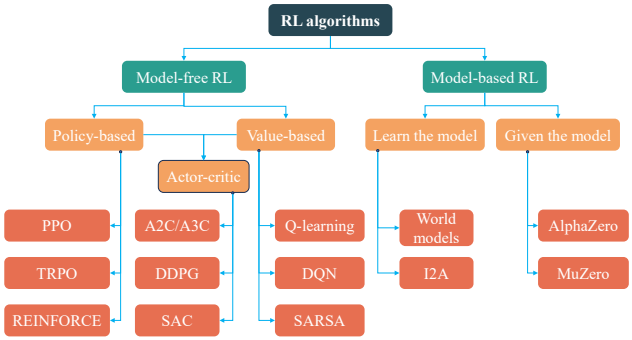

Fig. 7: Taxonomy of RL algorithms.

b) State-Action Value Function: The state-action value function, denoted as $Q_\pi(s,a)$, estimates the expected cumulative reward when starting from state s, taking action a, and following a certain policy thereafter. In other words, $Q_\pi(s,a)$ quantifies how good it is for the agent to take action a in state s and continue following the policy from that point onwards. The state-action value function is defined similarly to the state value function but takes into account both the immediate reward of taking action a and the expected future rewards, discounted by the factor γ. The Bellman equation for the state-action value function is:

$$Q_\pi(s,a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_t = s, a_t = a \right] \quad (5)$$

where $\mathbb{E}$ denotes the expected value, γ is the discount factor, r_{t+1} is the reward received at time step $t+1$, s_t is the state at time step t, a_t is the action at time step t, and π is the policy.

B. Taxonomy of RL Algorithms

As shown in Fig. 7, model-free and model-based are two main categories of reinforcement learning (RL) approaches. Model-free RL focuses on directly learning policies or value functions from interactions with the environment. It is often computationally more straightforward and can be applied when the environment's dynamics are complex or unknown. Model-based RL, on the other hand, learns a model of the environment, which can be used for planning and simulation to make decisions. It has the potential to be more sample-efficient, especially in tasks with fewer interactions with the environment. However, it requires an accurate estimation of the environment's dynamics, and the planning process can become computationally expensive for large and complex environments. In this context, model-free methods are more popular and have been more extensively developed and tested than model-based methods.

The deep neural networks in DRL can be trained using various techniques, such as Q-learning, policy gradients, and actor-critic methods. These algorithms iteratively improve the agent's policy over time, leading to better decision-making capabilities and achieving higher rewards in the given environment.

1) Policy-based agent: A policy-based agent in reinforcement learning directly learns a policy, which is a strategy or mapping from states to actions, without explicitly estimating value functions [38]. The agent's policy is typically

represented by a parametric function, such as a deep neural network. This function takes the current state as input and produces probabilities for various actions as output. Policy-based agents are particularly effective in handling continuous action spaces and complex, high-dimensional environments, and they offer more stable convergence compared to value-based methods in certain scenarios.

PPO is a widely used policy gradient algorithm that ensures stable updates by limiting the change in the policy parameters [39]. TRPO is another policy gradient method that guarantees monotonic improvement by constraining the policy updates within a trust region [40].

More characteristics of the policy-based reinforcement learning method are summarized as follows:

a) Network architecture: Policy-based models directly learn the policy function that maps states to actions. These models can be implemented using feedforward neural networks for simpler environments or Recurrent Neural Networks (RNNs) for environments where the agent's decision might depend on a sequence of previous states.

b) Hyperparameters: Key hyperparameters include the learning rate, the discount factor (γ), the policy's exploration strategy, and the architecture specifics.

c) Training Procedures: During training, the agent engages with the environment, collects experience, and updates its policy using optimization methods like policy gradients to increase the expected cumulative reward.

d) Performance Metrics: Common metrics include the cumulative reward, convergence speed, and the stability of the learning process.

2) Value-based agent: A value-based agent in reinforcement learning learns value functions, such as state-values or action-values, to estimate the expected cumulative reward from a given state or state-action pair under a specific policy. The agent typically employs iterative updates, using Bellman equations or variants, to improve its value function estimates. By choosing actions based on the highest value estimates, the agent makes decisions that maximize the expected cumulative reward. Value-based agents are efficient for handling discrete action spaces and can be less affected by variance in the gradients compared to policy-based methods. However, they may face challenges in handling continuous action spaces, as finding the optimal action requires an additional optimization step. Q-learning, Deep Q Network (DQN), and state-action-reward-state-action (SARSA) are representative value-based algorithms.

Q-learning is a popular off-policy algorithm that estimates the state-action value function (Q-values) and updates the Q-values using the Bellman equation. DQN is an extension of Q-learning that uses deep neural networks to represent the Q-values, enabling it to handle high-dimensional state spaces like images [41]. SARSA is an on-policy algorithm that updates the Q-values by considering the expected value of the subsequent state-action pair under the current policy.

More characteristics of the value-based reinforcement learning method are summarized as follows:

a) Network architecture: Value-based models focus on learning the value function, which estimates how good it is to be in a given state or how good it is to take a certain action from a given state.

b) Hyperparameters: Besides the learning rate and discount factor, important hyperparameters include the size of the replay buffer, the target network update frequency, and the mini-batch size used for training.

c) Training Procedures: The value function is learned by minimizing the difference between predicted and actual returns, typically using variants of Q-learning. Techniques like experience replay and fixed Q-targets are employed to stabilize training.

d) Performance Metrics: Evaluation is based on the accuracy of value estimation, the cumulative reward, and how consistently the agent achieves high rewards across various runs.

3) Actor-critic agent: An actor-critic agent in reinforcement learning combines policy-based and value-based approaches to improve learning stability and efficiency. The agent consists of two components: the "actor" and the "critic." The actor directly learns a policy, mapping states to actions, using policy gradient methods to update its policy based on the expected cumulative reward. The critic, on the other hand, estimates value functions to provide a more stable estimate of the expected cumulative reward and guide the actor's learning process.

For example, Advantage Actor-Critic (A2C)is an actor-critic algorithm that updates both the actor and critic networks in parallel, using the advantage function to estimate the advantage of taking an action in a state compared to the average value [42]. Deep Deterministic Policy Gradients (DDPG) is also an actor-critic method designed for continuous action spaces, where the actor learns a deterministic policy, and the critic estimates the action-value function [43]. Furthermore, Asynchronous Advantage Actor-Critic (A3C) is a parallelized version of A2C that uses multiple agents to update the actor and critic networks asynchronously, enabling more efficient exploration [44].

More characteristics of the actor-critic-based reinforcement learning method are summarized as follows:

a) Network architecture: Actor-critic models combine the advantages of policy-based and value-based approaches. The "actor" learns a policy function, while the "critic" estimates the value function. These models can be implemented using separate networks for the actor and critic or a shared architecture with distinct output layers.

b) Hyperparameters: This category inherits hyperparameters from both policy-based and value-based models, including learning rates for both the actor and the critic, discount factors, and exploration strategies. Additionally, the trade-off between the actor and critic's learning rates can significantly impact performance.

c) Training Procedures: The critic learns to predict the value of state-action pairs, and the actor updates its policy based on the critic's feedback. This feedback often comes in the form of an advantage function, which indicates how

much better an action is compared to the average action in a given state.

d) Performance Metrics: Performance is evaluated through a combination of the actor's ability to maximize cumulative rewards and the critic's accuracy in value estimation.

C. Procedure of Applying DRL for VEC Problems

Applying DRL methods for task scheduling and resource allocation in VEC involves four key steps. The detailed procedure is presented as follows:

1) Problem formulation: Problem formulation includes the definition of the environment, the identification of the actions, and the setting of the reward function. Firstly, we need to define the state space that represents the system's status at any given time. Secondly, determine the possible actions the DRL agent can take, such as allocating computational resources and scheduling tasks to specific edge servers. Thirdly, design a reward function that aligns with the goals of task scheduling and resource allocation.

When determining the actions, the scalability of the agents must be considered. In robotics scenarios, scaling is usually less of an issue since the focus is often on individual robots or small teams. For video games, while games can have many entities, they operate within the limits of the game engine, and the scale is controlled by the game developers. However, in vehicular networks, the DRL system must scale to handle a large number of vehicles and infrastructure elements, often in a distributed manner.

The setting of the reward function impacts the training performance. Rewards in robotics can be straightforward but may involve complex task-specific goals. In video games, rewards are typically well-defined by game rules and are designed to be achievable and motivating. However, in vehicular networks, the reward function often involves multiple objectives such as safety, efficiency, and energy consumption. Balancing these objectives can be complex.

2) DRL algorithm choosing: A suitable DRL algorithm needs to be selected based on the problem's characteristics (e.g., DQN for discrete action spaces, DDPG or SAC for continuous action spaces), the complexity of the environment, and the computational resources available.

3) Environment modelling: Three factors should be considered when modelling the environment: Environment simulation, state representation, and action representation. Firstly, develop or use an existing simulation environment that accurately represents the VEC scenario. This is crucial for training the DRL agent, as it provides a controlled setting to explore different strategies. Secondly, define a comprehensive yet efficient representation of the system's state, incorporating relevant information such as the current task queue, resource utilization levels, and network conditions. Thirdly, ensure the action space is well-defined, whether it's discrete or continuous, and represents the possible decisions the agent can make for task scheduling and resource allocation.

The environment in robotics is often more controlled and predictable, especially in industrial settings. Even in less structured environments, the physical interactions are governed by well-defined physical laws. Video game environments, while complex, are fully known and deterministic, allowing for easier simulation and experimentation. In contrast, the environment in vehicular networks is highly dynamic and unpredictable. Vehicles move at varying speeds, and their interactions can change rapidly due to traffic conditions, road layouts, and communication delays.

4) Training: Finally, the DRL agent can be trained using the simulation environment, monitoring its performance over time to ensure it is learning the intended task scheduling and resource allocation strategies.

D. Advantages of DRL in VEC Scenarios

1) Adaptability to dynamic environments: VEC environments are highly dynamic because of vehicle mobility, fluctuating network conditions, and various task demands. Unlike traditional optimization methods like convex optimization or heuristic algorithms, DRL algorithms can effectively learn and adapt to real-time changes without requiring the design of explicit system models [45].

2) Handling high-dimensional state-action spaces: The state space of VEC are typically multi-dimensional, including edge server workload, vehicular workload, vehicle position, channel states, task characteristics, and so on. Action space includes discrete actions like offloading targets and continuous actions like resource allocation. Compared with conventional methods, deep neural networks in DRL can compress high-dimensional inputs into actionable policies [46].

3) Balancing competing objectives: VEC requires trade-offs between latency, energy, and system cost. Reward function shaping in DRL enables multi-objective optimization without manual tuning.

4) Scalability and distributed coordination: In real-world scenarios, large-scale VEC networks involve hundreds of vehicles competing for edge resources. Facing these kinds of environments, multi-agent DRL can decentralize decision-making while avoiding the "curse of dimensionality."

5) Model-free learning: VEC systems usually face uncertainties like unpredictable vehicle trajectories or sudden task arrivals, which are hard to address for conventional optimization methods. Model-free DRL algorithms have the strength that they learn directly from interactions, eliminating reliance on idealized assumptions.

IV. DRL-based Algorithms for Task Scheduling and Resource Allocation

The DRL-based optimization methods can be classified either according to the utilized DRL algorithms, the DRL decisions, or the optimization objective. In this section, the latest studies on task scheduling and resource allocation are summarized. The structure of this section is shown in Fig. 8.

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

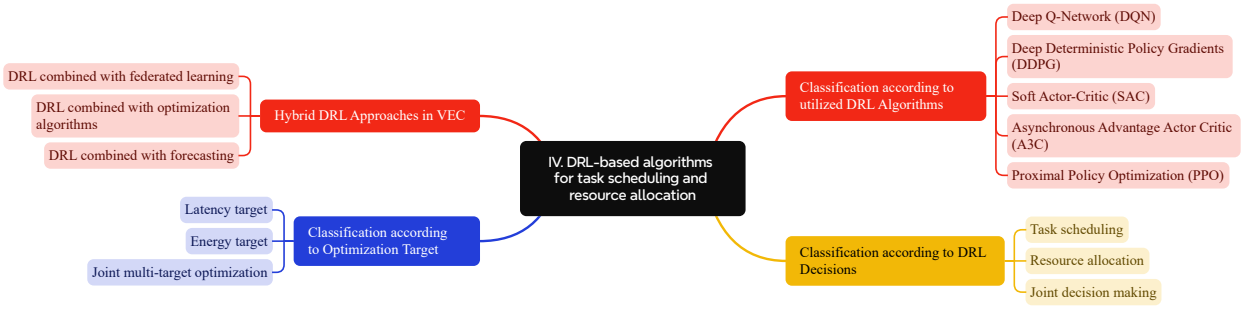

Fig. 8: Organization of Section IV.

A. *Classification according to utilized DRL Algorithms*

In this sub-section, the DRL-based optimization methods are classified and reviewed according to the utilized DRL algorithms.

1) Deep Q-Network (DQN): DQN combines deep neural networks with Q-learning, enabling agents to learn optimal action-value functions in complex environments. It uses experience replay and target networks to stabilize training and has been notably successful in training agents for tasks with high-dimensional state spaces and discrete action spaces. The three technical innovations in DQN are introduced as follows:

(a) The DQN algorithm relies on the Q-value function, $Q(s, a)$, which estimates the expected cumulative rewards for each action a in a given state s. Given a state s, the DQN algorithm outputs the best policy by selecting the action a that maximizes the Q-value. This can be expressed as:

$$\pi(s) = \underset{a}{argmax} Q(s, a) \tag{6}$$

This policy determines the action that the agent should take to maximize its expected rewards.

(b) Traditional Q-learning tables are impractical for environments with large state-action spaces due to memory and computation constraints. DQN addresses this by using deep neural networks (DNNs) to approximate the Q-value function. The neural network, parameterized by weights θ, takes the state s as input and outputs Q-values for all possible actions. The Q-value function is updated using the Bellman equation:

$$Q(s, a) \approx Q(s, a; \theta) \tag{7}$$

$$Q(s, a) = r + \gamma \underset{a'}{max} Q(s^{'}, a^{'}; \theta^{-}) \tag{8}$$

where r is the reward received after taking action a in state s, γ is the discount factor, and θ^{-} are the parameters of a target network, which is periodically updated to stabilize training.

(c) The third innovation in DQN is the introduction of the Experience Replay method, which improves the efficiency and stability of learning. Instead of learning from consecutive experiences, DQN stores the agent's experiences $(s, a, r, s^{'})$ in a replay buffer. During training, mini-batches of experiences are randomly sampled from this buffer to break the correlation between consecutive experiences and to smooth the training data distribution. This random sampling helps to reduce the variance of the updates and leads to more stable and efficient learning.

DQN works well in the context of task scheduling and resource allocation [47]:

Paper [48] addressed the challenge of executing computation-intensive applications on resource-constrained vehicles through vehicular computation offloading. The proposed solution involves a multi-agent DQN algorithm, where multiple vehicles make offloading decisions to minimize the total task processing delay over the long term. In this paper, the reward convergence performance was assessed over 100 training iterations. The proposed method's reward was compared with the rewards obtained from the Actor-Critic (AC) algorithm. The results demonstrated that the DRL approach outperformed AC algorithm in terms of reward convergence, indicating its superior capability in optimizing the given problem. In addition, the performance evaluation was conducted under varying numbers of vehicles and the evaluation metrics include delay and packet transmission performance.

Paper [49] focused on minimizing overall processing delay while considering energy limits and involved DQN for online offloading and a Lagrange-based migration algorithm for computation optimization. In this work, over 4000 training iterations, the reward convergence of the DQN algorithm was thoroughly analyzed under different learning rates. This comparison aimed to identify the optimal learning rate that maximizes reward efficiency, results show that training with a learning rate of 0.6 can achieve the highest reward and training with 0.8 can achieve the fastest convergence. The evaluation metrics chosen for the study were average delay and energy consumption. The performance of the utilized DQN algorithm was further tested across varying task workloads to examine its adaptability.

In [50], the authors introduced a collaborative three-tier decentralized network known as Vehicle-Assisted Multi-Access Edge Computing to tackle computation offloading challenges in highly mobile Internet of Vehicles (IoVs). In this context, they applied a Multi-Agent Deep Reinforce-

ment Learning-based Hungarian Algorithm to address the dynamic task offloading problem within the VMEC framework. In this paper, the DQN algorithm is assessed over 3000 training iterations, focusing on its reward convergence performance compared to two other extended DRL algorithms. The evaluation metrics include time delay, energy consumption, and average cost and the performance is tested under varying numbers of vehicles to simulate different traffic densities.

In [51], the NP-hard resource allocation problem was solved by applying DQN algorithm to dynamically learn network state dynamics and find optimal solutions for different vehicular application offloading, aiming to reduce response time and enhance QoS. In this paper, the DQN algorithm's reward convergence performance is analyzed and compared with those of basic greedy and random scheduling algorithms over 300 training iterations. The primary evaluation metric is the acceptance rate, which reflects the proportion of successfully scheduled tasks.

In [52], a method called CoOR was proposed, which focuses on collaborative task offloading and service caching replacement from a vehicle-centric viewpoint. To tackle issues associated with the interdependence of task offloading and service caching, diverse computation requests, and changing data transmission conditions, an iterative algorithm that combines Gibbs sampling and DQN was employed to determine optimal decisions. This paper focuses specifically on the total cost as the primary evaluation metric. The performance of the utilized DQN algorithm is tested under a variety of conditions including different vehicle speeds, varying coverage areas of RSUs, diverse starting locations of the tasks, and fluctuations in RSU computing resource prices. These diverse testing conditions simulate real-world variability in MEC environments.

As shown in the above studies, DQN introduced techniques like experience replay and target networks, which help stabilize training and improve convergence in discrete action spaces. However, DQN is not applicable to environments with continuous action spaces.

2) Deep Deterministic Policy Gradients (DDPG): DDPG is a model-free reinforcement learning algorithm that extends traditional DQN to continuous action spaces, enabling it to learn deterministic policies for tasks such as robotic control. DDPG combines elements of actor-critic methods and deep learning, utilizing both a policy network and a value function network to achieve stable and efficient learning in environments with continuous action spaces. Following is an introduction of the two innovations in DDPG:

(a) In DDPG, the goal is to find the best policy that maximizes the expected cumulative rewards. The algorithm uses a Q-value function $Q(s, a)$ that estimates the expected return for taking action a in state s. Additionally, an action-value function, $\mu(s)$, represents the policy that determines the best action to take in each state. Given state s and action a, the DDPG algorithm outputs the best policy by optimizing the action-value function and the Q-value function simultaneously.

(b) DDPG follows an actor-critic approach where two separate neural networks, the actor and the critic, work together. The actor network learns the policy. It takes the state s as input and outputs the action $a = \mu(s|\theta^{\mu})$, where θ^{μ} represents the parameters of the actor network. The critic network learns the Q-value function. It takes both the state s and the action a as inputs and outputs the Q-value $Q(s, a|\theta^{Q})$, where θ^{Q} represents the parameters of the critic network. The actor is updated by following the gradient of the expected return, improving the policy directly. The critic is updated using the Bellman equation to minimize the temporal difference error:

$$L\left(\theta^{Q}\right) = \mathbb{E}\left[\left(r + \gamma Q\left(s', \mu\left(s' \mid \theta^{\mu'}\right) \mid \theta^{Q'}\right) - Q\left(s, a \mid \theta^{Q}\right)\right)^{2}\right] \tag{9}$$

where $\theta^{\mu'}$ and $\theta^{Q'}$ are the parameters of target networks for the actor and critic, respectively, which are slowly updated to provide stable targets for training.

DDPG is typically used in VEC scenarios for task scheduling and resource allocation [59]:

In [53], a DDPG-based task offloading scheme for vehicular edge computing was proposed, which involves using mobile vehicles as mobile edge servers (MESs) to assist fixed edge servers in completing computation tasks of mobile devices. In this research, the DDPG algorithm's reward convergence performance is analyzed under varying conditions including different learning rates, batch sizes, and discount factors, to optimize its configuration for the task. The evaluation metrics employed are the average task offloading request hit ratio and Quality of Experience (QoE), which gauge the user satisfaction of task scheduling. Additionally, the performance of the proposed DDPG-based method is evaluated across edge servers with differing computation capacities.

In [54], the authors focused on VEC and addressed the challenge of jointly optimizing service caching and computation offloading in dynamic vehicular networks. They employed DDPG algorithm to obtain a suboptimal solution with low computational complexity. In this paper, the reward convergence of the designed method is thoroughly compared with other offloading scenarios: traditional offloading without edge caching, offloading aimed at minimizing latency, and offloading focused on reducing energy consumption. The primary evaluation metrics used to assess the algorithm's performance are total delay and total energy. Moreover, the effectiveness of the DDPG algorithm is tested under conditions of varying task sizes and different computational capacities of edge servers.

In [55], the authors formulated a joint task scheduling and resource allocation strategy to reduce energy cost while meeting delay constraints. To solve this problem, they employed the MADDPG method to obtain the optimal

TABLE II: Studies using DDPG and DQN algorithms

Paper	Year	Method	Action	Objective	Description
[53]	2023	DDPG	Offloading target selection.	Maximize the Quality of Experience (QoE).	To identify an efficient strategy for offloading computation tasks from mobile devices to achieve the maximum QoE for these devices.
[54]	2023		Proportion of the tasks offloaded to edge node and edge pool.	Minimize the average task completion time.	To solve a collective optimization challenge by incorporating both service caching and computation offloading in a typical scenario involving VEC with task requests that vary over time.
[55]	2020		Whether to offload, computation resource allocation, channels allocation.	Decrease the energy cost.	A vehicular speed-aware task scheduling and resource allocation strategy was proposed.
[56]	2022		Offloading decision, transmit power allocation	Minimize latency and energy	To utilize DQN for devising the offloading strategy and employ DDPG to formulate the strategy for determining the transmit power of vehicles.
[57]	2020		Selection of receiver RSU, helper RSU, and deliver RSU.	Minimize the service cost.	Designed a Task Partition and Scheduling Algorithm (TPSA) to determine the allocation of workload and the selection of servers.
[58]	2023	DQN	Offloading decision, resource allocation.	Maximize offloading success rate.	Proposed a collaborative computation offloading model that accommodates various offloading patterns.
[48]	2021		Server selection to receive the tasks.	Minimize the total task completion time.	Several mobile vehicles opt for nearby MEC servers to offload their computing tasks.
[49]	2020		Communicational and computational resource allocation.	Optimize overall processing delay.	A mechanism for joint communication and computational resource allocation (RJCC) is introduced to enhance the optimization of overall processing delay.
[50]	2022		Whether to offload, edge server selection, offloading to the cloud or not.	Minimize the cost of computation resources.	To develop a collaborative three-layer VMEC network structure where vehicles, under associated and neighbouring RSUs, constitute VMEC servers.
[51]	2022		Server selection.	Maximize the acceptance rate.	Proposed a collaborative decision-making approach between MEC and the central cloud for offloading tasks in various vehicular applications.

offloading and resource allocation strategy. The core aspect of this study is the comparison of the DDPG-based method's reward convergence under varying numbers of vehicles, which introduces different levels of complexity and demand on the system. The key evaluation metrics considered are task completion delay and energy consumption, crucial for assessing the performance of the scheduling solution.

In [56], a two-stage strategy utilizing DDPG for task scheduling and resource allocation was proposed to reduce execution and processing latency as well as energy consumption. This study specifically assesses the reward convergence performance of the DDPG algorithm, comparing it under various conditions including different numbers of vehicles and fluctuating task arrival rates. The evaluation metrics employed to gauge the effectiveness of the algorithm include long-term execution delay, energy consumption, and processing accuracy—factors critical to the reliability of MEC systems. The performance of the DDPG algorithm is tested against these different task arrival rates to determine its adaptability.

Paper [60] presented a VEC model with a focus on task offloading, considering data dependency of tasks in urban scenarios. To minimize system response time and energy consumption, the authors proposed a Mobility-aware dependent task offloading (MESON) Scheme and developed a DDPG based algorithm for training the offloading strategy. This study compares the reward convergence performance of the DDPG algorithm against two other DRL strategies, DQN and Q-learning. Evaluation metrics such as response time and energy consumption are employed to assess the algorithms' effectiveness. The performance of the DDPG algorithm is tested under scenarios with varying numbers of vehicles and different task loads, showcasing its ability to not only outperform DQN and Q-learning in terms of quicker response times and lower energy consumption but also demonstrating its robust adaptability to fluctuating network densities and task volumes. The DDPG-based and DQN-based methods are outlined in Table. II.

DDPG is well-suited for environments with continuous action spaces, such as adjusting the bandwidth or computing

resources in VEC. However, DDPG can suffer from overestimation bias due to noise in policy and value function estimation.

3) Soft Actor-Critic (SAC): SAC aims to maximize the expected cumulative reward in continuous action spaces. It introduces an entropy regularization term to encourage exploration, making it well-suited for problems with high-dimensional state and action spaces where both efficient exploration and stable learning are crucial. Following is the detailed introduction to SAC:

(a) SAC is based on the actor-critic architecture, where the actor network $\pi_\theta(a|s)$ (policy) decides the actions to take, and the critic networks $Q_{\phi_1}(s|a)$ and $Q_{\phi_2}(s|a)$ estimate the expected return (Q-value) for the given state-action pairs. Unlike traditional actor-critic methods, SAC includes an entropy term in its objective function to encourage exploration. The entropy term ensures that the policy remains stochastic, preventing premature convergence to suboptimal policies. The objective for the policy is to maximize the expected return while also maximizing the entropy of the policy:

$$J_\pi(\theta) = \mathbb{E}_{s_t\sim\mathcal{D},a_t\sim\pi_\theta}\left[\alpha\log\pi_\theta\left(a_t \mid s_t\right) - Q\left(s_t,a_t\right)\right] \quad (10)$$

where α is a temperature parameter that controls the trade-off between exploration (entropy) and exploitation (expected return).

(b) The temperature parameter α in the entropy term can be fixed or learned. SAC often includes an automatic entropy adjustment mechanism where $alpha$ is adjusted to maintain a desired level of policy stochasticity. The objective for α is to minimize the difference between the expected entropy of the policy and a target entropy $\mathcal{H}$:

$$J(\alpha) = \mathbb{E}_{a_t\sim\pi_\theta}\left[-\alpha\log\pi_\theta\left(a_t \mid s_t\right) - \alpha\mathcal{H}\right] \quad (11)$$

This adaptive adjustment allows the agent to balance exploration and exploitation dynamically during training.

Therefore, SAC is also utilized for decision making in VEC [61]:

A distributed dynamic many-to-many task offloading framework within vehicular fog computing (VFC) was proposed in paper [62], where vehicles serve as fog nodes. Th authors applied a Multi-Agent Gated Actor Attention Critic (MA-GAC) algorithm to optimize offloading efficiently in a distributed manner. Over 8000 training iterations, the reward convergence performance of the SAC algorithm is analyzed and compared to that of the DDPG algorithms. The primary evaluation metric used in the study is the external cost gained, which measures the overall expense incurred by the system during task scheduling.

In [63], a Vehicular Edge-Cloud Computing (VECC) network was introduced for processing delay-sensitive tasks in IoT devices, in which SAC algorithm was leveraged to develop effective task offloading policies in dynamic environments. The study compares the reward of the SAC algorithm using different target update rates and discount factors to identify the most effective parameters. The primary evaluation metric is tolerance time, which measures the system's ability to handle task scheduling within acceptable delays. Performance evaluations are conducted with varying numbers of tasks to assess the algorithm's scalability and robustness. The SAC algorithm's performance is benchmarked against several baseline algorithms, including SARSA, DQN, Double DQN, and Dueling DQN. Results from the experiments demonstrate that the SAC algorithm outperforms these baselines in terms of reward convergence and effectively reduces tolerance time.

Paper [64] presented a computation offloading and task scheduling scheme for Internet of Vehicles. This scheme utilizes SAC algorithm for effective decision-making and scheduling, with the goal of maximizing the number of computation task offloading executions within the constraints of the edge server's limited computing resources. This paper focuses on key evaluation metrics: timeout rate and energy consumption. The experiments evaluate the SAC algorithm's performance under varying numbers of tasks to assess its scalability. The results demonstrate that the SAC algorithm effectively reduces the timeout rate and energy consumption compared to traditional methods.

Experimental results in the above studies show that SAC is more sample-efficient than DDPG and DQN, making it suitable for environments where data collection is costly. However, the structure of SAC is more complex than DQN or DDPG, requiring careful implementation and tuning of its entropy regularization component.

4) Asynchronous Advantage Actor Critic (A3C): A3C uses multiple agents, or threads, to explore an environment concurrently. It combines actor-critic architecture, where one neural network (the actor) suggests actions and another (the critic) evaluates those actions, with asynchronous updates to enhance exploration and accelerate learning in complex environments. Following is the introduction to A3C, highlighting its key innovations:

(a) The first key innovations of A3C is the use of multiple parallel agents, each interacting with its own copy of the environment. These agents operate asynchronously, collecting experiences and updating the shared global model in parallel. This approach helps to stabilize training by decorrelating the data and reducing the risk of getting stuck in local optima. Each agent independently performs the following steps: 1) Interacts with its environment to collect experience tuples $(s, a, r, s^{'})$. 2) Computes gradients for both the policy and value networks. 3) Updates the global model asynchronously using these gradients.

(b) A3C is an on-policy algorithm, meaning it updates the policy based on the actions taken by the current policy. The advantage function is estimated using n-step returns, which balances bias and variance in the value estimates. The n-step return R is calculated as:

$$R = \sum_{k=0}^{n-1}\gamma^k r_{t+k} + \gamma^n V(s_{t+n}) \quad (12)$$

The actor and critic networks are updated using the computed advantage function:

$$Policy\ loss = -\log \pi_\theta(a_t|s_t)A(s_t, a_t) \tag{13}$$

$$Value\ loss = (R - V(s_t|\theta_v))^2 \tag{14}$$

Additionally, an entropy term is added to the policy loss to encourage exploration and prevent premature convergence:

$$Entropy\ loss = \beta H(\pi_\theta(\cdot|s_t)) \tag{15}$$

where H is the entropy of the policy, and β is a coefficient that controls the strength of the entropy regularization.

A3C has its own advantages in dynamic decision making [65]:

In [66], an end-edge-cloud architecture for computation offloading in the Internet of Vehicles was proposed. It utilizes an A3C-based offloading algorithm that considers efficiency and fairness factors in the reward function, enabling real-time and convenient computing service access for vehicle users. The experiments focus on the reward convergence performance of both the actor and critic networks, comparing their rewards under different learning rates to determine optimal configurations. The primary evaluation metrics include the relative efficiency factor and the relative fairness factor, which assess the algorithm's ability to balance task scheduling efficiency and fairness. Furthermore, performance evaluations are conducted under varying numbers of tasks and different computing capacities of the MEC server, demonstrating the algorithm's adaptability.

Paper [67] addressed the challenge of effectively aggregating and scheduling network resources for diverse tasks by formulating a joint optimization problem for task offloading and resource management, leveraging Asynchronous Advantage Actor-Critic algorithm to find optimal scheduling policies while considering the dynamics and randomness of vehicular networks. This study compares the convergence performance of the proposed A3C-based method under various learning rates to identify the most effective configuration. The primary evaluation metric is latency, which measures the time delay in task processing and completion. Performance evaluations are conducted under different conditions, including varying numbers of vehicles, different task sizes, diverse edge computation capacities, and different vehicle computation capacities.

The studies using A3C have advantages in parallel training, which allows for parallel training across multiple instances, significantly speeding up the learning process. However, in a distributed setting, the need for synchronization and communication between workers can introduce overhead, especially in VEC scenarios with limited bandwidth.

5) Proximal Policy Optimization (PPO): PPO seeks to optimize policies by iteratively updating them in a conservative manner, preventing large policy changes to ensure stable and reliable learning in environments with potentially nonlinear and complex dynamics. Here's an introduction incorporating the specified innovations:

(a) PPO is an on-policy algorithm that, given the current state s, outputs the action probabilities or specific actions to be taken by the agent, as well as a value estimate for those actions. The policy network, parameterized by θ, provides a probability distribution $\pi_\theta(a|s)$ over actions a given state s. The agent samples an action from this distribution or selects the action with the highest probability. Additionally, the value network, parameterized by ϕ, estimates the value function $V_\phi(s)$, which represents the expected return starting from state s. These components work together to ensure the agent not only decides on the best actions to take but also understands the value of being in each state, facilitating more informed policy updates.

(b) Another key innovation in PPO is the clipping mechanism, designed to prevent large, destabilizing updates to the policy. The objective function in PPO is modified to include a clipped probability ratio that ensures the new policy π_θ does not deviate too far from the old policy $\pi_{\theta_{old}}$. This can be expressed as:

$$L^{\mathrm{CLIP}}(\theta) = \mathbb{E}\left[\min\left(r_t(\theta)\hat{A}_t, \mathrm{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right)\right] \tag{16}$$

Here, $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio between the new and old policies, $\hat{A}_t$ is the advantage estimate, and ϵ is a small hyperparameter (e.g., 0.2) that determines the clipping range. This clipping mechanism ensures that updates to the policy are conservative, reducing the risk of policy collapse and leading to more stable and reliable training.

Recently, PPO has been widely used for obtaining optimal task scheduling and resource allocation decisions [72]:

Paper [69] addressed the challenge of multi-task offloading (MTO) in mobile edge computing scenarios. The proposed solution utilizes a PPO algorithm to reduce the execution time for tasks generated from various MTO scenarios by encoding sequential offloading actions and training a meta policy that can quickly adapt to new scenarios with minimal training steps. The performance of the PPO algorithm is thoroughly evaluated under various conditions including different numbers of subtasks, bandwidths of subchannels, and vehicle arrival rates to assess its efficiency and adaptability to dynamic network conditions. The algorithm's efficacy is compared against the DQN method, providing a benchmark for its performance. The results showcase that the PPO algorithm significantly improves computation waiting times, optimizes the offloading proportion, and achieves better load balance and fairness among vehicles compared to DQN.

Paper [70] addressed the challenge of resource allocation to minimize service latency in Internet of Vehicles environments. The method involves a heuristic algorithm that efficiently allocates limited fog resources and integrates the PPO algorithm to make resource allocation decisions by considering vehicle movement and parking status data from the smart city environment. The PPO algorithm's performance is compared against the Advantage Actor-Critic

TABLE III: Studies using SAC, A3C, and PPO algorithms

Paper	Year	Method	Action	Objective	Description
[62]	2023	SAC	Vehicle selection, buy or sell resource	Minimize QoS.	Computational resource trading among vehicles to execute task method.
[63]	2023		Offloading target selection.	Minimize the total time.	Proposed three potential types of task offloading: transferring to BS, shifting to VS, or directing to the cloud.
[68]	2023		Whether to trade, the amount of computational resources to sell or buy as trading intention.	Maximize the trading objective in the long run.	A framework for task offloading, utilizing a many-to-many approach and grounded in the vehicular trading paradigm, was introduced.
[64]	2023		Whether to offload or not.	Joint optimization of time and energy.	Determine whether to perform tasks locally or offload them to the server, considering the satisfaction of time delay constraints for locally executed tasks.
[66]	2022	A3C	Task offloading decision.	Minimize delay.	An end-edge-cloud architecture of vehicles was designed for task computation offloading.
[67]	2023		Offloading decision and resource assignment policy.	Maximize the system utility.	The novelty is the exploration of accessible vehicle resources and the incorporation of service migration considerations.
[69]	2023	PPO	Subchannel and processor selection.	Minimize task execution time.	The issue of multi-task offloading (MTO) was explored, involving multiple offloading scenarios with diverse parameters.
[70]	2020		Resource allocation.	Minimize the service latency.	To allocate the constrained fog resources to vehicular applications, minimizing service latency by leveraging parked vehicles.
[71]	2023		Subchannel selection and transmit power allocation.	Minimize the computational overhead.	Simultaneous optimization of decisions for task offloading, channel assignment, and transmit power allocation.

TABLE IV: Technical comparison among the DRL algorithms

Algorithm	Action space	Exploration	stability mechanisms	Strengths	Challenges	Use cases in VEC
DQN	Discrete	greedy	Experience Replay, Target Networks	Handles discrete offloading decisions, stable learning with Experience Replay	Less effective for continuous actions, high-dimensional state spaces	Simple offloading decisions
DDPG	Continuous	Noise	Experience Replay, Target Networks	Suitable for continuous adjustments, dynamic resource scheduling	Requires careful tuning, complex implementation	Dynamic resource allocation
SAC	Continuous	Entropy Max.	Twin Q-networks, Adaptive Entropy	Robust exploration, handles complex scheduling	Computationally intensive, extensive hyperparameter tuning	Complex scheduling scenarios
A3C	Both	Entropy Term	Asynchronous Parallel Agents	Efficient parallel learning, versatile action handling	Distributed setup complexity, high variance	Large-scale VEC environments
PPO	Both	Clipping	Clipped Objective	Stable learning, simple to implement, balances exploration and exploitation	Frequent updates needed in dynamic environments	Mixed decision types in scheduling

(A2C) method to gauge its relative efficiency. Results from the study demonstrate that the PPO algorithm significantly enhances service satisfaction and effectively manages the distribution between local and cloud processing, leading to an optimized average offloading count.

Paper [71] focused on the joint computation offloading and resource allocation problem in a nonorthogonal multiple access (NOMA) edge computing system. The goal is to minimize computational overhead, considering execution delay and energy consumption, in dynamic environments with time-varying wireless fading channels. The method employs the PPO algorithm to approximate different statistical models for continuous and discrete control, addressing this optimization challenge. In this paper, the PPO algorithm is benchmarked against the A2C, Twin Delayed DDPG (TD3), and SAC algorithms. The experiments focus on reward convergence performance, comparing the rewards achieved by PPO to those of the other algorithms. The average computational overhead of the PPO algorithm is tested under different scenarios including varying data sizes, the number of CPU cycles required by tasks, computing capacities of the MEC server, number of subchannels, and quantities of mobile devices. The results indicate that the PPO algorithm not only consistently achieves superior reward convergence but also maintains lower average computational overhead.

In order to address the challenge of computation-intensive and latency-sensitive tasks in the context of MEC, a method that considers dependency-aware task offloading was proposed [73]. To ensure privacy preservation, the authors propose a distributed DRL-based algorithm with an optimized structure for offloading strategy. The performance of the designed method is compared against the Actor-Critic

(AC) and DQN algorithms. The reward convergence of the PPO algorithm is compared under various learning rates to determine the optimal settings for maximum efficiency. The evaluation focuses on two critical metrics: time cost and energy cost, which are essential for assessing the practical utility of the algorithm in real-world settings. Performance evaluations are carried out under conditions with varying numbers of vehicles to test adaptability. The findings reveal that the PPO algorithm outperforms both the AC and DQN in terms of reward convergence, effectively minimizing both time and energy costs across different vehicle densities. The SAC-based, A3C-based, and PPO-based methods are outlined in Table. III.

PPO achieves a balance between the sample efficiency of methods like SAC and the simplicity of implementation, making it an attractive option for VEC. However, the clipping mechanism in PPO, designed to prevent large policy updates, can sometimes lead to sub-optimal policies if not properly tuned.

In summary, when choosing a DRL algorithm for VEC scenarios, the decision often involves trade-offs between sample efficiency, ease of implementation, ability to handle continuous vs. discrete action spaces, and the computational resources available for training and inference. For instance, while SAC and PPO offer advantages in terms of sample efficiency and robustness in continuous spaces, DQN and A3C may be preferable in scenarios with discrete action spaces or where parallel training infrastructure is available. The comparative summary of the DRL algorithms in VEC scenarios is presented in Table IV.

B. Classification according to DRL Decisions

According to the type of decision made by the DRL algorithms in VEC scenarios, the latest studies can be classified into three categories: making task offloading decisions, making resource allocation decisions, and jointly making task offloading and resource allocation decisions.

1) Task scheduling: Task scheduling is of paramount importance due to its significant impact on system performance. Some studies focused on making optimal scheduling decisions for individual task, multiple tasks, or dependency-aware tasks [17], [79], [80].

In [74], the authors proposed a non-orthogonal multiple access (NOMA) based architecture for VEC. They addressed the challenges of limited resources and high transmission demands by optimizing task offloading using a multi-agent distributed distributional deep deterministic policy gradient (MAD4PG) method.

In [75], a priority-sensitive task offloading scheme was presented for IoV networks. Vehicles periodically exchange beacon messages to inquire about available services, and a DRL algorithm is employed to classify tasks based on priority and computation size, with the goal of maximizing network utility while ensuring the quality of service for vehicles.

Paper [76] addressed the challenge of task offloading services in edge-enabled IoV to enhance the Quality of Experience (QoE) while considering various status of edge servers, vehicles, and vehicular offloading modes. The proposed approach employs DRL to optimize offloading modes, with a focus on saving energy consumption and stabilizing rewards during training.

Paper [77] addressed the challenge of task offloading in Vehicular Fog Computing (VFC) where energy-constrained Road Side Units (RSUs) need to efficiently schedule tasks to mobile fog vehicles.

In [81], the authors addressed task scheduling in a VEC network for intelligent vehicles, where tasks have different urgency and dependencies and must be completed within strict time constraints. The studies on making task scheduling decisions are outlined in Table. V.

These studies present improvements in reducing task execution and communication latency by efficiently scheduling tasks closer to their data sources or end-users. Furthermore, the task completion rates are also enhanced by minimizing task execution times and avoiding bottlenecks.

2) Resource allocation: Resource allocation ensures that the VEC system can effectively manage its limited resources and meet the dynamic demands of vehicular applications. The edge computing resource comprises transmit power, bandwidth, transmission channel, and computation resource.

Paper [82] focused on the bandwidth resource allocation for VEC that considers task dependencies and employs a DDPG algorithm to optimize allocation decisions in a vehicle-edge-cloud environment, efficiently addressing the continuous control problem and achieving rapid convergence.

Paper [83] focused on resource allocation in a Multi-Access Edge Computing (MEC)-based vehicular network, addressing spectrum, computing, and storage resource allocation for different vehicular applications. They employ reinforcement learning to achieve rapid resource allocation decisions to meet quality-of-service (QoS) requirements.

In [84], the authors proposed an architecture that combines centralized decision-making with distributed channel allocation to maximize spectrum efficiency, utilizing deep reinforcement learning techniques along with long short-term memory (LSTM) to adapt resource allocation dynamically based on changing user mobility, demand, and channel conditions.

Paper [85] focused on transmission power allocation for vehicular networks considering multiple stochastic tasks, varying wireless channels, and bandwidth. The proposed approach leverages deep reinforcement learning to address the continuous action space and aims to strike a balance between energy consumption and data transmission delay in an unstable environment.

Paper [87] addressed the task processing decision in vehicular edge computing where tasks can be processed locally or offloaded based on V2I and V2V communications. It focuses on optimizing power allocation in a complex and uncertain VEC environment using a decentralized deep reinforcement learning approach. The studies on making resource allocation decisions are outlined in Table. VI.

TABLE V: The studies on making task scheduling decisions.

Paper	Year	Action	Method	Objective	Description
[74]	2023	Task scheduling	MADDPG	Maximize the service ratio.	The cooperative resource optimization problem was decomposed into two subproblems: task offloading and resource allocation.
[75]	2022		DDPG	Maximize the mean utility.	A scheme for task offloading and resource allocation in an IoV network, considering priority sensitivity, was proposed.
[76]	2021		DDPG	Minimize energy consumption.	A task offloading mechanism for vehicles was suggested with a focus on optimizing QoE.
[77]	2021		SARSA	Minimize total energy consumption.	An energy-efficient vehicle scheduling problem for offloading tasks to mobile fog nodes subject to satisfying constraints of task deadline and resource availability was presented.
[78]	2022		double DQN	Maximize both system utility and energy consumption.	A shared task offloading strategy was proposed that the successfully matched task unit does not need to be recomputed by the VEC server.

TABLE VI: The studies on making resource allocation decisions.

Paper	Year	Action	Method	Objective	Description
[82]	2022	Resource allocation	DDPG	Minimize the average processing latency.	A task offloading scheme for VEC with collaborative computation between vehicles, edge, and cloud, taking into account dependencies, is put forward.
[83]	2020		DDPG	Maximize the number of tasks completed with QoS requirements.	Simultaneous allocation of spectrum, computing, and storage resources in a vehicular network based on Multi-Access Edge Computing (MEC).
[84]	2022		DQN	Maximize the spectrum efficiency of all vehicles involved.	A proposed architecture involves centralized decision-making alongside distributed channel allocation.
[85]	2020		DDPG	Minimize the energy consumption, transmission latency, and bandwidth.	A model for offloading task computations in a diverse vehicular network was developed, taking into account various stochastic tasks and the diversity of wireless channels and bandwidth.
[86]	2021		DQN	Reduce the processing latency and improve the reliability.	A strategy called Intelligent Communication and Computation Resource Allocation (ICCRA) was introduced, employing a multi-objective reinforcement learning approach.

These studies perform well in improving the utilization of edge computing resources, ensuring that computational and network resources are neither underused nor overburdened.

3) Joint decision making: Furthermore, Some studies focused on making task scheduling and resource allocation decisions jointly. Some of them decompose the joint optimization problem into two sub-problems.

Paper [88] addressed the challenge of joint optimization of computation offloading and resource allocation in a dynamic multiuser mobile-edge computing system. The goal is to minimize energy consumption while considering delay constraints and uncertain resource requirements of computation tasks.

Paper [89] addressed the challenging problem of joint task offloading, collaborative computing, and resource allocation in a multi-access edge computing system. They proposed a deep reinforcement learning-based optimization framework, where decisions related to task offloading are made at the upper level, and computing resource allocation is optimized at the lower level.

Paper [90] focused on addressing the challenges posed by data-intensive and latency-sensitive vehicular applications in the IoV through VEC. It employs the twin delayed deep deterministic policy gradient (TD3) algorithm to determine the task offloading and resource allocation strategy.

Paper [91] introduced a joint task type and vehicle speed-aware task offloading and resource allocation strategy for in-vehicle applications. The method involves establishing a model that considers task type and vehicle speed to optimize task offloading and resource allocation, with the aim of minimizing vehicle energy costs and increasing revenue while meeting delay constraints. The studies on jointly making task scheduling and resource allocation decisions are outlined in Table. VII.

Above studies simultaneously optimize both task scheduling and resource allocation decisions to maximize overall

TABLE VII: The studies on jointly making task scheduling and resource allocation decisions.

Paper	Year	Action	Method	Objective	Description
[88]	2022	Joint task scheduling and resource allocation.	DDQN	Minimize the energy consumption.	Joint optimization of computation offloading and resource allocation in a dynamic multi-user MEC system.
[89]	2023		DDQN	Minimize the total energy consumption.	The upper level addresses the task offloading as well as power and subcarriers allocation subproblems, while the lower level focuses on solving the computation resource allocation subproblem.
[90]	2022		TD3	Joint optimization of offloading delay and energy consumption.	The optimal offloading strategy is attained through the application of the twin delayed deep deterministic policy gradient (TD3) algorithm.
[91]	2022		MADDPG	Maximize the utility level of the vehicles.	A strategy for task offloading and resource allocation was suggested, taking into account the type of task and the speed of the vehicle.
[92]	2023		MADDPG	Minimize the system cost.	The actor network underwent a redesign, incorporating a transformer-based temporal feature extraction network and a policy decoupling network.
[93]	2022		DQN	Service cost minimization.	A learning-based channel allocation and task offloading strategy was proposed in temporary UAV-assisted VEC.

TABLE VIII: The studies focused on minimizing latency.

Paper	Year	Objective	Method	Action	Description
[94]	2023	Minimize the latency.	SAC	The proportion of the task offloaded to ES.	The multitype task cooperative offloading among multiple ESs is modelled as a system latency minimization problem.
[95]	2020		SAC	whether to offload, unit price paid to CPU, computation resource allocation.	Vehicles are encouraged to contribute their unused computing resources through a dynamic pricing mechanism.
[96]	2023		DQN	Task offloading decision and the percentage of the resources allocated.	The structure of the action generation network was adjusted to incorporate multiple branches, with each branch producing a one-dimensional action.
[97]	2023		DQN	Spectrum access selection, the transmit power allocation, and the computation block selection.	A collaborative strategy for secure offloading and resource allocation is proposed to enhance confidentiality performance and resource efficiency in VEC networks.

system performance. They generally provide the most comprehensive solutions, optimizing across multiple dimensions. However, they are also more complex to implement and may require more sophisticated DRL models and algorithms.

C. Classification according to Optimization Target

For task scheduling and resource allocation, the optimization target can be classified into three categories: Minimizing latency, minimizing energy consumption, and joint reduction of latency and energy consumption [109].

1) Latency target: Some studies target to minimize the latency, including transmission and computing latency.

Paper [94] focused on cooperative task offloading among edge servers within the context of the IoV. To minimize task execution delay, the authors treat cooperative offloading as a Markov decision process (MDP) and enhance the Soft Actor-Critic (SAC) algorithm's convergence speed and stability using an adaptive weight sampling mechanism.

Paper [95] addressed the challenge of motivating vehicles to share their idle computing resources in vehicular fog computing (VFC) while considering vehicle mobility, task priority, and service availability. The authors used a Soft Actor-Critic (SAC) algorithm to maximize the mean latency-aware utility of tasks in a period by optimizing the task offloading policy.

Paper [96] presented a joint optimization approach for task offloading decisions and resource allocation in a time-varying Mobile Edge Computing (MEC) system. The goal

TABLE IX: The studies focused on minimizing energy consumption.

Paper	Year	Objective	Method	Action	Description
[98]	2021	Minimize the energy consumption.	DQN	Power allocation.	A practical vehicular environment was considered by taking into account the dynamics of mobile vehicular networks.
[99]	2022		DDPG	Computation resource and caching resource allocation.	An optimization problem was formulated to minimize mobile network operator's energy costs by considering the computation and caching energy costs jointly.
[100]	2023		TD3	Decisions regarding task offloading for vehicles, task slicing for RSU, and bandwidth allocation for the MBS are made.	A scenario with random traffic flow and a dynamic network environment was taken into account, where MEC and cloud servers collaborated to process tasks that are sensitive to delays and computationally intensive.
[101]	2023		MADDPG	Policy for task offloading, policy for controlling transmission power, and policy for allocating local computing resources.	The framework takes into account various decision variables related to energy optimization, including constraints on transmission power, the local CPU cycle frequency for processing computational tasks, the quantity of computation tasks to be offloaded, and the allocation of computation resources on edge servers.

TABLE X: The studies focused on jointly minimizing latency and energy consumption.

Paper	Year	Objective	Method	Action	Description
[102]	2022	Minimize the latency and energy consumption.	TD3	transmission power allocation, task partition ratios.	Determine optimal movements for UAVs, allocate task offloading efficiently, and manage communication resources in dynamic MEC environments.
[103]	2023		DDPG	Flying direction and moving distance.	A collaborative algorithm was suggested for UAV movement control, MU association, and MU power control. The optimization process involves iterative enhancements of three sub-problems.
[104]	2022		DDPG	Local execution power and offloading power allocation.	A power allocation scheme that optimally addresses stochastic task arrival and channel variations was developed.
[105]	2023		SAC	Task executor selection for each task.	Task initiators (TIs) generate a set of correlated tasks and subsequently offload them to various task executors (TEs).
[106]	2023		DQN	Offloading decision.	The paper investigates the problem of joint task offloading, resource allocation, and the fast-changing channel between a vehicle and an edge server.
[107]	2022		SAC	Percentage of resources allocated to the vehicle.	A task offloading scheme was introduced for vehicular networks that prioritize considerations of latency and energy efficiency.
[108]	2022		DDPG	Pricing policy.	Based on the unit prices provided by the VEC server, vehicles decide the quantity of computational resources to acquire from the server.

of the paper is to reduce the average task latency and discard rate while operating within the constraints of latency and the server's limited computing resources.

In [97], a DRL-based approach was introduced to enhance resource efficiency in VEC networks. It optimizes transmit power, frequency spectrum selection, and computation resource allocation, with the goal of minimizing processing delay.

In [110], the authors established a dynamic offloading model for multiple moving vehicles, dividing tasks into sequential subtasks, and proposed a Dynamic Framing Offloading algorithm based on Double Deep Q-Network (DFO-DDQN) to find optimal offloading decisions for these sequential subtasks, minimizing total delay and waiting time.

Paper [111] introduced a strategy for cloudlet-based vehicular networks to enhance computation services by employing multi-agent deep reinforcement learning. Specifically, it focuses on optimizing task scheduling to reduce task processing delay. The studies focused on minimizing latency are outlined in Table. VIII.

These studies often demonstrate significant reductions in task completion times by optimizing the scheduling and allocation of resources to prioritize task execution speed. However, focusing solely on latency reduction can lead to suboptimal energy usage because the most rapid computational resources may also be the most energy-intensive.

2) Energy target: For some studies, the minimization of the total energy consumption is the only optimization target [112].

Paper [98] addressed the challenge of task offloading in vehicular networks with limited edge computing resources by jointly considering communication and computation resources. The authors formulated a non-linear problem to minimize energy consumption and tackle the dynamics of mobile vehicular networks.

In [99], a joint computing and caching framework was proposed for mobile network operators (MNOs) in an IoV scenario. The authors formulated an optimization problem aimed at minimizing the MNO's energy expenses, encompassing both computation and caching energy costs. The solution to this problem was obtained using the DDPG algorithm.

In [100], the authors presented a method called Computation Offloading and Resource Allocation (CORA) to process delay-sensitive and computation-intensive tasks in an IoV scenario, with the objective of minimizing the system cost.

Paper [101] aimed to reduce energy consumption without compromising performance by addressing the complex problem of optimal resource allocation in VEC. It jointly optimizes task distribution and radio resource allocation with the target of minimizing energy consumption, considering vehicle mobility and dynamic data traffic in Roadside Units (RSUs). The studies focused on minimizing energy consumption are outlined in Table. IX.

These studies perform well in energy efficiency by adjusting the task offloading based on energy considerations. However, prioritizing energy conservation may result in increased latency for some tasks, as energy-efficient computing resources may not be the fastest.

3) Joint multi-target optimization: Starting in 2022, more studies worked to jointly optimize task completion time and energy consumption.

In [102], a collaborative mobile edge computing system was proposed that utilizes multiple unmanned aerial vehicles (UAVs) and edge clouds (ECs) to offload computation-intensive tasks. The goal is to minimize the sum of execution delays and energy consumption by optimizing UAV trajectories, computation task allocation, and communication resource management.

Paper [103] addressed the resource allocation problem in unmanned aerial vehicle (UAV) networks for mobile edge computing with the goal of minimizing system latency and energy consumption.

Paper [104] focused on optimizing power allocation in a VEC system where each vehicular user (VU) allocates power for task processing through offloading and local execution. The Deep Deterministic Policy Gradient (DDPG) algorithm was utilized to find the optimal power allocation scheme, aiming to minimize long-term power consumption and latency.

In [105], The authors addressed the challenge of task offloading in a multi-access edge computing (MEC) system. The goal is to minimize the weighted sum of task processing latency and energy consumption in the MEC system. They proposed a two-phrase method: firstly, they use a DRL algorithm to optimize task offloading, and secondly, they determine the optimal transmission power for the offloaded tasks.

Paper [113] focused on addressing latency-sensitive and compute-intensive tasks in vehicular networks by utilizing fog computing, specifically in Vehicular Fog Computing (VFC) networks. The goal is to balance latency, computing, and communication constraints while considering energy consumption. The studies focused on jointly minimizing latency and energy consumption are outlined in Table. X.

These studies often employ multi-objective DRL frameworks or reward functions to quantify and optimize the trade-offs, leading to solutions that are more applicable in real-world scenarios where both task completion time and energy consumption are critical.

D. Hybrid DRL Approaches in VEC

Nowadays, some existing studies focus on the research of the integration of DRL with other optimization techniques in VEC scenarios.

1) DRL combined with federated learning: In VEC, privacy protection is a critical challenge due to the sensitive nature of shared vehicular data, such as location and driving patterns. Traditional centralized DRL approaches require raw data aggregation at edge servers, raising concerns about data leakage and user privacy. To address this, Federated Learning (FL) can be integrated with DRL to enable decentralized model training, where vehicles collaboratively train a shared DRL model while keeping their data locally [114]. FL-DRL frameworks leverage gradient aggregation instead of raw data transmission, ensuring privacy preservation while maintaining model accuracy [115].

For example, paper [116] proposed a novel multi-agent federated deep reinforcement learning framework to reduce task offloading latency for electric vehicles on electrified

roads by integrating wireless power transfer and full-duplex MIMO vehicular networks, enabling continuous charging and efficient computation. The framework optimizes task offloading across base stations, UAVs, and satellites while preserving data privacy through differential privacy techniques.

2) DRL combined with optimization algorithms: Unmanned Aerial Vehicles (UAVs) have been applied recently to play a pivotal role in extending edge computing coverage for vehicular networks, but their dynamic positioning presents complex optimization challenges. DRL alone may struggle with the high-dimensional action space of UAV trajectory planning and resource management. By integrating DRL with classical optimization algorithms, the strengths of both approaches can be harnessed: DRL handles high-level decision-making and adaptability to real-time changes, while optimization algorithms provide precise solutions for subproblems like UAV placement and energy-efficient path planning [117].

In [118], the UAV-assisted Two-stage Intelligent Collaboration (UTIC) method was proposed to optimize UAV positioning and task scheduling in VEC, addressing challenges like limited UAV communication range and energy constraints. The approach consists of three key contributions: (1) a UAV-assisted Two-stage Task Scheduling system model to streamline task allocation; (2) an Enhanced Particle Swarm Optimization algorithm to determine optimal UAV positions, minimizing the number of UAVs while ensuring full coverage of mobile vehicles (MVs); and (3) a Deep Deterministic Policy Gradient-based scheduler to optimize offloading decisions, balancing energy consumption, delay, and task priorities for efficient MV task processing.

3) DRL combined with forecasting: In VEC, Electric Vehicles (EVs) require intelligent charging strategies to optimize energy consumption and minimize grid load, but their unpredictable mobility and charging demands complicate decision-making. DRL can be enhanced with forecasting algorithms like LSTM and time-series models to predict EV trajectories and energy requirements, enabling proactive charging station selection and resource allocation [119]. The hybrid approach allows DRL agents to leverage historical and real-time data for more informed decisions, improving energy efficiency and reducing latency.

Paper [120] proposed a vehicular-cloud-assisted MEC network to enhance computation offloading for Intelligent Vehicles by efficiently utilizing idle resources from both MEC servers and dynamic vehicular traffic. Firstly, a deep neural network-based virtual platform that predicts vehicle trajectories and forms vehicular clouds to pool distributed computation resources; Then, a multi-constraint offloading scheme designed to maximize task throughput while ensuring long-term queue stability using Lyapunov optimization; Thirdly, a lightweight hybrid framework that decomposes the problem into per-slot subproblems via Lyapunov drift-plus-penalty and solves them using DQN to reduce complexity, addressing the coupling of variables across time slots.

V. Real-world applications

As shown in Section IV, DRL has shown remarkable potential in addressing task scheduling and resource allocation issues for VEC scenarios. In addition to the research progress, several real-world applications are presented in this section, as presented in Fig. 9, presenting a more in-depth discussion on the deployment and operational aspects of VEC in actual systems. In real-world scenarios, the edge server is equipped on roadside infrastructures like traffic lights, base stations, and roadside units to provide computation and storage sources [28].

A. Emergency Services

1) Overview: Emergency services in VEC require ultra-low latency and high reliability for tasks like collision warnings, ambulance routing, and disaster response. These scenarios involve high-priority tasks with strict deadlines [121].

PPO and A3C are suitable for these kind of scenarios. They provide stable, fast-converging policies crucial for time-sensitive emergency response, where even milliseconds matter. Their ability to handle partial observability is vital when emergency vehicles rapidly enter/exit coverage zones.

2) Implementation methodology: In deploying PPO and A3C for emergency services, the system first establishes a real-time monitoring framework where connected emergency vehicles and roadside units (RSUs) form a prioritized V2X network. For collision avoidance, PPO agents are deployed on edge servers with the following concrete implementation: 1) Input states include vehicle kinematics, LiDAR-based obstacle detection data, and network conditions; 2) The action space comprises discrete emergency maneuvers like hard braking and evasive steering with safety constraints enforced through reward shaping; 3) Training uses prioritized experience replay with synthetic near-crash scenarios from SUMO simulations, progressively fine-tuned with real-world data from Tokyo's emergency vehicle fleet, achieving 93.7% decision accuracy in field tests [122]. For ambulance routing, A3C's parallel actors simultaneously explore alternative routes in a digital twin of the city, processing real-time traffic data from 200+ IoT cameras, reducing emergency response time by 28% compared to traditional navigation systems.

B. Traffic Management and Control

1) Overview: DRL algorithms can optimize traffic signals in real-time by processing data from connected vehicles and infrastructure. This application aims to reduce congestion, enhance traffic flow, and minimize waiting times at intersections. By utilizing the VEC structure, the roadside smart infrastructures can quickly analyze data on traffic density, speeds, and incidents to dynamically adjust traffic signals and reroute traffic. In addition, by leveraging machine learning algorithms at the edge, VEC can predict traffic patterns and potential bottlenecks, enabling proactive management.

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

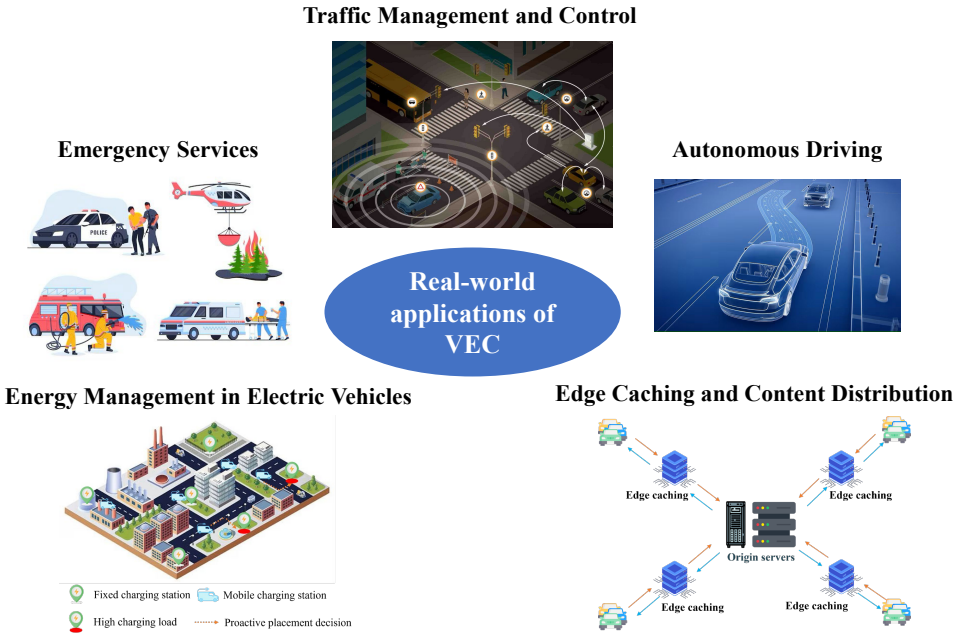

Fig. 9: Overview of the typical real-world applications using DRL for VEC scenarios.

DQN is commonly utilized in this application because discrete but large action spaces like traffic signal combinations benefit from value-based methods with efficient action selection mechanisms.

2) Implementation methodology: In practical deployment, DQN-based traffic control systems are implemented through a three-tier architecture: 1) Vehicle-side OBUs collect real-time speed/direction data and transmit via V2I to edge servers; 2) Edge servers run parallel DQN instances, where the state space incorporates traffic flow metrics, signal phase timing, and queue lengths, while the action space encodes all valid signal phase combinations; 3) The reward function combines multi-objective terms: +0.1 for each vehicle cleared per green phase, -0.05 per second of accumulated wait time, and -2.0 for emergency vehicle interruptions. For example, federal government agencies have proposed an Active Transportation and Demand Management (ATDM) approach [123]. In ATDM, an actor-critic method could be used to continuously adjust traffic signal timings based on the current traffic conditions, learning from the cumulative feedback to minimize overall congestion and improve traffic throughput.

C. Autonomous Driving

1) Overview: In autonomous driving, policy-based DRL can help in making real-time navigation decisions, such as lane changing, speed adjustment, and path planning, by considering the dynamic vehicular environment, including the behaviour of other drivers, traffic conditions, and road constraints. It can be significantly enhanced by VEC. On the one hand, VEC provides the necessary low-latency communication for AVs to share and process critical data, such as obstacle detection. On the other hand, AVs can share sensor data with VEC nodes to build a more comprehensive understanding of their surroundings, improving situational awareness.

Recently, Virginia Tech Transportation Institute researchers put an autonomous Ford F-150 through a series of driving scenarios related to public safety on the 395 express lanes in Arlington, Va. [124].

For this application, SAC and DDPG are suitable choices because continuous action spaces in vehicle control like steering and acceleration match these algorithms' strengths. SAC's entropy maximization is particularly valuable for exploring safe manoeuvres in uncertain environments.

2) Implementation methodology: For autonomous vehicle control, SAC and DDPG are deployed in a hierarchical VEC architecture where: 1) Vehicle-level agents (SAC) process local sensor data to handle immediate driving actions, with entropy regularization enabling safe exploration during uncertain scenarios like pedestrian crossings; 2) Edge-level DDPG coordinators at RSUs aggregate V2X data from 8-12 vehicles within 150m range to optimize platooning strategies, using a centralized critic that evaluates group rewards. Recently, Virginia Tech Transportation Institute researchers put an autonomous Ford F-150 through a series of driving scenarios related to public safety on the 395 express lanes in Arlington, Va. [124].

D. Edge Caching and Content Distribution

1) Overview: DRL can also be applied to decide what content is cached at the edge of vehicular networks in order to minimize content delivery times and reduce backhaul network load. This is particularly relevant for video

streaming, software updates, and map data for autonomous and connected vehicles. By caching frequently accessed content at the edge of the network, VEC reduces the time needed to fetch data from remote servers, improving the driving experience. In addition, VEC minimizes the need for redundant data transmissions over the core network, reducing bandwidth consumption. Since DQN is efficient for discrete server selection, it could be applied to address the content distribution problem.

2) Implementation methodology: For edge caching optimization, a DQN-based system is implemented through a three-phase process: 1) Content popularity prediction at RSUs using LSTM networks processing historical request patterns, classifying content into hot (top 5% requested), warm (next 15%), and cold categories; 2) A distributed DQN architecture where each edge server runs local agents with discrete action spaces (cache/replace/ignore decisions) based on real-time states including cache occupancy (%), request rates (reqs/min), and vehicle density (vehicles/RSU coverage), using a reward function that combines cache hit rate (+0.5 per hit) and backhaul savings (+0.1 per MB conserved); 3) Federated updates every 6 hours to share popularity models across the caching hierarchy. For example, Varnish Software provides fast, efficient content delivery and caching solutions [125]. By using the actor-critic models, the caching strategies can be adjusted dynamically based on user demand and network status, learning to predict and cache the most requested content closer to the users to minimize latency.

E. Energy Management in Electric Vehicles

1) Overview: DRL can optimize the energy consumption of electric vehicles (EVs), including decisions about when and where to charge to minimize costs and ensure the vehicle meets its range requirements, considering factors like the availability of charging stations and real-time electricity prices. Specifically, policy-based DRL algorithms can be used to make decisions on charging behaviour by directly learning a policy that maps state information (e.g., battery level, location, electricity prices) to charging actions (e.g., start charging, stop charging, select charging station). In this process, VEC nodes can manage charging schedules based on real-time data on grid load, electricity prices, and vehicle usage patterns, optimizing energy costs and grid stability. SAC is more suitable for this application because it maximizes entropy for adaptive charging decisions in uncertain environments.

2) Implementation methodology: For optimal EV charging management, a SAC-based system is deployed through a hierarchical architecture where: 1) Vehicle-level agents continuously monitor battery state-of-charge (1% resolution), driving patterns, and real-time electricity prices (5-minute updates from grid APIs); 2) Edge servers at charging stations run federated SAC instances that process aggregated demand forecasts from 50-100 EVs, with continuous action spaces (charging rates: 0-150kW) and entropy-regulated exploration to adapt to dynamic pricing fluctuations; 3) The reward function combines multiple objectives: +2.0 for completing charge within deadline, -0.3 per kWh cost, and +1.5 for utilizing renewable energy peaks. Federal Energy Management Program introduces a Managed Electric Vehicle Charging project, in which managed charging can ensure that vehicles are properly powered when needed while reducing unnecessary burdens on the site's building infrastructure [126].

VI. Unified Evaluation Metrics

In order to evaluate the performance of the designed DRL-based decision-making methods, some metrics are applied according to the optimization objective. Typically, core performance metrics, algorithm efficiency metrics, and scalability metrics are indeed key evaluation metrics used for evaluation in VEC systems. In this section, an explanation of these metrics is presented.

A. Core Performance Metrics

1) Task Completion Time: Task completion time measures the time delay between the initiation of a task or a request and the completion of that task in a VEC system [127]. For example, in [73], the influence of different numbers of vehicles on average time and total latency was explored.

2) Scheduling Time: Scheduling time is the duration taken by the DRL algorithm to generate a scheduling and resource allocation decision for a given set of tasks and resources. Lower scheduling time is preferred as it indicates that the algorithm is capable of quickly providing solutions. In scenarios where real-time decision-making is crucial, minimizing scheduling time becomes essential. Efficient scheduling time also contributes to the responsiveness of the system.

3) Energy Consumption: Energy consumption measures the amount of electrical power or energy used by the VEC system for processing tasks and communication [128]. Optimizing energy consumption is vital, especially in resource-constrained environments or in applications where energy efficiency is a key concern. In [81], the comparison of average energy consumption under different weighting coefficients and methods is presented.

4) Resource utilization: Resource utilization evaluates how efficiently edge computing resources are allocated and utilized in VEC systems. It measures the percentage of available computational resources and communication resources that are effectively used for task processing at any given time. High resource utilization indicates that the DRL algorithm successfully maximizes infrastructure usage, while low utilization suggests inefficiencies in task scheduling or resource allocation.

B. Algorithm Efficiency Metrics

1) Reward Performance: The "reward" is used to assess the overall benefit or utility that the vehicle or system derives

from a specific action or task. In the VEC scenario, the reward could represent the total income generated by the vehicle during a given time period [129].

2) Convergence speed: The convergence speed of reward is critical to evaluate the performance of the proposed method. Faster convergence is desirable because it means that the algorithm is efficiently adapting to the dynamics of the task scheduling and resource allocation problem. Slow convergence may lead to delays in making optimal decisions and, in some cases, may prevent the algorithm from finding an optimal solution at all. In [113], the convergence of the proposed method was analyzed with different learning rates.

3) Computational overhead: Computational overhead quantifies the additional processing resources required by DRL algorithms themselves during both training and inference phases. In VEC environments where edge devices have constrained capabilities, algorithms with lower computational overhead are preferred as they leave more resources available for actual task processing.

C. Scalability Metrics

1) Performance under varying vehicle densities: This critical metric assesses how DRL algorithms scale with changing numbers of connected vehicles in the network. The performance could be evaluated across a spectrum from sparse to ultra-dense scenarios that represent different real-world conditions. This evaluation reveals which algorithm can handle the inherent scalability challenges of VEC, where the same infrastructure must serve fluctuating numbers of vehicles with diverse service requirements.

VII. Challenges and future Prospects

In terms of task scheduling and resource allocation in VEC, some challenges need to be solved. In this section, current challenges are discussed, which represent the future research direction.

A. Mobility and Channel Uncertainty

The mobility of vehicles and channel uncertainty pose a significant challenge in the context of VEC scenarios [130]. The high mobility of vehicles introduces dynamic network topologies, making it difficult to maintain stable connections between vehicles and edge servers. This variability can lead to frequent handovers, increased latency, and potential service disruptions. Nowadays, many existing studies address mobility-aware task offloading and resource allocation utilizing deep reinforcement learning. These works highlight the need for adaptive algorithms that can predict vehicle trajectories and optimize offloading decisions in advance. In this context, future research could explore hybrid models combining DRL with trajectory prediction techniques to further mitigate mobility-induced instability.

On the other hand, vehicular networks are prone to rapid channel variations due to factors like signal fading, interference, and environmental obstacles. This uncertainty complicates resource allocation and task scheduling [46]. Recent studies propose DRL-based channel selection strategies to handle dynamic channel conditions. These approaches leverage real-time feedback to adapt to dynamic environments. However, gaps remain in urban canyons or dense traffic, where channel conditions are highly unpredictable. Therefore, future work could Integrate physical-layer insights with DRL to enhance robustness.

B. Completion Rate

Increasing the completion rate of tasks, which refers to the successful execution of requests or computation tasks within the Maximum tolerable latency, is also one of the main challenges [131]. On the one hand, network connectivity can be unreliable, especially in scenarios where vehicles are constantly on the move. Disruptions and signal interference can lead to task failures, affecting the task completion rate. On the other hand, as vehicles move, the latency of data transmission to and from edge servers may vary. The delay caused by data transmission can lead to service delays, affecting the services that require low latency, such as real-time traffic information, autonomous driving, and safety-critical systems.

C. Resource Management

VEC systems must efficiently allocate computational, storage, and communication resources to meet the varying demands of connected vehicles [132]. The resources are finite, and their allocation needs to be optimized to serve a dynamic set of vehicles. As vehicles move through different regions, the allocation of resources to them may need to change dynamically. Efficiently managing and allocating resources to meet the varying demands of vehicles is a non-trivial task. In addition, different vehicular applications have varying QoS requirements. Some applications demand low latency, while others prioritize data throughput or energy efficiency. Resource management must cater to these specific QoS requirements, which can be conflicting at times.

D. Deployment of Edge Servers

Having an adequate number of edge servers plays a pivotal role in enhancing the vehicular network's performance, especially on a large scale. However, the deployment of these edge servers comes with a substantial cost. Therefore, it becomes crucial to strategically determine the optimal quantity and position of edge servers that should be installed [133], [134]. This optimization process primarily involves identifying suitable locations where the efficiency of vehicular networks can be maximized. Additionally, it's imperative to manage the available resources effectively, ensuring that the edge servers are deployed in a manner that optimizes costs. Consequently, the ultimate goal is to create a model that optimally calculates the minimum requirement for deploying edge servers [135].

E. Security and Privacy

VEC systems handle a vast amount of data, including location information, V2V communications, and sensor data. Protecting these data from unauthorized access, tampering, or theft is crucial [136], [137]. Furthermore, sharing data among vehicles and edge servers can enhance safety and efficiency, but it also introduces privacy concerns. Mechanisms to aggregate data and remove personally identifiable information should be in place.

F. Edge AI for Traffic Systems

The concept of edge intelligence, where DRL models are not just deployed but also trained at the edge, offers the potential for real-time learning [138]. Research will explore distributed learning approaches, such as federated learning, to train DRL models across multiple edge traffic infrastructures, enhancing privacy, reducing latency, and decreasing the bandwidth needed for data transmission from traffic infrastructures to centralized clouds [139].

G. Enhanced Learning Efficiency

Future research can aim to improve the learning efficiency of DRL algorithms, reducing the amount of data and time required to train effective models [140]. Techniques such as transfer learning, meta-learning, and few-shot learning could enable DRL models to adapt to new tasks or environments more quickly, leveraging prior knowledge [141]. This is particularly relevant for dynamic urban environments where traffic conditions change rapidly.

H. Application of Foundation Models

Foundation models are a broader category of AI models designed to provide a general, adaptable base for various applications [142]. They can encompass not only language tasks but also other modalities like images or even multimodal tasks. These models, like GPT-4, are trained on vast amounts of data, enabling them to generate, interpret, and predict text with remarkable accuracy. Furthermore, Large Language Models (LLMs) can be understood as a subset of fine-tuned foundation models with a specific focus on text-based tasks [143]. In addition to LLMs, we can further have large vision models (LVMs), large audio models (LAMs), and large multimodal models (LMMs) [144]. These large models can be applied to the VEC-powered intelligent transportation systems and play a crucial role in improving transportation intelligence, optimizing traffic management, and advancing smart city initiatives by harnessing their language understanding and data analysis capabilities [145].

Looking ahead, further research should delve into large models' capabilities in addressing emerging challenges, refine their real-time decision-making in traffic management, tailor them for smart cities' specific needs, and encourage interdisciplinary collaborations to fully unlock their potential in creating sustainable, intelligent, and people-centric transportation ecosystems.

I. UAV-assisted Vehicle-road-cloud Collaboration

Unmanned Aerial Vehicles (UAVs) play an increasingly vital role in enhancing the communication and computing capabilities of vehicular networks by acting as flexible, mobile edge nodes that can dynamically support data relaying, task offloading, and coverage extension in areas with limited infrastructure [146]. The integration of UAVs with vehicles and roadside infrastructure enables multi-source data fusion, which significantly improves situational awareness, traffic perception, and decision-making accuracy in intelligent transportation systems. However, realizing such vehicle-road-cloud collaboration poses several challenges, including the need for dynamic coordination among heterogeneous agents, ensuring data consistency across distributed sources, managing the limited energy resources of UAVs, dynamic path planning of multiple UAVs, and efficiently allocating computation and communication resources in highly dynamic and uncertain environments.

VIII. Conclusion

In the ongoing pursuit of optimizing VEC systems, DRL algorithms have gained considerable attention. Their ability to make intelligent decisions within the dynamic and complex VEC environment has unlocked new possibilities for enhancing task scheduling and resource allocation. In this paper, the architecture of vehicular edge computing is presented in detail first. Then the concept of DRL and some DRL algorithms are introduced. Subsequently, the latest research on DRL-based task scheduling and resource allocation in VEC scenarios is reviewed. Moreover, open challenges and several future directions are discussed for academics and researchers related to this field.

References

[1] P. Li, Z. Xiao, H. Gao, X. Wang, and Y. Wang, "Reinforcement learning based edge-end collaboration for multi-task scheduling in 6g enabled intelligent autonomous transport systems," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[2] G. Dimitrakopoulos and P. Demestichas, "Intelligent transportation systems," *IEEE Vehicular Technology Magazine*, vol. 5, no. 1, pp. 77–84, 2010.

[3] G. Abdelkader, K. Elgazzar, and A. Khamis, "Connected vehicles: Technology review, state of the art, challenges and opportunities," *Sensors*, vol. 21, no. 22, p. 7712, 2021.

[4] F. Z. Bousbaa, C. A. Kerrache, N. Lagraa, R. Hussain, M. B. Yagoubi, and A. E. K. Tahari, "Group data communication in connected vehicles: A survey," *Vehicular Communications*, p. 100518, 2022.

[5] C. Campolo, A. Molinaro, and R. Scopigno, "Vehicular ad hoc networks," *Standards, Solutions, and Research*, 2015.

[6] A. Boukerche and E. Robson, "Vehicular cloud computing: Architectures, applications, and mobility," *Computer networks*, vol. 135, pp. 171–189, 2018.

[7] W. Duan, X. Gu, M. Wen, Y. Ji, J. Ge, and G. Zhang, "Resource management for intelligent vehicular edge computing networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 9797–9808, 2021.

[8] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2017.

[9] H. Guo, X. Chen, X. Zhou, and J. Liu, "Trusted and efficient task offloading in vehicular edge computing networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 10, no. 6, pp. 2370–2382, 2024.

[10] A. M. A. Hamdi, F. K. Hussain, and O. K. Hussain, "Task offloading in vehicular fog computing: State-of-the-art and open issues," *Future Generation Computer Systems*, vol. 133, pp. 201–212, 2022.

[11] N. Keshari, D. Singh, and A. K. Maurya, "A survey on vehicular fog computing: Current state-of-the-art and future directions," *Vehicular Communications*, vol. 38, p. 100512, 2022.

[12] S. Raza, S. Wang, M. Ahmed, M. R. Anwar *et al.*, "A survey on vehicular edge computing: architecture, applications, technical issues, and future directions," *Wireless Communications and Mobile Computing*, vol. 2019, 2019.

[13] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mobile networks and applications*, vol. 26, pp. 1145–1168, 2021.

[14] R. Meneguette, R. De Grande, J. Ueyama, G. P. R. Filho, and E. Madeira, "Vehicular edge computing: Architecture, resource management, security, and challenges," *ACM Computing Surveys (CSUR)*, vol. 55, no. 1, pp. 1–46, 2021.

[15] S. Sheng, P. Chen, Z. Chen, L. Wu, and Y. Yao, "Deep reinforcement learning-based task scheduling in iot edge computing," *Sensors*, vol. 21, no. 5, p. 1666, 2021.

[16] Z. Xiao, J. Shu, H. Jiang, G. Min, H. Chen, and Z. Han, "Perception task offloading with collaborative computation for autonomous driving," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 457–473, 2022.

[17] P. Li, Z. Xiao, X. Wang, K. Huang, Y. Huang, and H. Gao, "Eptask: Deep reinforcement learning based energy-efficient and priority-aware task scheduling for dynamic vehicular edge computing," *IEEE Transactions on Intelligent Vehicles*, 2023.

[18] G. Li, T.-H. Nguyen, and J. J. Jung, "Traffic incident detection based on dynamic graph embedding in vehicular edge computing," *Applied Sciences*, vol. 11, no. 13, p. 5861, 2021.

[19] A. Hammoud, H. Sami, A. Mourad, H. Otrok, R. Mizouni, and J. Bentahar, "Ai, blockchain, and vehicular edge computing for smart and secure iov: Challenges and directions," *IEEE Internet of Things Magazine*, vol. 3, no. 2, pp. 68–73, 2020.

[20] A. Thakur and R. Malekian, "Fog computing for detecting vehicular congestion, an internet of vehicles based approach: A review," *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 2, pp. 8–16, 2019.

[21] B. Lin, K. Lin, C. Lin, Y. Lu, Z. Huang, and X. Chen, "Computation offloading strategy based on deep reinforcement learning for connected and autonomous vehicle in vehicular edge computing," *Journal of Cloud Computing*, vol. 10, no. 1, p. 33, 2021.

[22] Z. Wang, H. Rong, H. Jiang, Z. Xiao, and F. Zeng, "A load-balanced and energy-efficient navigation scheme for uav-mounted mobile edge computing," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3659–3674, 2022.

[23] L. Mendiboure, M.-A. Chalouf, and F. Krief, "Edge computing based applications in vehicular environments: Comparative study and main issues," *Journal of Computer Science and Technology*, vol. 34, pp. 869–886, 2019.

[24] W. Feng, N. Zhang, S. Li, S. Lin, R. Ning, S. Yang, and Y. Gao, "Latency minimization of reverse offloading in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 5, pp. 5343–5357, 2022.

[25] X. Gu and G. Zhang, "Energy-efficient computation offloading for vehicular edge computing networks," *Computer Communications*, vol. 166, pp. 244–253, 2021.

[26] H. Guo, X. Zhou, J. Wang, J. Liu, and A. Benslimane, "Intelligent task offloading and resource allocation in digital twin based aerial computing networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3095–3110, 2023.

[27] S. Talal, W. S. Yousef, and B. Al-Fuhaidi, "Computation offloading algorithms in vehicular edge computing environment: A survey," in *2021 International Conference on Intelligent Technology, System and Service for Internet of Everything (ITSS-IoE)*. IEEE, 2021, pp. 1–6.

[28] L. U. Khan, I. Yaqoob, N. H. Tran, S. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 200–10 232, 2020.

[29] H. Guo, Y. Wang, J. Liu, and C. Liu, "Multi-uav cooperative task offloading and resource allocation in 5g advanced and beyond," *IEEE Transactions on Wireless Communications*, vol. 23, no. 1, pp. 347–359, 2024.

[30] M. S. Bute, P. Fan, L. Zhang, and F. Abbas, "An efficient distributed task offloading scheme for vehicular edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 149–13 161, 2021.

[31] Y. Xia, H. Zhang, X. Zhou, and D. Yuan, "Location-aware and delay-minimizing task offloading in vehicular edge computing networks," *IEEE Transactions on Vehicular Technology*, 2023.

[32] Y. Hou, Z. Wei, R. Zhang, X. Cheng, and L. Yang, "Hierarchical task offloading for vehicular fog computing based on multi-agent deep reinforcement learning," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[33] S. Liu, J. Yu, X. Deng, and S. Wan, "Fedcpf: An efficient-communication federated learning approach for vehicular edge computing in 6g communication networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1616–1629, 2021.

[34] S. Li, N. Zhang, H. Chen, S. Lin, O. A. Dobre, and H. Wang, "Joint road side units selection and resource allocation in vehicular edge computing," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 190–13 204, 2021.

[35] 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 Journal on Selected Areas in Communications*, vol. 41, no. 2, pp. 432–445, 2022.

[36] H. Tran-Dang, S. Bhardwaj, T. Rahim, A. Musaddiq, and D.-S. Kim, "Reinforcement learning based resource management for fog computing environment: Literature review, challenges, and open issues," *Journal of Communications and Networks*, vol. 24, no. 1, pp. 83–98, 2022.

[37] Y. Fei, Z. Yang, Y. Chen, and Z. Wang, "Exponential bellman equation and improved regret bounds for risk-sensitive reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20 436–20 446, 2021.

[38] P. Li, M. Yi, M. Iqbal, X. Wang, and Z. Xiao, "Graph-based proximal policy optimization empowered adaptive task scheduling leveraging cloud-edge collaboration for consumer electronics," *IEEE Transactions on Consumer Electronics*, 2024.

[39] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[40] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[41] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A theoretical analysis of deep q-learning," in *Learning for dynamics and control*. PMLR, 2020, pp. 486–489.

[42] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[43] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[44] H. Shen, K. Zhang, M. Hong, and T. Chen, "Towards understanding asynchronous advantage actor-critic: Convergence and linear speedup," *IEEE Transactions on Signal Processing*, 2023.

[45] X. Hong, H. Liang, H. Zhang, X. Tang, and L. Zhao, "Enhanced drl strategy for distributed edge computing in vehicular networks," *IEEE Network*, 2024.

[46] Q. Chen, X. Song, T. Song, and Y. Yang, "Vehicular edge computing networks optimization via drl-based communication resource allocation and load balancing," *IEEE Transactions on Mobile Computing*, 2025.

[47] Q. Luo, C. Li, T. H. Luan, and W. Shi, "Collaborative data scheduling for vehicular edge computing via deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9637–9650, 2020.

[48] 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 of Things Journal*, vol. 8, no. 12, pp. 9763–9773, 2020.

[49] S. Xu, Q. Liu, B. Gong, F. Qi, S. Guo, X. Qiu, and C. Yang, "Rjcc: Reinforcement-learning-based joint communicational-and-

computational resource allocation mechanism for smart city iot," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8059–8076, 2020.

[50] M. Z. Alam and A. Jamalipour, "Multi-agent drl-based hungarian algorithm (madrlha) for task offloading in multi-access edge computing internet of vehicles (iovs)," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 7641–7652, 2022.

[51] E. Karimi, Y. Chen, and B. Akbari, "Task offloading in vehicular edge computing networks via deep reinforcement learning," *Computer Communications*, vol. 189, pp. 193–204, 2022.

[52] Z. Li, C. Yang, X. Huang, W. Zeng, and S. Xie, "Coor: Collaborative task offloading and service caching replacement for vehicular edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 7, pp. 9676–9681, 2023.

[53] J. Lin, S. Huang, H. Zhang, X. Yang, and P. Zhao, "A deep reinforcement learning based computation offloading with mobile vehicles in vehicular edge computing," *IEEE Internet of Things Journal*, 2023.

[54] Z. Xue, C. Liu, C. Liao, G. Han, and Z. Sheng, "Joint service caching and computation offloading scheme based on deep reinforcement learning in vehicular edge computing systems," *IEEE Transactions on Vehicular Technology*, 2023.

[55] X. Huang, L. He, and W. Zhang, "Vehicle speed aware computing task offloading and resource allocation based on multi-agent reinforcement learning in a vehicular edge computing network," in *2020 IEEE International Conference on Edge Computing (EDGE)*. IEEE, 2020, pp. 1–8.

[56] H. Yang, Z. Wei, Z. Feng, X. Chen, Y. Li, and P. Zhang, "Intelligent computation offloading for mec-based cooperative vehicle infrastructure system: A deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 7, pp. 7665–7679, 2022.

[57] M. Li, J. Gao, L. Zhao, and X. Shen, "Deep reinforcement learning for collaborative edge computing in vehicular networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 4, pp. 1122–1135, 2020.

[58] P. Hou, X. Jiang, Z. Lu, B. Li, and Z. Wang, "Joint computation offloading and resource allocation based on deep reinforcement learning in c-v2x edge computing," *Applied Intelligence*, pp. 1–21, 2023.

[59] H. Huang, Q. Ye, and Y. Zhou, "Deadline-aware task offloading with partially-observable deep reinforcement learning for multi-access edge computing," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 6, pp. 3870–3885, 2021.

[60] L. Zhao, E. Zhang, S. Wan, A. Hawbani, A. Y. Al-Dubai, G. Min, and A. Y. Zomaya, "Meson: A mobility-aware dependent task offloading scheme for urban vehicular edge computing," *IEEE Transactions on Mobile Computing*, pp. 1–15, 2023.

[61] T. Xiao, Y. Qi, T. Shen, Y. Feng, and L. Huang, "Intelligent task offloading method for vehicular edge computing based on improved-sac," in *2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, vol. 5. IEEE, 2022, pp. 1720–1725.

[62] Z. Wei, B. Li, R. Zhang, X. Cheng, and L. Yang, "Dynamic many-to-many task offloading in vehicular fog computing: A multi-agent drl approach," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 6301–6306.

[63] T. H. Binh, H. Vo, B. M. Nguyen, H. T. T. Binh *et al.*, "Reinforcement learning for optimizing delay-sensitive task offloading in vehicular edge-cloud computing," *IEEE Internet of Things Journal*, 2023.

[64] X. Ju, S. Su, C. Xu, and H. Wang, "Computation offloading and tasks scheduling for the internet of vehicles in edge computing: A deep reinforcement learning-based pointer network approach," *Computer Networks*, vol. 223, p. 109572, 2023.

[65] L. Lu, J. Yu, H. Du, and X. Li, "A3c-based load-balancing solution for computation offloading in sdn-enabled vehicular edge computing networks," *Peer-to-Peer Networking and Applications*, vol. 16, no. 2, pp. 1242–1256, 2023.

[66] C. Chen, H. Li, H. Li, R. Fu, Y. Liu, and S. Wan, "Efficiency and fairness oriented dynamic task offloading in internet of vehicles," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 3, pp. 1481–1493, 2022.

[67] L. Liu, J. Feng, X. Mu, Q. Pei, D. Lan, and M. Xiao, "Asynchronous deep reinforcement learning for collaborative task computing and on-demand resource allocation in vehicular edge computing," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[68] Z. Wei, B. Li, R. Zhang, X. Cheng, and L. Yang, "Many-to-many task offloading in vehicular fog computing: A multi-agent deep reinforcement learning approach," *IEEE Transactions on Mobile Computing*, 2023.

[69] P. Dai, Y. Huang, K. Hu, X. Wu, H. Xing, and Z. Yu, "Meta reinforcement learning for multi-task offloading in vehicular edge computing," *IEEE Transactions on Mobile Computing*, 2023.

[70] S.-s. Lee and S. Lee, "Resource allocation for vehicular fog computing using reinforcement learning combined with heuristic information," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 450–10 464, 2020.

[71] C. Shang, Y. Sun, H. Luo, and M. Guizani, "Computation offloading and resource allocation in noma-mec: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, 2023.

[72] L. Niu, X. Chen, N. Zhang, Y. Zhu, R. Yin, C. Wu, and Y. Cao, "Multi-agent meta-reinforcement learning for optimized task scheduling in heterogeneous edge computing systems," *IEEE Internet of Things Journal*, 2023.

[73] J. Wang, H. Zhao, H. Liu, L. Geng, and Z. Sun, "A distributed vehicle-assisted computation offloading scheme based on drl in vehicular networks," in *2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, 2022, pp. 200–209.

[74] X. Xu, K. Liu, P. Dai, F. Jin, H. Ren, C. Zhan, and S. Guo, "Joint task offloading and resource optimization in noma-based vehicular edge computing: A game-theoretic drl approach," *Journal of Systems Architecture*, vol. 134, p. 102780, 2023.

[75] B. Hazarika, K. Singh, S. Biswas, and C.-P. Li, "Drl-based resource allocation for computation offloading in iov networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 11, pp. 8027–8038, 2022.

[76] X. He, H. Lu, M. Du, Y. Mao, and K. Wang, "Qoe-based task offloading with deep reinforcement learning in edge-enabled internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2252–2261, 2020.

[77] S. Vemireddy and R. R. Rout, "Fuzzy reinforcement learning for energy efficient task offloading in vehicular fog computing," *Computer Networks*, vol. 199, p. 108463, 2021.

[78] X. Peng, Z. Han, W. Xie, C. Yu, P. Zhu, J. Xiao, and J. Yang, "Deep reinforcement learning for shared offloading strategy in vehicle edge computing," *IEEE Systems Journal*, 2022.

[79] X. Wang, Z. Ning, S. Guo, and L. Wang, "Imitation learning enabled task scheduling for online vehicular edge computing," *IEEE Transactions on Mobile Computing*, vol. 21, no. 2, pp. 598–611, 2020.

[80] Y. Liu, S. Wang, Q. Zhao, S. Du, A. Zhou, X. Ma, and F. Yang, "Dependency-aware task scheduling in vehicular edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4961–4971, 2020.

[81] C. Li, F. Liu, B. Wang, C. L. P. Chen, X. Tang, J. Jiang, and J. Liu, "Dependency-aware vehicular task scheduling policy for tracking service vec networks," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 3, pp. 2400–2414, 2023.

[82] G. Liu, F. Dai, B. Huang, Z. Qiang, S. Wang, and L. Li, "A collaborative computation and dependency-aware task offloading method for vehicular edge computing: a reinforcement learning approach," *Journal of Cloud Computing*, vol. 11, no. 1, p. 68, 2022.

[83] H. Peng and X. Shen, "Deep reinforcement learning based resource management for multi-access edge computing in vehicular networks," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2416–2428, 2020.

[84] A. S. Kumar, L. Zhao, and X. Fernando, "Multi-agent deep reinforcement learning-empowered channel allocation in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 1726–1736, 2021.

[85] H. Ke, J. Wang, L. Deng, Y. Ge, and H. Wang, "Deep reinforcement learning-based adaptive computation offloading for mec in heterogeneous vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7916–7929, 2020.

[86] Y. Cui, L. Du, H. Wang, D. Wu, and R. Wang, "Reinforcement learning for joint optimization of communication and computation in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 12, pp. 13 062–13 072, 2021.

[87] D. Long, Q. Wu, Q. Fan, P. Fan, Z. Li, and J. Fan, "A power allocation scheme for mimo-noma and d2d vehicular edge computing based on decentralized drl," *Sensors*, vol. 23, no. 7, p. 3449, 2023.

[88] H. Zhou, K. Jiang, X. Liu, X. Li, and V. C. Leung, "Deep reinforcement learning for energy-efficient computation offloading in mobile-edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1517–1530, 2021.

[89] L. Tan, Z. Kuang, J. Gao, and L. Zhao, "Energy-efficient collaborative multi-access edge computing via deep reinforcement learning," *IEEE Transactions on Industrial Informatics*, 2022.

[90] L. Yao, X. Xu, M. Bilal, and H. Wang, "Dynamic edge computation offloading for internet of vehicles with deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[91] X. Huang, L. He, X. Chen, L. Wang, and F. Li, "Revenue and energy efficiency-driven delay-constrained computing task offloading and resource allocation in a vehicular edge computing network: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8852–8868, 2021.

[92] Z. Gao, L. Yang, and Y. Dai, "Fast adaptive task offloading and resource allocation via multiagent reinforcement learning in heterogeneous vehicular fog computing," *IEEE Internet of Things Journal*, vol. 10, no. 8, pp. 6818–6835, 2022.

[93] C. Yang, B. Liu, H. Li, B. Li, K. Xie, and S. Xie, "Learning based channel allocation and task offloading in temporary uav-assisted vehicular edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 9, pp. 9884–9895, 2022.

[94] Y. Cui, H. Li, D. Zhang, A. Zhu, Y. Li, and H. Qiang, "Multi-agent reinforcement learning based cooperative multitype task offloading strategy for internet of vehicles in b5g/6g network," *IEEE Internet of Things Journal*, 2023.

[95] J. Shi, J. Du, J. Wang, J. Wang, and J. Yuan, "Priority-aware task offloading in vehicular fog computing based on deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16 067–16 081, 2020.

[96] Y. Sun and Q. He, "Joint task offloading and resource allocation for multi-user and multi-server mec networks: A deep reinforcement learning approach with multi-branch architecture," *Engineering Applications of Artificial Intelligence*, vol. 126, p. 106790, 2023.

[97] Y. Ju, Y. Chen, Z. Cao, L. Liu, Q. Pei, M. Xiao, K. Ota, M. Dong, and V. C. Leung, "Joint secure offloading and resource allocation for vehicular edge computing network: A multi-agent deep reinforcement learning approach," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[98] S. A. Kazmi, S. Otoum, R. Hussain, and H. T. Mouftah, "A novel deep reinforcement learning-based approach for task-offloading in vehicular networks," in *2021 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2021, pp. 1–6.

[99] X. Kong, G. Duan, M. Hou, G. Shen, H. Wang, X. Yan, and M. Collotta, "Deep reinforcement learning-based energy-efficient edge computing for internet of vehicles," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 9, pp. 6308–6316, 2022.

[100] J. Huang, J. Wan, B. Lv, Q. Ye, and Y. Chen, "Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning," *IEEE Systems Journal*, 2023.

[101] A. S. Kumar, L. Zhao, and X. Fernando, "Task offloading and resource allocation in vehicular networks: A lyapunov-based deep reinforcement learning approach," *IEEE Transactions on Vehicular Technology*, 2023.

[102] N. Zhao, Z. Ye, Y. Pei, Y.-C. Liang, and D. Niyato, "Multi-agent deep reinforcement learning for task offloading in uav-assisted mobile edge computing," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 6949–6960, 2022.

[103] J. Chen, X. Cao, P. Yang, M. Xiao, S. Ren, Z. Zhao, and D. O. Wu, "Deep reinforcement learning based resource allocation in multi-uav-aided mec networks," *IEEE Transactions on Communications*, vol. 71, no. 1, pp. 296–309, 2022.

[104] H. Zhu, Q. Wu, X.-J. Wu, Q. Fan, P. Fan, and J. Wang, "Decentralized power allocation for mimo-noma vehicular edge computing based on deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 9, no. 14, pp. 12 770–12 782, 2021.

[105] J. Li, B. Gu, Z. Qin, and Y. Han, "Graph tasks offloading and resource allocation in multi-access edge computing: A drl-and-optimization-aided approach," *IEEE Transactions on Network Science and Engineering*, 2023.

[106] J. Gao, Z. Kuang, J. Gao, and L. Zhao, "Joint offloading scheduling and resource allocation in vehicular edge computing: A two layer solution," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 3, pp. 3999–4009, 2022.

[107] I. Budhiraja, N. Kumar, H. Sharma, M. Elhoseny, Y. Lakys, and J. J. Rodrigues, "Latency-energy tradeoff in connected autonomous vehicles: a deep reinforcement learning scheme," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[108] X. Zhu, Y. Luo, A. Liu, N. N. Xiong, M. Dong, and S. Zhang, "A deep reinforcement learning-based resource management game in vehicular edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 3, pp. 2422–2433, 2021.

[109] Z. Ning, P. Dong, X. Wang, J. J. Rodrigues, and F. Xia, "Deep reinforcement learning for vehicular edge computing: An intelligent offloading system," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 6, pp. 1–24, 2019.

[110] H. Tang, H. Wu, G. Qu, and R. Li, "Double deep q-network based dynamic framing offloading in vehicular edge computing," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 3, pp. 1297–1310, 2023.

[111] X. Nie, Y. Yan, T. Zhou, X. Chen, and D. Zhang, "A delay-optimal task scheduling strategy for vehicle edge computing based on the multi-agent deep reinforcement learning approach," *Electronics*, vol. 12, no. 7, p. 1655, 2023.

[112] J. Sun, Q. Gu, T. Zheng, P. Dong, and Y. Qin, "Joint communication and computing resource allocation in vehicular edge computing," *International Journal of Distributed Sensor Networks*, vol. 15, no. 3, p. 1550147719837859, 2019.

[113] K. Mishra, G. N. V. Rajareddy, U. Ghugar, G. S. Chhabra, and A. H. Gandomi, "A collaborative computation and offloading for compute-intensive and latency-sensitive dependency-aware tasks in dew-enabled vehicular fog computing: A federated deep q-learning approach," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2023.

[114] Y. Lin, Z. Gao, H. Du, J. Kang, D. Niyato, Q. Wang, J. Ruan, and S. Wan, "Drl-based adaptive sharding for blockchain-based federated learning," *IEEE Transactions on Communications*, vol. 71, no. 10, pp. 5992–6004, 2023.

[115] S. Wu, N. Chen, G. Wen, L. Xu, P. Zhang, and H. Zhu, "Virtual network embedding for task offloading in iiot: a drl-assisted federated learning scheme," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 4, pp. 6814–6824, 2024.

[116] A. Paul, K. Singh, and C.-P. Li, "A multi-agent federated drl model for vehicular task offloading in wpt-aided eroad environment," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–17, 2025.

[117] Z. Liu, L. Huang, Z. Gao, M. Luo, S. Hosseinalipour, and H. Dai, "Ga-drl: Graph neural network-augmented deep reinforcement learning for dag task scheduling over dynamic vehicular clouds," *IEEE Transactions on Network and Service Management*, 2024.

[118] M. Yi, V. C. Lee, P. Yang, P. Li, Y. Zhang, W. Wei, and H. Gao, "The distributed intelligent collaboration to uav-assisted vec: Joint position optimization and task scheduling," *IEEE Internet of Things Journal*, pp. 1–1, 2025.

[119] D. Sun, Y. Chen, and H. Li, "Intelligent vehicle computation offloading in vehicular ad hoc networks: A multi-agent lstm approach with deep reinforcement learning," *Mathematics*, vol. 12, no. 3, p. 424, 2024.

[120] G. Ma, X. Wang, M. Hu, W. Ouyang, X. Chen, and Y. Li, "Drl-based computation offloading with queue stability for vehicular-cloud-assisted mobile edge computing systems," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 4, pp. 2797–2809, 2023.

[121] J. Lv, K. Li, A. Slowik, and H. Jiang, "Distributed edge intelligence for rapid in-vehicle medical emergency response in internet-of-vehicles," *IEEE Internet of Things Journal*, 2024.

[122] Y. Luzon, O. Baron, V. Verter, and O. Berman, "On designing a fire emergency vehicle fleet," *Available at SSRN 4497843*, 2023.

[123] S. Network, "A guide to active traffic management & traffic control methods," https://www.safetynetworkinc.com/a-guide-to-active-traffic-management-traffic-control-methods/, 2024, [Online; accessed 30-January-2024].

[124] T. Williams, "Autonomous driving tested with real-world scenarios," https://news.vt.edu/articles/2023/10/research-autonomousvehicledemo.html, 2023.

[125] V. Software, "Fast, efficient content delivery and caching solutions," https://www.varnish-software.com/solutions/, 2024.

[126] F. E. M. Program, "Managed electric vehicle charging," https://www.energy.gov/femp/managed-electric-vehicle-charging, 2024.

[127] L. Geng, H. Zhao, J. Wang, A. Kaushik, S. Yuan, and W. Feng, "Deep reinforcement learning based distributed computation offloading in vehicular edge computing networks," *IEEE Internet of Things Journal*, 2023.

[128] B. Hu, Y. Shi, and Z. Cao, "Adaptive energy-minimized scheduling of real-time applications in vehicular edge computing," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 5, pp. 6895–6906, 2023.

[129] H. Liu, H. Zhao, L. Geng, Y. Wang, and W. Feng, "A distributed dependency-aware offloading scheme for vehicular edge computing based on policy gradient," in *2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*. IEEE, 2021, pp. 176–181.

[130] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2169–2182, 2022.

[131] Z. Nan, S. Zhou, Y. Jia, and Z. Niu, "Joint task offloading and resource allocation for vehicular edge computing with result feedback delay," *IEEE Transactions on Wireless Communications*, vol. 22, no. 10, pp. 6547–6561, 2023.

[132] M. K. Hasan, N. Jahan, M. Z. A. Nazri, S. Islam, M. A. Khan, A. I. Alzahrani, N. Alalwan, and Y. Nam, "Federated learning for computational offloading and resource management of vehicular edge computing in 6g-v2x network," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 3827–3847, 2024.

[133] F. Guo, B. Tang, Y. Wang, and X. Luo, "Vehicle edge server deployment based on reinforcement learning in cloud-edge collaborative environment," *Cluster Computing*, vol. 27, no. 10, pp. 14 539–14 556, 2024.

[134] B. Shen, X. Xu, L. Qi, X. Zhang, and G. Srivastava, "Dynamic server placement in edge computing toward internet of vehicles," *Computer Communications*, vol. 178, pp. 114–123, 2021.

[135] X. Dai, Z. Xiao, H. Jiang, and J. C. Lui, "Uav-assisted task offloading in vehicular edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 4, pp. 2520–2534, 2023.

[136] H. Xiao, J. Zhao, J. Feng, L. Liu, Q. Pei, and W. Shi, "Joint optimization of security strength and resource allocation for computation offloading in vehicular edge computing," *IEEE Transactions on Wireless Communications*, vol. 22, no. 12, pp. 8751–8765, 2023.

[137] S. Jiang, J. Li, G. Sang, H. Wu, and Y. Zhou, "Vehicular edge computing meets cache: An access control scheme with fair incentives for privacy-aware content delivery," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 8, pp. 8404–8418, 2024.

[138] T. Meuser, L. Lovén, M. Bhuyan, S. G. Patil, S. Dustdar, A. Aral, S. Bayhan, C. Becker, E. De Lara, A. Y. Ding *et al.*, "Revisiting edge ai: Opportunities and challenges," *IEEE Internet Computing*, vol. 28, no. 4, pp. 49–59, 2024.

[139] H. Hua, Y. Li, T. Wang, N. Dong, W. Li, and J. Cao, "Edge computing with artificial intelligence: A machine learning perspective," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[140] X. Qiang, Z. Chang, Y. Hu, L. Liu, and T. Hämäläinen, "Adaptive and parallel split federated learning in vehicular edge computing," *IEEE Internet of Things Journal*, 2024.

[141] J. Zhang, Y. Wu, G. Min, and K. Li, "Neural network-based game theory for scalable offloading in vehicular edge computing: A transfer learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 7, pp. 7431–7444, 2024.

[142] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[143] G. Qu, Q. Chen, W. Wei, Z. Lin, X. Chen, and K. Huang, "Mobile edge intelligence for large language models: A contemporary survey," *IEEE Communications Surveys & Tutorials*, 2025.

[144] C. Cui, Y. Ma, X. Cao, W. Ye, Y. Zhou, K. Liang, J. Chen, J. Lu, Z. Yang, K.-D. Liao *et al.*, "A survey on multimodal large language models for autonomous driving," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 958–979.

[145] M. R. Shoaib, H. M. Emara, and J. Zhao, "A survey on the applications of frontier ai, foundation models, and large language models to intelligent transportation systems," in *2023 International Conference on Computer and Applications (ICCA)*. IEEE, 2023, pp. 1–7.

[146] J. Yan, X. Zhao, and Z. Li, "Deep-reinforcement-learning-based computation offloading in uav-assisted vehicular edge computing networks," *IEEE Internet of Things Journal*, vol. 11, no. 11, pp. 19 882–19 897, 2024.

Peisong Li (Member, IEEE) is currently with the Hangzhou Institute of Technology, Xidian University, Hangzhou 311200, China. He received a B.S. degree from the Guilin University of Electronic Technology in 2017, a M.S. degree from the Shanghai Maritime University, China, in 2020, and a Ph.D. degree from Xi'an Jiaotong-Liverpool University in 2024. His research interests include Edge computing, Vehicular edge computing, Internet of Vehicles, Edge AI, and Deep Reinforcement Learning.

Xinheng Wang (Senior Member, IEEE) received the B.E. and M.Sc. degrees in electrical engineering from Xian Jiaotong University, Xi'an, China, in 1991 and 1994, respectively, and the Ph.D. degree in electrical engineering from Brunel University, Uxbridge, U.K., in 2001. He is currently a Professor with the School of Advanced Technology and was the founding Head of Department of Mechatronics and Robotics, Xi'an Jiaotong-Liverpool University (XJTLU), Suzhou 215123, China. Prior to joining XJTLU, he was a professor with different universities in the UK.

He has been an Investigator or Co-Investigator of 30+ research projects sponsored from EU, UK EPSRC, Innovate UK, China NSFC, and industry. He has authored or coauthored 220+ referred papers. He holds 22 granted patents, including 1 US, 1 Japan, 4 South Korea and 16 China patents. He was one of the key developers of the world's first smart trolley to provide intelligent services to passengers at airports. He is currently leading the XJTLU-uGo Robotics Research Centre with multi-million Yuan sponsorship from industry to develop airport robots. His current research interests include intelligent and connected systems, including robotics and healthcare systems, indoor acoustic localization, acoustic posture detection and recognition, digitalization of traditional Chinese medicine, and SLAM and navigation for robots.

Changle Li (Senior Member, IEEE) received the Ph.D. degree in communication and information system from Xidian University, China, in 2005. He conducted his post-doctoral research in Canada and the National Institute of information and Communications Technology, Japan, respectively. He had been a Visiting Scholar with the University of Technology Sydney. He is currently a Professor with the State Key Laboratory of Integrated Services Networks, Xidian University. His research interests include intelligent transportation systems, vehicular networks, mobile ad-hoc networks, and wireless sensor networks.

This article has been accepted for publication in IEEE Transactions on Intelligent Transportation Systems. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TITS.2025.3607910

Dr Muddesar Iqbal (Member, IEEE) is a Associate Professor in Communications and Networks Engineering Department, College of Engineering, Prince Sultan University, Saudi Arabia. He did his PhD from Kingston University UK. His research interests include Rebotics, Internet of Senses Digital Twin, V2X technologies, Disaster Management and IoT. He is a visiting Research fellow at School of Computer Science and Electronic Engineering, University of Essex, UK. He has won 17 R&D and Capacity building funding Grants from different national and international funding agencies. He has won two Awards of Appreciation for Tutoring the Prize Winner from the Association of Business Executive (ABE) the UK, the first in 2005 for tutoring on Computer Fundamentals and second in 2010 for tutoring on Information System Project. He has won Australia Award to complete two months Block-based Business Incubator Management Certificate from Queensland University Australia in 2015. He has published 100 peer reviewed articles (including Patents, IEEE reputed Q1 journals, conference proceedings. Books Collections and book Chapters and Co-Invented 5 patented inventions in IoT and Software Defined Networks Technologies and Autonomous Vehicles area.

Chih-Lin I (Fellow, IEEE) is CMCC Chief Scientist of Wireless Technologies. She received Ph.D. EE from Stanford University. She has won 2005 IEEE ComSoc Stephen Rice Prize, 2018 IEEE ComSoc Fred W. Ellersick Prize, the 7th IEEE Asia-Pacific Outstanding Paper Award, and 2015 IEEE Industrial Innovation Award for Leadership and Innovation in Next-Generation Cellular Wireless Networks. She is the Chair of O-RAN Technical Steering Committee and an O-RAN Executive Committee Member, the Chair of FuTURE 5G/6G SIG, an Executive Board Member of GreenTouch, a Network Operator Council Founding Member of ETSI NFV, a Steering Board Member and Vice Chair of WWRF, a Steering Committee member and the Publication Chair of IEEE 5G and Future Networks Initiatives, the Founding Chair of IEEE WCNC Steering Committee, the Director of IEEE ComSoc Meetings and Conferences Board, a Senior Editor of IEEE Trans. Green Comm. Networking, an Area Editor of ACM/IEEE Trans. Networking; Executive Co-chair of IEEE Globecom 2020, IEEE WCNC 2007, IEEE WOCC 2004 and 2000; a member of IEEE ComSoc SDB, SPC, and CSCN-SC, and a Scientific Advisory Board Member of Singapore NRF. She has published over 200 papers in scientific journals, book chapters and conferences and holds over 100 patents. She is a Fellow of IEEE and a Fellow of WWRF. Her current research interests center around ICDT Deep Convergence: "From Green Soft to Open Smart".

Anwer Al-Dulaimi (Senior Member, IEEE) is currently a Senior Strategy Manager for Connectivity and Industry 4.0 at Veltris, Toronto, Canada. He received the Ph.D. degree in electrical and computer engineering from Brunel University, London, U.K., in 2012 after receiving M.Sc. and B.Sc. honors degrees in communication engineering. He was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Toronto, sponsored by Blackberry's advanced research team. He is the chair of the newly established IEEE 5G/6G Innovations Testbed project working to develop a convergent testing platform for E2E network innovation. He is also Associate Professor at CAT-ZU teaching networking subjects. His research interests include 5G and 6G communications, cloud computing, and cybersecurity. Anwer is the editor of Future Networks Series on 6G in IEEE Vehicular Technology magazine, Editor of Vehicular Networking Series in IEEE Standards Magazine, Editor of IEEE ITSS Series on Intelligent Autonomous Transport, and guest editor for many IEEE journals. He was the recipient of the 2013 Worldwide Universities Network best paper for outstanding research in cognitive communications, best IEEE/WWRF VTM best paper award for 3 times, as well as many other awards and grants. He has published many papers in refereed publications and patents. He provided many talks about 5G/6G networks in academia, industry forums, O-RAN and ITU. He is the chair of the IEEE 1932.1 'Standard for Licensed/Unlicensed Spectrum Interoperability in Wireless Mobile Network', invited expert to ITU-T SG11, and voting member of IEEE MobiNet Standards Committee. He is an IEEE ComSoc Distinguished Lecturer, Fellow of the Institution of Engineering and Technology (FIET), Fellow of the British higher education Academy (FHEA) and registered as a Chartered Engineer (CEng) by the British Engineering Council since 2010.

Pablo Casaseca-de-la-Higuera received the M.Eng. degree in telecommunications engineering and the Ph.D. degree in information and communication technologies from the University of Valladolid (UVA), Valladolid, Spain, in 2000 and 2008, respectively. After working for three years in the aerospace industry (Alcatel Espacio), in 2003 he joined the Laboratory of Image Processing, UVA, where he is currently a Full Professor in the Department of Signal Theory and Communications, and Telematics Engineering. Between 2013 and 2017, he held academic positions (Lecturer and Senior Lecturer) at the School of Engineering and Computing, University of the West of Scotland, Paisley, U.K. He has published several book chapters and more than 100 papers in indexed journals and flagship international conferences. His research interests include signal and image processing and artificial intelligence with applications in biomedical engineering, computer vision, telecommunications, and unmanned aerial vehicles.