Toward Secure Task Offloading in Vehicular Edge Computing Networks

Zhu Xiao, *Senior Member, IEEE*, Yanxun Chen, Tong Li, *Member, IEEE*, Shujuan Tian, Geyong Min, *Member, IEEE*, Zhuo Tang, *Member, IEEE*, Kenli Li, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*—Vehicular Edge Computing (VEC) enables vehicular services but still struggles to achieve secure and efficient task offloading under highly dynamic and resource-constrained conditions. To tackle this problem, we develop a secure task offloading framework where task vehicles (TVs) determine offloading decisions while eavesdropping vehicles (EVs) attempt to intercept sensitive data. The joint optimization of offloading ratio, spectrum allocation, and transmission power is modeled as a cooperative multi-agent Markov decision process (MDP) that captures mobility-driven channel variations and resource competition. We then design a Deep Deterministic Policy Gradient (DDPG)-based Secure Task Offloading (DSTO) scheme to solve this continuous-control problem. DSTO encodes vehicular states into compact representations and uses an actor–critic (AC) architecture to learn adaptive strategies. The actor outputs continuous offloading and power-control actions, while the critic evaluates long-term utility under integrated security, delay, and energy constraints. Security is explicitly incorporated through the security rate, reflecting the legitimate link's advantage over the eavesdropping link. Leveraging DDPG's capability in high-dimensional continuous spaces, DSTO robustly adapts to dynamic vehicular conditions and balances security with system performance. Extensive experiments show that DSTO significantly reduces task delay and energy consumption, achieving over 12.7% improvement in comprehensive utility compared with baseline approaches.**

***Index Terms*—Vehicular edge computing, secure task offloading, comprehensive utility, deep reinforcement learning.**

Received 4 December 2025; revised 16 January 2026; accepted 23 May 2026. Date of publication 27 May 2026; date of current version 1 September 2026. This work was supported in part by the National Natural Science Foundation of China under Grant U24A20247, Grant 62471277, and Grant 62272152, in part by the Young Elite Scientists Sponsor Ship Program by China Association for Science and Technology (CAST) under Grant ZB2025-293, in part by the Hunan Natural Science Foundation of China under Grant 2025JJ50351, in part by Major Landmark Innovation Demonstration Project in Hunan Province under Grant 2022XK2301, in part by the National Natural Science Foundation of Hunan Province under Grant 2023JJ30597, and in part by Shenzhen Science and Technology Program under Grant JCYJ20250604185802004. *(Corresponding author: Tong Li.)*

Zhu Xiao, Yanxun Chen, Tong Li, Zhuo Tang, and Kenli Li are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410012, China, and also with Shenzhen Research Institute, Hunan University, Shenzhen 518055, China (e-mail: zhxiao@hnu.edu.cn; chenyanxun@hnu.edu.cn; liay@hnu.edu.cn; ztang@hnu.edu.cn; lkl@hnu.edu.cn).

Shujuan Tian is with the School of Computer Science & School of Cyberspace Security, Xiangtan University, Xiangtan 411105, China (e-mail: sjtianwork@xtu.edu.cn).

Geyong Min is with the Department of Mathematics and Computer Science, College of Engineering, University of Exeter, EX4 4PY Exeter, U.K. (e-mail: g.min@exeter.ac.uk).

Keqin Li is with the Department of Computer Science, State University of New York, Albany, NY 12246 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TDSC.2026.3697937

I. Introduction

RECENT years, as the Internet of Vehicle (IoV) continues to advance rapidly, vehicle users have generated a large number of data-intensive [1], [2] and delay-sensitive tasks [3], [4] in many emerging applications. As a promising solution, vehicle edge computing (VEC) improves the task computing efficiency and the response speed of resource-constrained vehicles by offloading tasks to VEC servers [5]. Although VEC can provide flexible computation task offloading services on demand, the inherent openness of vehicular communication channels and the dynamic nature of IoV topology inevitably pose a risk of eavesdropping during the offloading process of computation tasks [6]. With the increasing security awareness among vehicle users, secure task offloading in VEC has become a crucial requirement for ensuring data confidentiality and integrity while preventing potential security threats.

Existing studies have conducted lots of efforts to address the issues of secure task offloading in VEC networks. Traditional methods, including information encryption [7], identity authentication [8] and blockchain technology [9] are widely used to prevent unauthorized access from eavesdroppers. However, these offloading mechanisms struggle to achieve the best secure offloading results. Firstly, adopting information encryption technology is faced the limited scope of trust policies based on intrusion detection and the difficulty of predicting entrusted vehicles in advance [10]. Secondly, existing authentication mechanisms often rely on trust-centralized third parties, which can easily lead to a single point of failure [11]. Thirdly, the blockchain-based trust solutions are secure, while they encounter scalability issues and require a large amount of storage and computing resources [12]. In addition, those methods, on one side, fail to quickly adapt to highly dynamic and heterogeneous VEC environments; on the other side, they may overlook the optimization of delay and energy consumption due to excessive focus on security, thereby affecting the overall performance of VEC networks.

In this paper, motivated by the above-mentioned analysis, we aim at achieving secure task offloading in VEC networks, via resolving the joint optimization problem of dynamic vehicle task offloading, delay and energy consumption. To achieve this goal, two major technical challenges remain unresolved.

- *How to balance security and efficiency in VEC task offloading:* In VEC environment, reducing security makes offloading tasks vulnerable to eavesdropping; decreasing efficiency leads to excessive delay or accelerated energy consumption for vehicles. An imbalance between security and efficiency may compromise task confidentiality or result in excessive delay and energy consumption, thereby degrading overall system performance in VEC environments.

- *How to address the impact of dynamic characteristics in VEC task offloading:* The high dynamism causes rapid fluctuations in system states (including channel quality, vehicle location, and security risks), making the optimization problem highly time-varying and difficult to predict or model accurately. The strong coupling between communication, computation, and power control decisions further exacerbates the complexity of the problem, as each action dimension simultaneously affects multiple performance metrics, forming a non-convex and interdependent action space.

To tame those challenges, we propose the DDPG-based Secure Task Offloading (DSTO) method to enable secure and efficient task offloading in highly dynamic VEC networks. Within the proposed method, we model the secure task offloading by considering task vehicles (TVs) that offload computation to nearby roadside units (RSUs) while eavesdropping vehicles (EVs) may attempt to intercept sensitive information. In particular, the problem is formulated as a joint optimization problem. As for seeking solutions, we define a comprehensive utility function L that captures those multiple objectives, and specify the system state and action spaces for each vehicle, enabling continuous decision-making in the deep reinforcement learning (DRL) framework. Specifically, we leverage the Actor-Critic (AC) architecture in DDPG, where the actor network generates offloading and resource allocation actions and the critic network evaluates the expected cumulative reward. Following that, an experience replay buffer and soft target network updates are utilized to stabilize training. We derive a multi-objective reward function to adaptively balance security, delay, and energy consumption. As such, via addressing the coupled challenges of secure transmission, energy consumption, and task delay, the proposed DSTO method can effectively enhance secure transmission performance while significantly reducing energy usage and task completion delay in VEC task offloading.

The main contributions of this paper are highlighted as follows.

- We propose a DSTO scheme based on DRL that uses joint optimization to resolve the conflict between security, delay, and energy consumption during task offloading in VEC scenario.
- We formulate the joint optimization problem into a continuous control and NP-hard problem. A DDPG-based method is designed to efficiently solve the problem, which is able to adaptively balance security and performance under dynamic vehicular conditions.
- Experimental results show that the proposed DSTO can provide joint optimization for security and efficiency. Specifically, the proposed method significantly improves the security rate, reduces the task delay and energy consumption, and improves the comprehensive utility by more than 12.7%, demonstrating a clear advantage over existing benchmark methods.

The remainder of this paper is organized as follows. Section II presents the overview of the related work. Section III details the system's architectural structure. In Section IV, we define the problem definition and mathematical formulation. Section V presents the analytical framework and the DSTO scheme. Following that, Section VI presents analyses of the experimental findings. Finally, we conclude the paper in Section VII.

II. Related Work

The rise of VEC paradigm in the IoV has attracted widespread attention [13], [14]. Numerous researches effort have been dedicated to exploring task management and resource distribution within VEC networks [15]. The authors in [16] developed a practical rule-based algorithm to efficiently coordinate task distribution and resource management in VEC systems. An optimizing geo-distributed edge layering framework [17] based on Double Deep Q-Networks (DDQNs) was proposed to support mobility-aware and predictive task offloading in mobile edge computing (MEC) environments. In addition, metaheuristic optimization algorithms [18] have been surveyed for cloud–fog–edge collaboration, highlighting their effectiveness in heterogeneous resource coordination. Beyond these, a blockchain-enabled multi-user optimization strategy for MEC systems [19] jointly considers offloading performance and IoT security, while an advanced DRL-based three-layer D2D–edge–cloud framework [20] focuses on improving task allocation efficiency and reducing service delay. In [21], the authors investigated task offloading in a layered UAV-assisted MEC system and proposes a hybrid metaheuristic algorithm to jointly optimize caching, offloading, UAV deployment, and resource allocation.

Several studies have further investigated vehicle security research in the VEC environment. In [22], the authors proposed a max–min optimization problem to jointly refine offloading decisions and transmit power, aiming to enhance security capacity and computational efficiency. The authors in [23] proposed a novel security framework utilizing blockchain in Named Data Networking (NDN) for VEC networks, focusing on trust management and data integrity protection. In [24], [25], the authors aim to enhance the performance of vehicular edge computing systems by designing edge-based frameworks that support low-latency, delay-sensitive vehicular services through efficient resource management mechanisms. Furthermore, the author in [26] proposed ASDIA framework which enhances privacy preservation in federated learning through adversarial sample generation while maintaining model accuracy. Nevertheless, most existing security-related studies consider security optimization in isolation, and there are still few works on collaborative optimization of security and resource allocation under secure task offloading, especially when delay and energy consumption constraints are jointly taken into account.

Meanwhile, existing methods have made efforts to cope with the challenges of dynamic VEC networks arising from high vehicle mobility and rapidly changing network conditions [27], [28]. In [29], the authors developed a deep-learning-based virtual platform to predict vehicle trajectories and form in-car clouds for computing resource integration. In [30], [31], deep learning (DL) is leveraged to enhance vehicular network intelligence by enabling efficient communication and computation management, while simultaneously improving security and task offloading performance in dynamic vehicular environments. In [32], the authors represented the resource allocation challenge in the IoV using a semi-Markov decision process process was employed, integrating resource reservation strategies along with secondary mechanisms for resource allocation. Inspired by advances in reinforcement learning (RL), RL-based intelligent decision-making has been leveraged to enhance the robustness of autonomous driving behaviors and to jointly optimize security and task offloading in next-generation connected vehicular systems [33], [34]. The authors in [35] designed a DRL-based partial offloading method to adaptively balance local and edge computation in IoV. However, despite these advances, existing methods often address mobility adaptation, security enhancement, or resource optimization separately, and still struggle to cope with fast-varying topology, channel fluctuations, and security risks in highly dynamic VEC environments.

As shown in Table I, the reviewed studies cover a wide range of topics, including resource allocation, task offloading strategies, mobility management, and security enhancement in VEC systems. Most existing works focus on optimizing individual aspects such as delay minimization, resource utilization, or data security. However, few studies have jointly considered the secure task offloading and dynamic resource allocation problem under multi-vehicle collaboration scenarios. Moreover, traditional optimization methods rely on static or quasi-static system models, which are difficult to sustain in highly dynamic vehicular environments. Metaheuristic algorithms typically involve iterative global searches with high computational overhead, limiting their applicability to real-time VEC systems. To bridge this research gap, our proposed DSTO scheme integrates DRL into secure task offloading, achieving dynamic decision-making that simultaneously ensures low delay, high energy efficiency, and robust security performance in VEC environments.

III. System Model

We construct a rapidly changing VEC system. The system involves both vehicle to infrastructure (V2I) and vehicle to vehicle (V2V) communication links. These communication types play a crucial role in enabling seamless connectivity and coordination within the system. The V2I link establishes a connection between vehicles and RSUs through cellular interfaces, providing high-speed data transmission, denoted as $\mathcal{U} = \{1, \ldots, u, \ldots, U\}$, and U subchannels. While the V2V link enables D2D communication for the periodic transmission of messages [36], [37], denoted as $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$. Meanwhile, eavesdropping vehicles are denoted by $\mathcal{J} = \{1, \ldots, j, \ldots, J\}$, and the RSU/MEC server is denoted as K.

We consider mode-4 cellular V2X networks in which vehicles autonomously select wireless resources, adopting an OFDM-based spectrum-sharing framework that enables multiple V2I task-offloading links to reuse the spectrum occupied by V2V links. The system incorporates distinct communication protocols for V2I and V2V. V2I uses OFDMA-based uplink transmission to communicate with the RSU [38]. V2V, on the other hand, is suitable for broadcast-based high-mobility environments [39]. Compared with network-coordinated Mode-3, although Mode-4 may suffer from suboptimal resource allocation due to the lack of centralized coordination, it avoids continuous infrastructure-assisted scheduling and signaling overhead, making it more suitable for highly dynamic vehicular scenarios with fast mobility and potentially intermittent RSU coverage.

A. Vehicle Mobility Model

To capture the inherent dynamics of vehicular networks, vehicle mobility is explicitly incorporated into our system model. Each vehicle's location evolves over time according to its instantaneous velocity, thus introducing continuous variations in communication distance and link quality. Specifically, the position of vehicle u at time t is denoted by $x_u(t)$, and its movement follows

$$x_u(t + Qt) = x_u(t) + v_u(t)Qt, \tag{1}$$

where $v_u(t)$ represents the instantaneous speed influenced by traffic conditions and road topology. This dynamic movement results in a time-varying distance $d_u(t)$ between vehicle u and its associated RSU, directly affecting the communication channel.

Consequently, the V2I channel gain $g_u^{tr}(t)$ becomes time-dependent and is modeled by combining large-scale path loss with small-scale Rayleigh fading, i.e.,

$$g_u^{tr}(t) = g_0 d_u(t)^{-\tau} \xi_u(t), \tag{2}$$

where g_0 denotes the reference channel gain, τ is the path-loss exponent, and $\xi_u(t)$ is the fading coefficient. The variation of $d_u(t)$ introduces non-stationary into the wireless link, making the offloading environment highly dynamic. Rayleigh fading is adopted as a simplifying model to capture rapid small-scale channel variations in highly dynamic vehicular environments, particularly under non-line-of-sight (NLoS) conditions. Although Rician or Nakagami models can better characterize strong line-of-sight (LoS) links, the proposed learning-based framework is agnostic to the specific fading distribution, and alternative channel models can be incorporated without modifying the problem formulation or learning architecture.

By explicitly modeling this mobility, our system captures transient connectivity and dynamic channel variations, enabling adaptive, mobility-aware secure offloading decisions.

B. Computation Task Model

The TVs can choose the flexibility to handle tasks on-site or transfer them to an external server for processing at the edge network, we need to consider these two different situations separately. Meanwhile, in VEC systems, the interaction between vehicles and MEC servers results in delays and energy consumption, which significantly impact the comprehensive utility of the vehicles. Therefore, we explore the effect of the delay and energy consumption in VEC systems.

TABLE I
SUMMARY AND COMPARISON OF RELATED WORKS

Ref.	Research Focus	Methodology / Framework	Main Contributions	Limitations / Gaps
[13]	AI for edge service optimization in IoV	Survey and taxonomy	Comprehensive overview of AI-based optimization methods in IoV	Lacks specific discussion on secure task offloading mechanisms
[14]	Offloading dependent tasks with unknown information	Learning-based adaptive offloading	Efficient offloading with partial information	Does not consider security or multi-vehicle cooperation
[15]	V2V computation offloading	Reinforcement learning approach	Reduces delay through V2V collaboration	No security or energy-delay consideration
[16]	Resource allocation in IoV	Rule-based algorithm	Simplifies task distribution in dynamic VEC	Limited adaptability and lacks security analysis
[17]	Mobility-aware task offloading in geo-distributed edge systems	DDQN-based hierarchical edge framework	Enables predictive mobility-aware offloading and improves decision accuracy	No consideration of security threats; mobility modeling not VEC-specific
[18]	Cloud–fog–edge resource collaboration	Metaheuristic optimization survey	Summarizes heuristic scheduling strategies for heterogeneous architectures	Survey only; lacks secure or real-time offloading solutions
[19]	Blockchain-enabled MEC offloading	Multi-user offloading strategy	Improves IoT performance through blockchain-enhanced task offloading	No mobility modeling; not tailored to VEC environments
[20]	Offloading in D2D–edge–cloud architectures	DRL-based three-layer offloading framework	Improves task allocation and reduces delay	Does not consider vehicular mobility or secure offloading
[21]	Cost-efficient task offloading	MINLP + hybrid metaheuristic	Joint caching/offloading, delay and energy reduction	Heuristic-based, limited dynamics, no security
[22]	Secure offloading and resource allocation	Joint optimization framework	Balances computation and security strength	Focused on single-user optimization; lacks multi-vehicle collaboration
[23]	Blockchain-based security in NDN-VEC	Blockchain and privacy control	Enhances data integrity and access control	Computationally heavy; no dynamic offloading adaptation
[24]	Secure edge-assisted maneuver control	Edge-cloud architecture	Ensures secure autonomous driving control	Security addressed, but offloading optimization is limited
[25]	Offloading–caching joint optimization	Full-duplex enabled optimization framework	Enhances resource utilization and reduces backhaul traffic	Does not include mobility or security; full-duplex assumption may be unrealistic
[26]	Privacy preservation in IoV federated learning	ASDIA with adversarial sample generation	Strengthens FL privacy and robustness	Focuses on FL rather than secure offloading; lacks delay/energy considerations
[27]	Mobility-aware multi-hop offloading	Task migration and scheduling	Considers mobility in task distribution	Ignores security and computational cost trade-offs
[28]	Perception task offloading for autonomous driving	Collaborative computation	Improves perception accuracy and delay	No attention to data security or joint optimization
[29]	Dynamic task offloading	Deep neural network (DNN) model	Predicts optimal offloading policies via DL	No explicit security modeling
[30]	Intrusion detection in IVNs	ML/DL taxonomy and survey	DL effectiveness, emerging FL/TL trends	No unified benchmarks, limited real-time validation
[31]	Traffic offloading in SDN-V2X	Deep learning and SDN	Enhances network efficiency and adaptability	Lacks joint optimization and security perspective
[32]	Dynamic resource allocation	Reinforcement learning	Allocates computing resources efficiently	Security and privacy not incorporated
[33]	Lane-change decision robustness	Adversarial RL	Improves robustness under uncertainty	Not related to offloading or VEC security
[34]	Secure offloading in IoV–MEC systems	DDQN-based multi-objective framework	Optimizes security, delay, and resource use jointly	Lacks mobility awareness; focuses on static IoV settings
[35]	Partial offloading in IoV	DRL-based partial offloading scheme	Improves delay performance and offloading flexibility	Lacks security constraints; not designed for multi-vehicle coordination

1) Local Execution Model: The vehicle processes computation tasks using its own resources once they are generated in the local execution model. Let f_u^l (in cycles/s) denotes the computational capacity available locally on the uth vehicle, and B_u denotes the size of the computing task. The time needed for local executing can be expressed as

$$t_u^l = \frac{\psi \cdot B_u}{f_u^l}, \tag{3}$$

where ψ (in cycles/bit) indicates the task computation load required for the processors to handle communication tasks. As a result, the energy consumed during local execution can be expressed as

$$E_u^l = \eta_1 \cdot (f_u^l)^3 \cdot t_u^l, \tag{4}$$

where $\eta_1 = 1 \times 10^{-27}$ [40] represents the efficiency factor of the processor's computing chipset on vehicle u.

2) Remote Execution Model: In the remote execution model, the TV offloads computation tasks to a nearby MEC server via wireless vehicular channels, leveraging the server's high computing capacity to reduce local energy consumption and processing time. The overall delay consists of task transmission time to the server and the server's computation time, where the latter is formulated as

$$t_u^{exe} = \frac{\psi \cdot B_u}{f_K}, \tag{5}$$

where f_K denotes the computation frequency of MEC server CPU.

The delay associated with uplink transmission for offloading tasks to the RSU can be expressed as

$$t_u^{tr} = \frac{B_u}{R_u^{tr}}, \tag{6}$$

where R_u^{tr} denotes the rate at which data is transmitted uplink, which can be expressed as

$$R_u^{tr} = W \cdot \log_2 \left(1 + \frac{P_{V2I} \cdot g_{u,K}}{\delta_{tr}^2 + I_u}\right), \tag{7}$$

where W is the bandwidth. P_{V2I} represents the power used for uplink transmission of vehicle u. $g_{u,K}$ represents the link quality between vehicle u and RSU K, capturing the channel gain, while δ_{tr}^2 represents the fluctuation in the additive white Gaussian noise variance (AWGN). Without loss of generality, the delay associated with the process of sending back the computation results is disregarded, as commonly assumed in [41], [42]. I_u means the uth TV offloading link's received interference, expressed as

$$I_u = \sum_{n=1}^{N} x_{nu} P_{V2V} g_{n,K}, \tag{8}$$

where x_{nu} refers to the element responsible for multiplexing and allocating subchannels. If the nth V2V link reuses the uth V2I link, then $x_{nu} = 1$; otherwise, $x_{nu} = 0$. We use $\mathcal{X} = \{x_{nu}, n = 1, \ldots, N, u = 1, \ldots, U\}$ to represent the spectrum access. P_{V2V} means the transmission power of nth V2V link. We use $\mathcal{Y} = \{y_{nm}, n = 1, \ldots, N, m = 1, \ldots, M_P\}$ to represent. We constrain the transmission power to M_P discrete levels, i.e., $\{P_m, m = 1, \ldots, M_P\}$. Next, we calculate the transmission power of the nth V2V link, i.e., $P_{V2V} = \sum_{m=1}^{M_P} y_{nm} P_m$. In addition, $g_{n,K}$ denotes the signal strength of the nth V2V communication link, which shares the same subchannel as the uth V2I link to the RSU.

In each coherence time period, we apply the Rayleigh fading channel model to characterize signal propagation [43], [44]. The link quality $g_{n,K}$ between the nth V2V link to the RSU K is

$$g_{n,K} = \alpha_{n,K} \cdot h_{n,K}, \tag{9}$$

where $\alpha_{n,K} = C_L \cdot L_{n,K}^{-\tau_c} \cdot \Delta_{n,K}$ denotes the large-scale fading component. Here, C_L denotes the path-loss constant, $L_{n,K}$ denotes the spatial separation between the nodes, τ_c denotes the attenuation factor, and $\Delta_{n,K}$ is represented by a random variable following a log-normal distribution for shadow fading, characterized by a specific standard deviation. Additionally, $h_{n,K}$ represents the small-scale fading component adheres to an exponential distribution, characterized by a mean value of 1, contributes to the overall channel variation. Likewise, the link quality between the TV and the RSU, along with the link quality between the EV and the TV, can be described as

$$g_{u,K} = \alpha_{u,K} \cdot h_{u,K}, \tag{10}$$

$$g_{u,E} = \alpha_{u,E} \cdot h_{u,E}. \tag{11}$$

The energy consumed by the uth TV is given by

$$E_u^{tr} = P_{V2I} \cdot t_u^{tr}. \tag{12}$$

The energy consumed by the uth TV for the RSU K is given by

$$E_u^K = \eta_2 \cdot f_K^3 \cdot t_u^{exe}, \tag{13}$$

where $\eta_2 = 1 \times 10^{-29}$ [40] represents the efficiency factor of the computing chipset's capacitance at the MEC server. As a result, the system's total delay and total energy consumption of vehicle u are determined as

$$T_u = max\left[(1-\gamma_u) \cdot t_u^l, \gamma_u \cdot \left(t_u^{tr} + t_u^{exe}\right)\right], \tag{14}$$

$$E_u = (1-\gamma_u) \cdot E_u^l + \gamma_u \cdot \left(E_u^{tr} + E_u^K\right), \tag{15}$$

where $\gamma_u \in [0, 1]$ denotes the ratio of tasks offloaded to the RSU, $(1-\gamma_u)$ represents the portion of computation tasks that are locally executed at the TV. $\Gamma = \{\gamma_u \mid \forall u \in U\}$ represents the set of offloading ratios. If the security rate is no greater than zero, i.e., $R_u^S \leq 0$, the system prevents the task from being securely transmitted to the RSU. In such cases, the vehicle may choose to execute delay-sensitive tasks locally rather than offloading them. In this scenario, $\gamma_u = 0$; otherwise, $\gamma_u = 1$.

C. Secure Offloading Model

We mainly consider the situation of a single eavesdropping vehicle. It is assumed that EV possesses advanced eavesdropping abilities, allowing them to intercept all spectral sub-bands. According to [41] and [42], we obtain the eavesdropping rate of the EV who is wiretapping the uth TV, which can be expressed by

$$R_u^E = W \cdot log_2 \left(1 + \frac{P_{V2I} \cdot g_u^E}{\delta_E^2 + I_{ev}}\right), \tag{16}$$

where I_{ev} denotes the interference to the eavesdropper. It can be expressed as

$$I_{ev} = \sum_{u=1}^{U} x_{mu} P_{V2I} g_{u,E} + \sum_{u=1}^{U} \sum_{l \neq m}^{M} x_{mu} a_{lu} P_{V2V} g_{l,E}, \tag{17}$$

where $g_{l,E}$ is the interference gain of lth V2V link.

The worst-case security rate for a single eavesdropper can be obtained by comparing the uplink transmission rate in (7) with the eavesdropper's rate in (16), as follows

$$R_u^S = \left[R_u^{tr} - W \cdot log_2 \left(1 + \frac{P_{V2I} \cdot g_u^E}{\delta_E^2 + I_{ev}}\right)\right]^+, \tag{18}$$

where $[x]^+ \triangleq max(x, 0)$. δ_E^2 represents the noise variance at the eavesdropper's receiver, modeled as AWGN. The security rate must remain positive to maintain secure wireless offloading. Consequently, the security model can be expressed as

$$R_u^S > 0, \forall u \in U. \tag{19}$$

Particularly, in the scenario where multiple eavesdroppers exist within the system, the worst-case security rate considering all eavesdroppers can be determined as

$$R_u^S = \left\{ R_u^{tr} - \max_j \left(R_{u,j}^E \right) \right\}^+, \tag{20}$$

where $R_{u,j}^E$ refers to the security rate encountered by the jth eavesdropper, in relation to vehicle u, reflecting the level of security or confidentiality maintained during the transmission. It is expressed as

$$R_{u,j}^E = W \cdot log_2 \left(1 + \frac{P_{V2I} \cdot g_{u,j}^E}{\delta_E^2 + I_{ev}} \right), \tag{21}$$

where $g_{u,j}^E$ represents the wireless link quality from vehicle u to the specific jth eavesdropper, capturing the strength of the signal intercepted by the eavesdropper.

In more general scenarios with multiple eavesdroppers, especially when collusion among eavesdroppers is possible, the worst-case formulation in (20) enables the system to model multiple eavesdroppers as a single virtual eavesdropper with the strongest interception capability. Under this generalized setting, the necessary adaptations mainly involve augmenting the state representation with additional eavesdropping channel information and redefining the reward function based on the worst-case secrecy rate, while the underlying DDPG-based learning architecture and the centralized training–distributed execution paradigm remain unchanged. Therefore, the proposed secure offloading framework is readily extensible to more stringent security threat models without increasing algorithmic complexity. For simplicity, we focus on the case with a single eavesdropper, i.e., $J = 1$.

IV. Problem Formulation

This paper focuses on secure task offloading in VEC networks, formulating it as a multi-objective optimization problem that jointly considers security, computation delay, and energy consumption. We introduce a comprehensive utility function L to balance these objectives, guiding offloading decisions that optimize performance and energy efficiency while ensuring system security.

A. System Comprehensive Utility

The ultimate goal is to jointly optimize security, delay, and energy consumption. To that end, we define a comprehensive utility function by

$$L = \phi \cdot (\lambda_1 \cdot E + \lambda_2 \cdot T) - (1 - \phi) \cdot (\lambda_3 \cdot R^S), \tag{22}$$

where ϕ is a weighting coefficient that reflects the relative significance of energy consumption and delay in the optimization goal. λ_1, λ_2, and λ_3 are scaling constants introduced to standardize energy consumption, system delay, and overall security are on the same scale. Additionally, we define $C = (\lambda_1 \cdot E + \lambda_2 \cdot T)$ to evaluate the overall energy consumption and the delay of the offloading procedure.

B. System Computation Delay

In scenarios involving multiple TVs offloading tasks, system processing delay is considered a key performance indicator. With the usage of OFDMA technology in VEC communication network, tasks can be transmitted and executed simultaneously. Hence, the system delay is characterized by the longest delay experienced across all vehicles, expressed as

$$T = \max_{u \in U} \{T_u\}. \tag{23}$$

C. System Energy Consumption

In the VEC system, multiple TVs offload tasks concurrently. Hence, the overall energy consumption of every TVs become a crucial efficiency indicator for measuring the overall energy consumption of the vehicle system. We represent the system's energy consumption as

$$E = \sum_{u \in U} E_u. \tag{24}$$

D. Confidentiality Probability

The confidentiality rate refers to the actual transmission rate of a TV while ensuring security, reflecting the performance of the data transmission rate. Hence, the overall confidentiality rate of every TVs is an important performance indicator and can be expressed as

$$R^S = \sum_{u \in U} R_u^S. \tag{25}$$

E. The Joint Optimization Problem

The central objective of this paper is to devise a technique for optimizing the secure task offloading in VEC system. By optimizing the TVs scheduling and task offloading strategy Γ, the optimal solution can be obtained to the joint optimization problem. Specifically, for the VEC computation task optimization problem described in (26), our aim is to strengthen security, minimize delays, and decrease energy consumption, while operating within the limits of the available resources. Hence, the optimization challenge is articulated as

$$\mathcal{P} : \min_{\Gamma, \mathcal{X}, \mathcal{Y}} L. \tag{26}$$

$$\text{s.t. } C1 : R_u^S > 0, \forall u \in \mathbf{U}, \tag{27}$$

$$C2 : E \leq E^{max}, \forall u \in \mathbf{U}, \tag{28}$$

$$C3 : \gamma_u \in [0, 1], \forall u \in \mathbf{U}, \tag{29}$$

$$C4 : T \leq T^{max}, \tag{30}$$

$$C5 : T_u^{offload} \leq T_u^{link}, \tag{31}$$

$$C6 : \sum_{u=1}^{U} x_{nu} \leq 1, \forall n \in \mathbf{N}, \tag{32}$$

$$C7 : x_{nu} = \{0, 1\}, \tag{33}$$

$$C8 : \sum_{m=1}^{M_P} y_{nm} = 1, \forall n \in \mathbf{N}, \tag{34}$$

$$C9 : y_{nm} = \{0, 1\}. \tag{35}$$

To ensure both security and optimal performance of the VEC system, we introduce a set of constraints in the optimization process. C1 guarantees the security and confidentiality of the task offloading procedure. C2 is used to ensure the energy consumption limits of vehicles. C3 limits the scope of offloading decisions. C4 ensures that the system's delay remains below the upper limit. C5 depends the offloading process must be completed within the effective connection duration T_u^{link} when considering vehicle mobility. C6 indicates that the nth V2V link is limited to reusing only one V2I link at most. C7 ensures that subchannel multiplexing allocation element is constrained to binary values, either 0 or 1. C8 implies that each V2V link can select only a single transmission power level. C9 indicates transmission power decision variable is binary indicators.

Proposition 1: $\mathcal{P}$ is a non-linear and non-convex problem.

Proof: $\mathcal{P}$ involves multiple decision variables. In the comprehensive utility function L, we can observe that E, T, and R^S involve non-linear relationships, such as power, ratio, and logarithmic functions. Therefore, $\mathcal{P}$ is a non-linear problem. Additionally, due to the max function within T is not continuously differentiable, its derivatives are discontinuous across different regions, making T itself non-convex. The non-linear characteristics of E and R^S further contribute to the complexity of the problem. As a result, $\mathcal{P}$ is characterized by its non-convexity. This completes the proof.

Proposition 2: $\mathcal{P}$ is a NP-hard problem.

Proof: The optimization challenge of $\mathcal{P}$ can be mapped to a complex variant of the knapsack problem, where task offloading decisions are similar to item selection in the knapsack problem, and the offloading of each task consumes system resources (such as energy consumption, delay, etc.), while generating certain benefits (such as increased security). Each resource (such as bandwidth, computing power, etc.) is similar to a capacity limit in a backpack, and the optimization goal is to select tasks that maximize total revenue, while resource consumption does not exceed system capacity. Given that the knapsack problem is classified as NP-hard and the optimization challenge $\mathcal{P}$ involves multiple resource constraints and decision variables, the optimization problem of $\mathcal{P}$ is also NP-hard. This completes the proof.

To address the aforementioned nonlinear, nonconvex, and NP-hard optimization problem, this paper proposes a DSTO scheme based on DRL. By rationally constructing the vehicle state, action space, and multi-objective reward function, DSTO can autonomously learn optimal task offloading and resource allocation strategies in a dynamic environment, effectively balancing security, delay, and energy consumption, thus achieving efficient and secure task offloading in VEC networks. This scheme will be elaborated in detail in the next section.

V. Solutions

In this section, we optimize the comprehensive utility function L proposed in this paper by designing a DRL-based solution. Specifically, the proposed DSTO scheme is a DDPG-driven secure task offloading scheme, which uses multiple agents to jointly explore the environment and adjust strategies based on changes in the environment, enabling effective optimization in continuous and high-dimensional action spaces.

We adopt the DDPG algorithm as the learning backbone of the proposed DSTO framework. The choice of DDPG is motivated by its strong capability in handling continuous and high-dimensional action spaces, which aligns well with the nature of vehicular task offloading involving continuous variables such as transmission power, spectrum allocation, and offloading ratios. Unlike traditional RL methods designed for discrete decisions, DDPG employs an AC architecture that enables smooth policy optimization and stable convergence through the use of target networks and experience replay. The Actor network generates offloading actions based on real-time observations, while the Critic network evaluates the long-term utility of these actions, allowing the policy to be continuously optimized and make secure and efficient decisions. This makes it particularly suitable for dynamic vehicular edge environments, where network states and channel conditions change rapidly. The following subsections will explain in detail the three key elements of the proposed solution.

A. State, Action and Reward Definitions

We assume that the offloading process of each task is represented by stage t. The combined optimization challenge is expressed in the form of a MDP. We use DRL to address the MDP, which contains four elements: agent, state, action and reward function. To enable the DSTO agent to adapt to vehicular dynamics, mobility-related parameters are embedded into the state space. Specifically, each state $s(t)$ includes the vehicle's instantaneous speed $v_u(t)$, relative distance to the RSU d_u, and current channel condition $h_u(t)$. This allows the learning agent to perceive mobility-induced environmental changes and dynamically adjust its offloading decisions. The continuous action space enables DSTO scheme to fine-tune the offloading ratio and transmission power in real time, maintaining secure and efficient task execution under fluctuating mobility conditions.

1) State: The environment state consists of both the TV's state and the RSU's state, as shown in Fig. 1. The agent's state s_t in the DSTO framework incorporates channel conditions, interference, computation task size, and vehicular mobility (e.g., velocity and position), while the RSU's state includes channel information and allocated computing resources. By embedding dynamic mobility and channel variations, DSTO can perceive environmental changes and adapt offloading decisions accordingly.

$$s_t = \begin{Bmatrix} g_{1,K}^t, \ldots, g_{u,K}^t, \ldots, g_{U,K}^t \\ g_{1,E}^t, \ldots, g_{u,E}^t, \ldots, g_{U,E}^t \\ B_1^t, \ldots, B_u^t, \ldots, B_U^t \\ v_1(t), \ldots, v_u(t), \ldots, v_U(t) \\ I_1^t, \ldots, I_u^t, \ldots, I_U^t \end{Bmatrix}. \tag{36}$$

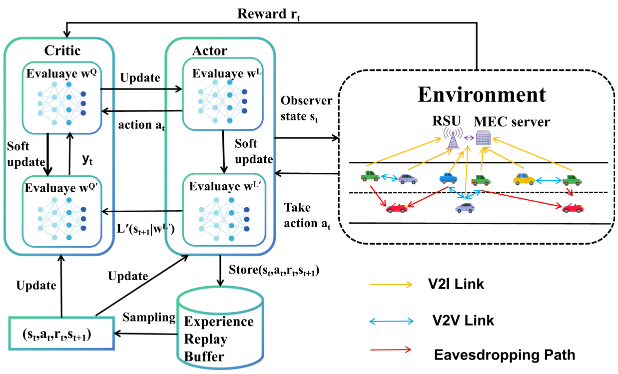

Fig. 1. The proposed secure task offloading framework for the VEC network.

2) Action: The action a_t taken by each agent at time t determines task offloading to the RSU based on the current system state. It consists of spectrum access selection, transmission power, and task offloading ratio. Thus, the action space at time t is the set of all possible actions that govern task offloading or local processing.

$$a_t = \begin{Bmatrix} \gamma_1^t, \ldots, \gamma_u^t, \ldots, \gamma_U^t \\ a_{11}^t, \ldots, x_{nu}^t, \ldots, a_{NU}^t \\ y_{11}^t, \ldots, y_{nm}^t, \ldots, y_{NM_P}^t \end{Bmatrix}. \quad (37)$$

3) Reward: Reward design is critical for model convergence. In the DSTO scheme, the reward function jointly reflects energy consumption, delay, and security performance, while accounting for vehicular mobility. By incorporating mobility-induced variations in link quality and coverage duration, the reward guides the agent to make robust, efficient, and secure offloading decisions under dynamic vehicular conditions. Formally, the reward is designed to minimize energy and delay costs while ensuring secure offloading.

$$R_t = -L. \quad (38)$$

B. The Proposed DSTO

Fig. 1 showcases the proposed secure task offloading framework for solving joint optimization problems. The framework is deployed on the MEC server and adopts the DDPG-based algorithm. DDPG mainly includes actor-critic structure, in which actions are produced by the actor network according to the current state. The evaluation of the actions is carried out by the critic network, it evaluates the standard of these actions and computes the expected cumulative rewards. At the same time, experience playback and target network mechanism is employed to help improve both the robustness and efficiency of the training process.

Specifically, we adopt centralized learning for the training process, while utilizing distributed execution for task implementation. With each pair of actor and critic networks representing an individual agent. In the centralized offline training phase, apart from the state information s_u^t observed locally by the agent u and the action a_u^t adopted by the current vehicle, the action information $\{a_1^t, \ldots a_u^t, \ldots, a_U^t\}$ and state information $\{s_1^t, \ldots s_u^t, \ldots, s_U^t\}$ of other agents are also incorporated. The integrated data is collected and stored in the experience pool of the current agent, enabling centralized training of the critic network. Algorithm 1 presents the procedure for the proposed DSTO.

Algorithm 1: DDPG-Based Secure Task Offloading (DSTO).

Input: Initializing state of the TVs and the RSU, action space Ac, replay buffer size n, mini-batch size B;
Output: Maximizing the reward R_t;
Initialize replay buffer Po with maximum capacity n. Set the initial weights for the actor network $\psi(s \mid \theta^\psi)$ and the critic network $Q(s, a \mid \theta^Q)$ by assigning values to θ^ψ and θ^Q.
Set the initial weights for the target actor network ψ' and the target critic network Q': $\theta^{\psi'} \leftarrow \theta^\psi$, $\theta^{Q'} \leftarrow \theta^Q$.
Set the initial the experience replay buffer Po as an empty data structure.
for *episode = 1 to* M_c **do**
  Set the initial action exploration process.
  Set the initial observation state s_{in} and initialize the cumulative reward $R_t = 0$.
  for *t = 1 to T* **do**
    Compute action a_t, compute the instantaneous reward r_t, and obtain the next state s_{t+1}.
    Compute the cumulative reward: $R_t = R_t + r_t$.
    Record the transition (s_t, a_t, r_t, s_{t+1}) in the prioritized replay buffer Po.
    Sample a mini-batch of B transitions from Po (buffer size n).
    Compute the target L-value y_t using equation (41).
    Compute the critic network parameters are adjusted through the minimization of the loss function defined in equation (42).
    Compute the resource allocation policy according to equation (43).
    Compute the target networks as specified in equations (45) and (46).

We begin by initializing the parameters for the proposed algorithm. A replay buffer P_o with a maximum capacity of n is first created to store transition samples. The actor network $\psi(s \mid \theta^\psi)$ and the critic network $Q(s, a \mid \theta^Q)$ are then initialized by assigning initial values to their parameters θ^ψ and θ^Q. To ensure stable learning, the target actor network ψ' and the target critic network Q' are initialized by directly copying the parameters of the corresponding online networks, i.e., $\theta^{\psi'} \leftarrow \theta^\psi$ and $\theta^{Q'} \leftarrow \theta^Q$. Finally, the replay buffer P_o is set as an empty data structure ready to receive transition tuples.

During each interaction with the environment, an action a_t is selected from the policy network by

$$a_t = \psi(s \mid \theta^{\psi}) + M_{tc}, \tag{39}$$

where M_{tc} represents the additive noise used for exploration, which provides more flexible and diverse exploration compared with the conventional Ornstein-Uhlenbeck process.

The vehicles execute computation offloading and content downloading according to action, while the RSU and BS allocate the corresponding computing, caching, and communication resources. After performing the action, the agent receives an immediate reward r_t and observes the next state s_{t+1}. Each state-action pair, along with the reward and next state, is stored in the experience replay buffer Po for subsequent learning. Mini-batches are then randomly sampled from Po to update the actor network, while the critic network is refined by minimizing the mean squared error between the estimated and target values based on the Bellman equation $Q(s, a \mid \theta^{Q})$, thereby guiding the agent to learn optimal or near-optimal task offloading policies in dynamic vehicular environments.

We adopt transitions (s_t, a_t, r_t, s_{t+1}) as training samples and store them in the replay buffer P_o to train the actor network. The critic network then evaluates the online policy by estimating its corresponding Q value based on the Bellman equation. When considering n steps, the Q value function $Q(s_t, a_t)$ is given by

$$Q(s_t, a_t) = E\left[R(s_t, a_t) + \gamma Q(s_{t+1} \mid \omega^{Q})\right]. \tag{40}$$

Then, the target Q value is computed using the critic-target network, which serves as a stable reference for training the critic network. The target Q value is obtained according to the following equation

$$y_t = R_t + \gamma_c Q'\left(S', \psi'(S' \mid \theta^{\psi'}) \mid \theta^{Q'}\right). \tag{41}$$

The target Q value is computed with the output of the target network $\psi'(S')$, which denotes the action at S', along with the discount factor γ. Subsequently, the parameters θ^{Q} of the critic network are modified through reducing the discrepancy between the predicted values and the target values, achieved through the minimization of the MSE loss, which is defined

$$L_c(\theta) = \frac{1}{M_{tc}} \sum_{t} (y_t - Q(s_t, a_t \mid \theta_Q))^2, \tag{42}$$

where M_{tc} is batch size.

The policy network undergoes refinement through the deterministic policy gradient $\psi(s \mid \theta^{\psi})$, with its gradient update governed by the following rule

$$\nabla_{\theta^{\psi}} \approx \frac{1}{M_{tc}} \sum_{t} \nabla_a Q(s_t, a \mid \theta^{Q}) \nabla_{\theta^{\psi}} \psi(s_t \mid \theta^{\psi}), \tag{43}$$

where $\nabla_a Q(s_t, a \mid \theta^{Q})$ denotes the action's influence on the Critic network's output gradient, indicating how the chosen action a influences the value Q in the state s_t. $\nabla_{\theta^{\psi}} \psi(s_t \mid \theta^{\psi})$ denotes the gradient of the Actor network concerning the actions, reflecting how changes in the policy parameters θ^{ψ} affect the chosen actions.

Next, the policy network parameters are adjusted through the gradient ascent technique, which is given by

$$\theta^{\psi} \leftarrow \theta^{\psi} + \alpha^{\psi} \nabla_{\theta^{\psi}} J(\theta), \tag{44}$$

where α^{ψ} represents the learning rate associated with the actor network. Through the process of, the target network is updated softly, where the parameters of the target networks are adjusted by $\theta^{\psi'}$ and $\theta^{Q'}$ are:

$$\theta^{\psi'} \leftarrow \tau_c \theta^{\psi} + (1 - \tau_c)\theta^{\psi'}, \tag{45}$$

$$\theta^{Q'} \leftarrow \tau_c \theta^{Q} + (1 - \tau_c)\theta^{Q'}, \tag{46}$$

where τ_c is the soft update parameter, typically a small value close to 1, ensuring a gradual update to the target networks. Through continuous interaction with an initially unknown environment, the agent gradually refines its policy. After training, the DDPG model can operate effectively in unseen scenarios, reducing exploration cost and computational burden by iteratively optimizing the action space and decision policy.

C. Algorithm Complexity

As shown in Algorithm 1, the complexity of our proposed algorithm is $O(M_c T B n)$, where M_c denotes the number of training episodes, T is the number of time steps per episode, B is the mini-batch size, and n is the size of the experience replay buffer. This arises from the fact that, at each time step, the agent updates the actor and critic networks using a batch of samples drawn from the replay buffer. Since each update scales linearly with both the batch size and the buffer operations, the overall cost accumulates proportionally across all episodes and time steps, resulting in the stated algorithm complexity.

D. Algorithm Sensitivity

To evaluate the stability and robustness of the proposed DSTO algorithm, we conducted a systematic sensitivity analysis on the key parameters—the learning rate α, discount factor γ, and exploration noise σ. Each parameter was independently varied within $\pm 20\%$ of its nominal value while monitoring the comprehensive utility L. The results show that α has the most pronounced impact on convergence behavior: overly small values lead to slow learning, whereas excessively large values induce oscillations and instability. The discount factor γ governs the trade-off between short-term and long-term gains; higher values enhance long-term utility but slightly slow convergence. In addition, moderate exploration noise σ improves policy robustness by preventing early convergence to suboptimal actions, whereas excessive noise introduces unnecessary fluctuations.

VI. Performance Evaluation

A. Experimental Settings

We simulate vehicle mobility and road traffic flow using the Urban Traffic Simulation (SUMO) platform. Road traffic flow was initially set to 10 vehicles per minute, with an additional monitoring vehicle, and vehicle speeds varied between 10 to 50 $\mathrm{km/h}$. Each RSU covers a 200×200 m^2 area, while vehicles

TABLE II
SIMULATION PARAMETERS

Parameters	Value
Bandwidth of sub-band, W	1 MHz
Variance of AWGN, δ_{tr}^2 and δ_E^2	-100 dBm
Delay cap, T^{max}	2 s
Coefficient depending on local chip, η_1	10^{-27}
Coefficient depending on device chip, η_2	10^{-29}
Vehicular CPU frequency, f_u^l	2 GHz
Frequency of the MEC server, f_K	4 GHz
Energy consumption limit, E^{max}	10 J
The computation intensity of processors, ψ	1000 cycles/bit
Transmission power of V2I , P_{V2I}	20 dBm
Transmission power of V2V , P_{V2V}	[5,10,15,20] dBm
Task size, B_u	[1,1000] MB
Carrier frequency,f_c	2 GHz
BS antenna height	25 m
Vehicle antenna height	1.5 m
Antenna gain of the RSU	10 dBi
Antenna gain of vehicles	3 dBi
Attenuation factor,τ_c	2
Computing density	50 cycles/bit
Vehicular computing cycles, $c_{t_r}^{req}$	[250,500] Mcycles

are randomly distributed within the coverage region. The size of the computation task B_u follows a uniform distribution within $[1, 1000]$ MB, according to [45], [46]. The communication and computing parameters are configured as follows: transmission bandwidth of 1 MHz, noise power of -100 dBm. Other simulation parameters are shown in the Table II.

We adopt path loss and shadowing models for V2I and V2V links as described below, and further calibrate vehicle mobility using real-world traffic traces. The path loss models for V2I and V2V are respectively $128.1 + 37.6 \log_{10}(d)$ and $\epsilon + 10nlog_{10}(d) + X_\sigma$, and their corresponding shadow distribution methods are $Log\text{-}normal$, with shadow standard deviations of 8 dB and 3 dBm, respectively. To incorporate realistic traffic dynamics, we utilize the CN+ dataset—a real-world vehicular dataset [47] recorded at a signal-controlled intersection in Bremen, Germany. This data provides accurate vehicle flow counts and timing information, which we use to calibrate our vehicle arrival rates and intersection behavior in the SUMO-based mobility simulation.

All simulations are implemented in Python 3.11.2 on Windows 10 using PyTorch 2.3.0, and each result is averaged over 100 Monte Carlo runs to ensure statistical reliability. The actor and critic networks adopt three fully connected layers with ReLU activation and are trained using the Adam optimizer, while Gaussian exploration noise with adaptive variance is gradually reduced through linear annealing. In configuring the DSTO algorithm, key parameters including learning rates, discount factor, exploration noise level, and buffer size are selected based on their demonstrated ability to stabilize DRL in dynamic wireless environments. The complete parameter settings, summarized in Table III, follow realistic VEC system configurations and widely referenced studies in vehicular communication [45], [48], [49].

B. Benchmarks

To assess the efficacy of the DSTO scheme, we perform a comparison with the following benchmark methods.

TABLE III
PARAMETER SETTINGS OF THE PROPOSED DSTO SCHEME

Parameter	Value
Actor learning rate, α_{actor}	1×10^{-4}
Critic learning rate, α_{critic}	1×10^{-3}
Discount factor, γ	0.99
Soft update rate, τ	1×10^{-3}
Exploration noise, σ	0.1
Replay buffer size, n	10^6
Batch size, M_{tc}	128
Training episodes	1000
Steps per episode	100
Number of vehicles, N	10
Target update type	Soft update
Random seeds	$\{0, 1, 2, 3, 4\}$

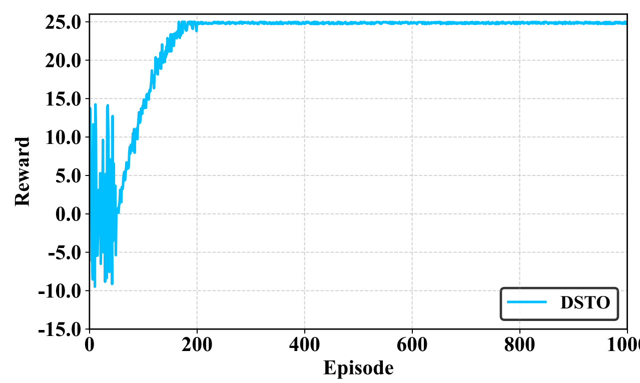

Fig. 2. The convergence behavior of DSTO scheme.

1) *Deep Q-Network (DQN) scheme:* DQN scheme utilizes the L-value function, which is estimated using a neural network model [50]. It enhances training stability by utilizing experience replay and employing a fixed target network to enhance stability [51]. The experience replay buffer enables the algorithm to draw samples from past experiences, effectively breaking the dependencies within the data.

2) *Double Deep Q-Network (DDQN) scheme:* DDQN is an enhancement over the DQN algorithm, specifically developed to resolve the problem of L-value exaggeration that can occur in traditional DQN methods [52]. DDQN introduces dual Q-networks that decouple action selection from value evaluation, effectively mitigating L-value overestimation. This design improves training stability, accelerates convergence.

3) *Asynchronous Advantage Actor Critic (A3C) scheme:* A3C is an asynchronous variant of the Actor-Critic architecture that enhances training speed and stability through multi-threaded parallelization [53]. Each thread in A3C maintains its own AC model and asynchronously updates a shared global model, preventing convergence to local optima and enabling efficient parallel learning on multi-core CPUs.

C. Results Interpretation and Discussion

Fig. 2 depicts the convergence behavior of the DSTO scheme. As iterations progress, the cumulative reward gradually increases and stabilizes, confirming that DSTO scheme effectively optimizes problem P, which aims to balance resource utilization

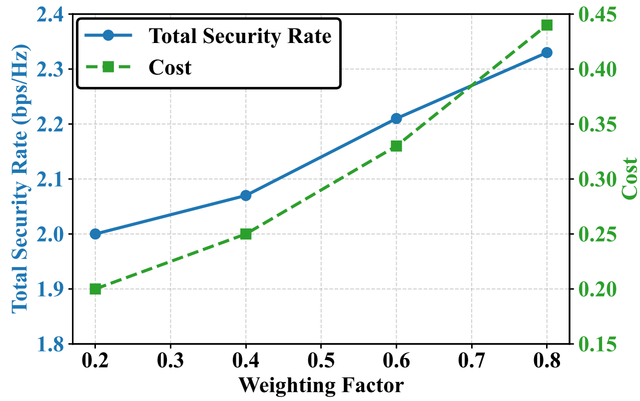

Fig. 3. Influence of the weighting factor ϕ on system performance.

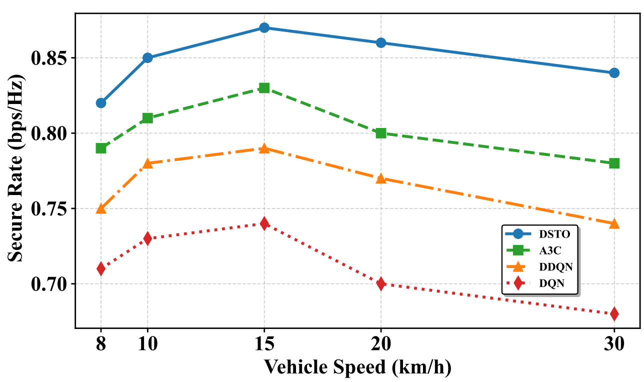

Fig. 4. Vehicle speed versus security rate.

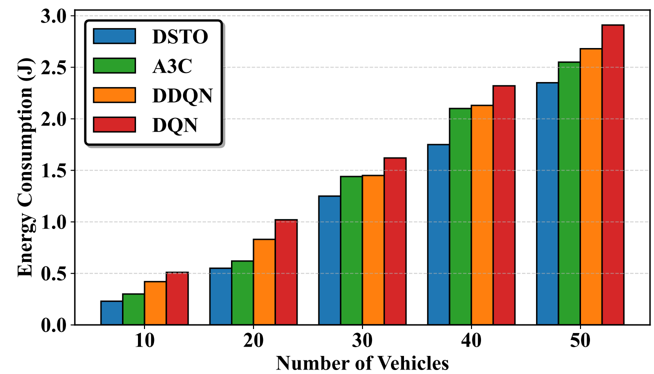

Fig. 5. Energy consumption versus number of vehicles.

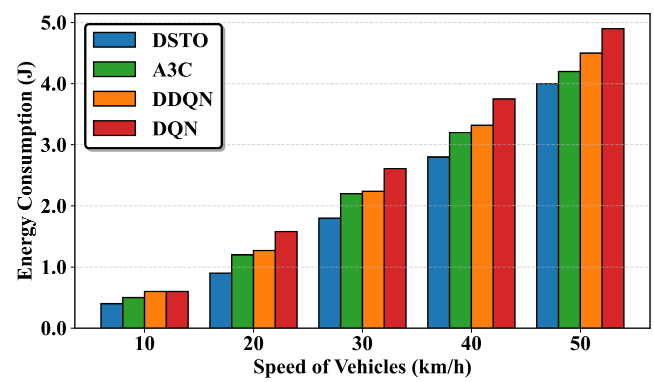

Fig. 6. Energy consumption versus vehicle speed.

and system performance. Initially, the DSTO scheme exhibits irregular fluctuations and insufficient convergence, reflecting the exploratory stage of learning. As iterations progress, the reward values increase rapidly, indicating the algorithm's fast convergence and efficient identification of near-optimal policies. Eventually, the rewards stabilize at a relatively steady value, demonstrating that the DSTO algorithm achieves reliable and consistent performance.

Fig. 3 illustrates how the weighting factor ϕ affects both the cost and the total security rate when $U = 5$. As ϕ increases, both metrics rise accordingly. This occurs because a larger ϕ places greater emphasis on the security rate in the optimization objective, prompting the algorithm to allocate more power and computational resources to enhance security, even at the expense of higher energy cost. This trade-off highlights the flexibility of DSTO scheme in adjusting to different system priorities between energy efficiency and security assurance.

Fig. 4 shows the security rate under different vehicle speeds based on the CN+ mobility dataset [47]. As speed increases, the security rate first rises and then declines. At speeds below 15 $\mathrm{km/h}$, mobility helps vehicles move away from eavesdroppers and introduces beneficial channel fluctuations, temporarily enhancing secrecy performance. However, at speeds above 15 $\mathrm{km/h}$, factors such as shorter RSU dwell time, and exposure to more potential eavesdropping locations all contribute to a decrease in security rate·.

Fig. 5 shows the variation of total energy consumption as the number of vehicles increases from 10 to 50. Compared with other schemes, the proposed DSTO consistently achieves the lowest energy consumption. Specifically, when the number of vehicles is 10, DSTO can reduce energy consumption by at least 23.3%, and for 50 vehicles, the reduction is at least 19.2%. When the number of vehicles is less than 30, the energy consumption reduction is significant, mainly because the DSTO scheme can efficiently utilize idle computing resources and minimize unnecessary transmissions. When the number of vehicles is more than 30, the energy consumption reduction of the proposed method decreases slightly due to increased channel contention and edge server load. However, it can maintains excellent performance by dynamically optimizing the offloading strategy and adaptively allocating computing and transmission resources.

Fig. 6 illustrates the total energy consumption of different schemes under varying vehicle speeds. As the vehicle speed increases from 10 to 50 $\mathrm{km/h}$, the energy consumption generally rises. Specifically, when vehicles move at 10 km/h, DSTO reduces energy consumption by $20.3\% - -33.2\%$ compared with other schemes. At a speed of 50 $\mathrm{km/h}$, the reduction ranges from $5.7\% - -18.4\%$. At higher vehicle speeds, the performance gap narrows because rapid channel variations and shorter RSU connection times limit optimization. DSTO maintains superior

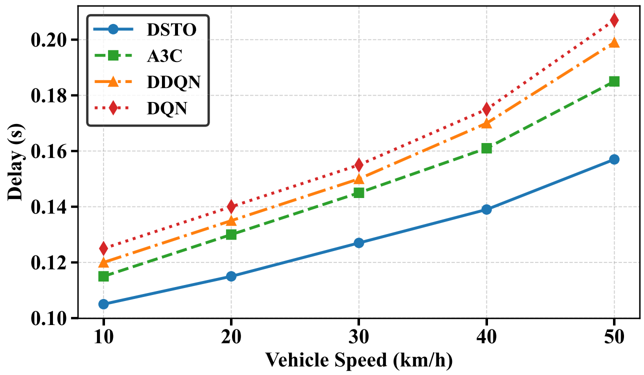

Fig. 7. Delay versus vehicle speed.

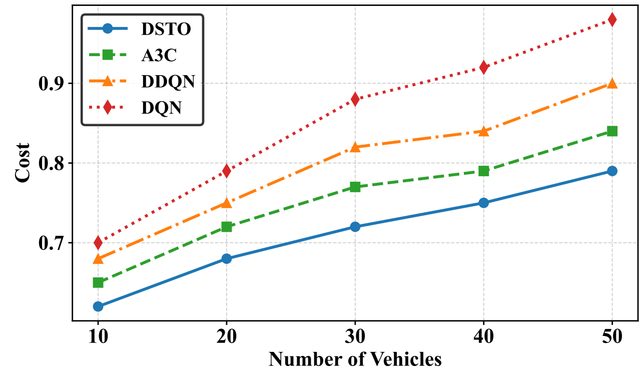

Fig. 9. Cost versus number of vehicles.

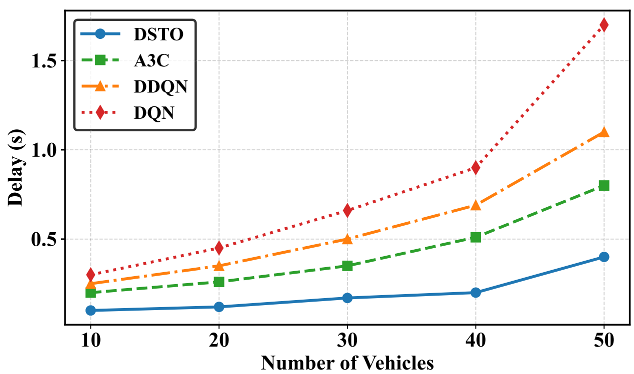

Fig. 8. System delay versus number of vehicles.

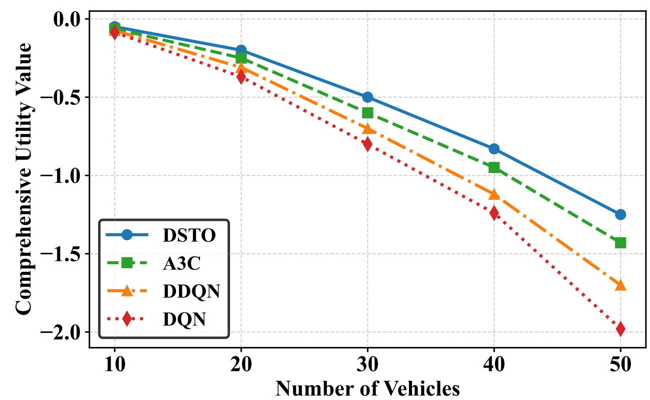

Fig. 10. L-value variation with number of vehicles.

energy efficiency compared with other schemes, owing to its adaptive learning-based optimization.

As illustrated in Fig. 7, the task processing delay of all schemes gradually increases with vehicle speed, reflecting the impact of mobility on V2I communication and task offloading. The DSTO scheme reduced delay by $16.1\% - -24.3\%$ compared to other schemes. Specifically, when the vehicle speed exceeds $30\ \mathrm{km/h}$, the dwell time of the vehicle within the RSU coverage area will be shortened, thereby shortening the offloading window and potentially causing waiting or retransmission delays. Moreover, fast-moving vehicles experience more frequent channel variations, which degrade the effective transmission rate and further increase delay. Among the four schemes, DSTO shows the smallest delay growth because its continuous-action DDPG framework adapts offloading, power, and resource allocation to channel and mobility changes. In contrast, DQN and DDQN suffer larger delay increases since their discrete actions cannot track fast-varying vehicular conditions.

Fig. 8 presents the variation of system delay as the number of vehicles increases when $\phi = 0.7$. The DSTO scheme consistently achieves the lowest delay across all scenarios. When the number of vehicles is 10, DSTO reduces delay by $50.3\% - -66.4\%$ compared with other schemes, and when the number reaches 50, the reduction ranges from $52.6\% - -76.4\%$. When the number of cars is less than 30, the system benefits from fewer competing tasks and more efficient resource allocation, allowing faster task completion. When the number of cars is more than 30, DSTO effectively mitigates delay escalation by dynamically balancing computation loads and optimizing spectrum and transmission power allocation, whereas DQN and DDQN experience performance degradation due to discrete action spaces and limited adaptability under heavy load.

Under the realistic traffic flow generated by the CN+ dataset [47], Fig. 9 shows the variation of system cost with the number of vehicles. As vehicle density increases, the total cost rises due to intensified channel contention and computational resource competition, resulting in higher transmission energy consumption E and longer processing delay T. Across all density levels, the proposed DSTO scheme consistently achieves the lowest system cost and exhibits a noticeably slower growth trend compared with the baselines. Specifically, at 10 vehicles, DSTO reduces the cost by approximately 4.6%–11.2%, and the reduction increases to 6.1%–19.4% at 50 vehicles. These gains stem from DSTO's capability to adaptively balance task offloading and computation by accurately perceiving network states and mobility patterns. In contrast, other algorithms have limited robustness under conditions of high mobility and interference.

Fig. 10 shows the variation of comprehensive utility L-value for different schemes as the number of vehicles increases. Overall, the L-value decrease with vehicle density, reflecting the reduced overall security caused by tighter resource allocation and increased interference. Specifically, when the number of vehicles is 50, the maximum reduction reaches 37.5%. When the number of vehicles is between 10 and 30, the L-value

value decreases less, because the computing and communication resources are relatively sufficient, enabling DSTO to maintain a high-quality offloading strategy. When the number of cars exceeds 30, the increased competition for limited resources leads to greater reductions in L-value; however, DSTO still preserves higher L-value than other methods by adaptively optimizing offloading decisions, transmission power, and spectrum allocation.

VII. Conclusion and Future Work

In this paper, we propose a Deep Deterministic Policy Gradient based Secure Task Offloading (DSTO) scheme for VEC networks, aiming to jointly optimize vehicle scheduling, offloading ratio, transmission power, and spectrum allocation. By leveraging a multi-agent DDPG framework, DSTO effectively addresses the NP-hard nature of the optimization problem and enables vehicles to autonomously learn adaptive offloading strategies under dynamic and uncertain network conditions. The experimental results confirm that DSTO significantly reduces average task delay by $51.5\% - -76.3\%$ and energy consumption by $16.4\% - -18.7\%$, while enhancing the overall system utility by more than 12.7%. These results confirm the effectiveness and practicality of DSTO in ensuring secure, low-delay, and energy-efficient VEC.

Future research will focus on extending the DSTO framework to large-scale heterogeneous vehicular environments and incorporating advanced reinforcement learning and blockchain-based mechanisms to further enhance security and intelligence in task offloading systems.

References

[1] Z. Xiao, J. Shu, H. Jiang, G. Min, J. Liang, and A. Iyengar, "Toward collaborative occlusion-free perception in connected autonomous vehicles," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4918–4929, May 2024.

[2] Z. Xiao et al., "Multi-objective parallel task offloading and content caching in D2D-aided MEC networks," *IEEE Trans. Mobile Comput.*, vol. 22, no. 11, pp. 6599–6615, Nov. 2023.

[3] C. Zhu, Y.-H. Chiang, Y. Xiao, and Y. Ji, "FlexSensing: A QoI and latency-aware task allocation scheme for vehicle-based visual crowdsourcing via deep Q-network," *IEEE Internet Things J.*, vol. 8, no. 9, pp. 7625–7637, May 2021.

[4] Z. Xiao, J. Shu, H. Jiang, G. Min, H. Chen, and Z. Han, "Overcoming occlusions: Perception task-oriented information sharing in connected and autonomous vehicles," *IEEE Netw.*, vol. 37, no. 4, pp. 224–229, Jul./Aug. 2023.

[5] X. Dai, Z. Xiao, H. Jiang, and J. C. S. Lui, "UAV-assisted task offloading in vehicular edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 4, pp. 2520–2534, Apr. 2024.

[6] X. Dai et al., "A learning-based approach for vehicle-to-vehicle computation offloading," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 7244–7258, Apr. 2023.

[7] W. Zhan, X. Luo, F. Long, and G. Deng, "Semi-transparent encryption to prevent leakage of sensitive information," in *Proc. IEEE 2nd Int. Conf. Mobile Netw. Wireless Commun.*, 2022, pp. 1–4.

[8] H. Zhang, G. Dong, D. Liu, G. Wen, J. Xie, and Y. He, "Research on edge cloud storage identity authentication mechanism based on multi-layer integration," in *Proc. 8th Int. Conf. Commun., Image Signal Process.*, 2023, pp. 1–6.

[9] M. Firdaus and K.-H. Rhee, "On blockchain-enhanced secure data storage and sharing in vehicular edge computing networks," *Appl. Sci.*, vol. 11, no. 1, 2021, Art. no. 414.

[10] W. Li, W. Meng, and L. F. Kwok, "Surveying trust-based collaborative intrusion detection: State-of-the-art, challenges and future directions," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 280–305, First Quarter 2022.

[11] M. AlMarshoud, M. S. Kiraz, and A. H. Al-Bayatti, "Security, privacy, and decentralized trust management in VANETs: A review of current research and future directions," *ACM Comput. Surveys*, vol. 56, no. 10, pp. 1–39, 2024.

[12] W. Li, J. Wu, J. Cao, N. Chen, Q. Zhang, and R. Buyya, "Blockchain-based trust management in cloud computing systems: A taxonomy, review and future directions," *J. Cloud Comput.*, vol. 10, no. 1, 2021, Art. no. 35.

[13] X. Xu, H. Li, W. Xu, Z. Liu, L. Yao, and F. Dai, "Artificial intelligence for edge service optimization in Internet of Vehicles: A survey," *Tsinghua Sci. Technol.*, vol. 27, no. 2, pp. 270–287, 2021.

[14] X. Dai et al., "Offloading dependent tasks in edge computing with unknown system-side information," *IEEE Trans. Serv. Comput.*, vol. 16, no. 6, pp. 4345–4359, Nov./Dec. 2023.

[15] X. Dai et al., "A learning-based approach for vehicle-to-vehicle computation offloading," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 7244–7258, Apr. 2023.

[16] A. Salehi and S. Azizi, "An efficient resource allocation algorithm for task offloading in the Internet of Vehicles," *Comput. Knowl. Eng.*, vol. 8, pp. 1–10, 2024.

[17] A. M. Rahmani et al., "An optimizing geo-distributed edge layering with double deep Q-networks for predictive mobility-aware offloading in mobile edge computing," *Ad Hoc Netw.*, vol. 172, 2025, Art. no. 103804.

[18] A. M. Rahmani et al., "Optimizing task offloading with metaheuristic algorithms across cloud, fog, and edge computing networks: A comprehensive survey and state-of-the-art schemes," *Sustain. Comput.: Inform. Syst.*, vol. 45, 2025, Art. no. 101080.

[19] A. M. Rahmani, J. Tanveer, F. S. Gharehchopogh, S. Rajabi, and M. Hosseinzadeh, "A novel offloading strategy for multi-user optimization in blockchain-enabled mobile edge computing networks for improved Internet of Things performance," *Comput. Elect. Eng.*, vol. 119, 2024, Art. no. 109514.

[20] K. Moghaddasi, S. Rajabi, F. S. Gharehchopogh, and A. Ghaffari, "An advanced deep reinforcement learning algorithm for three-layer D2D-edge-cloud computing architecture for efficient task offloading in the Internet of Things," *Sustain. Comput., Inform. Syst.*, vol. 43, 2024, Art. no. 100992.

[21] H. Yuan et al., "Cost-efficient task offloading in mobile edge computing with layered unmanned aerial vehicles," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 30496–30509, Oct. 2024.

[22] 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 Trans. Wireless Commun.*, vol. 22, no. 12, pp. 8751–8765, Dec. 2023.

[23] K. Lei et al., "Blockchain-based cache poisoning security protection and privacy-aware access control in NDN vehicular edge computing networks," *J. Grid Comput.*, vol. 18, pp. 593–613, 2020.

[24] A. Tesei, M. Luise, P. Pagano, and J. Ferreira, "Secure multi-access edge computing assisted maneuver control for autonomous vehicles," in *Proc. IEEE 93rd Veh. Technol. Conf.*, 2021, pp. 1–6.

[25] X. Dai, S. Tian, H. Liu, Z. Li, H. Jiang, and Q. Deng, "Joint optimization of offloading and caching in full-duplex-enabled edge computing networks," *IEEE Trans. Mobile Comput.*, vol. 24, no. 8, pp. 6996–7011, Aug. 2025.

[26] S. Tian, Y. Tan, H. Wang, H. Liu, and Z. Li, "ASDIA: An adversarial sample to preserve privacy program in federated learning," *IEEE Trans. Dependable Secure Comput.*, vol. 22, no. 4, pp. 4330–4343, Jul./Aug. 2025.

[27] L. Liu, M. Zhao, M. Yu, M. A. Jan, D. Lan, and A. Taherkordi, "Mobility-aware multi-hop task offloading for autonomous driving in vehicular edge computing and networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2169–2182, Feb. 2023.

[28] Z. Xiao, J. Shu, H. Jiang, G. Min, H. Chen, and Z. Han, "Perception task offloading with collaborative computation for autonomous driving," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 457–473, Feb. 2023.

[29] S. Yang, G. Lee, and L. Huang, "Deep learning-based dynamic computation task offloading for mobile edge computing networks," *Sensors*, vol. 22, no. 11, 2022, Art. no. 4088.

[30] M. Almehdhar et al., "Deep learning in the fast lane: A survey on advanced intrusion detection systems for intelligent vehicle networks," *IEEE Open J. Veh. Technol.*, vol. 5, pp. 869–906, 2024.

[31] B. Fan, Z. He, Y. Wu, J. He, Y. Chen, and L. Jiang, "Deep learning empowered traffic offloading in intelligent software defined cellular V2X networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13328–13340, Nov. 2020.

[32] H. Liang et al., "Reinforcement learning enabled dynamic resource allocation in the Internet of Vehicles," *IEEE Trans. Ind. Inform.*, vol. 17, no. 7, pp. 4957–4967, Jul. 2021.
[33] X. He, H. Yang, Z. Hu, and C. Lv, "Robust lane change decision making for autonomous vehicles: An observation adversarial reinforcement learning approach," *IEEE Trans. Intell. Veh.*, vol. 8, no. 1, pp. 184–193, Jan. 2023.
[34] K. Moghaddasi, S. Rajabi, and F. S. Gharehchopogh, "Multi-objective secure task offloading strategy for blockchain-enabled IoV-MEC systems: A double deep Q-network approach," *IEEE Access*, vol. 12, pp. 3437–3463, 2024.
[35] S. Tian, X. Zhu, B. Feng, Z. Zheng, H. Liu, and Z. Li, "Partial offloading strategy based on deep reinforcement learning in the Internet of Vehicles," *IEEE Trans. Mobile Comput.*, vol. 24, no. 7, pp. 6517–6531, Jul. 2025.
[36] M. Fu, T. Zhang, W. Song, Y. Yang, and M. Wang, "Trajectory prediction-based local spatio-temporal navigation map for autonomous driving in dynamic highway environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6418–6429, Jul. 2022.
[37] M. S. Bute, P. Fan, L. Zhang, and F. Abbas, "An efficient distributed task offloading scheme for vehicular edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13149–13161, Dec. 2021.
[38] J. Li et al., "Effective joint scheduling and power allocation for URLLC-oriented V2I communications," *IEEE Trans. Veh. Technol.*, vol. 73, no. 8, pp. 11694–11705, Aug. 2024.
[39] B. Saoud et al., "5G-based V2V broadcast communications: A security perspective," *ICT Exp.*, vol. 10, no. 4, pp. 959–980, 2024.
[40] Y. Ju et al., "NOMA-assisted secure offloading for vehicular edge computing networks with asynchronous deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 3, pp. 2627–2640, Mar. 2024.
[41] T. Zhao, L. He, X. Huang, and F. Li, "DRL-based secure video offloading in MEC-enabled IoT networks," *IEEE Internet Things J.*, vol. 9, no. 19, pp. 18710–18724, Oct. 2022.
[42] T. Zhao, F. Li, and L. He, "DRL-based joint resource allocation and device orchestration for hierarchical federated learning in NOMA-enabled industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 19, no. 6, pp. 7468–7479, Jun. 2023.
[43] H. Taramit, J. J. Camacho-Escoto, J. Gomez, L. Orozco-Barbosa, and A. Haqiq, "Accurate analytical model and evaluation of Wi-Fi HaLow based IoT networks under a Rayleigh-fading channel with capture," *Mathematics*, vol. 10, no. 6, 2022, Art. no. 952.
[44] M. Chen et al., "Modeling and performance analysis of single-server database over quasi-static Rayleigh fading channel," *IEEE Trans. Veh. Technol.*, vol. 72, no. 7, pp. 9602–9607, Jul. 2023.
[45] Z. Ning et al., "Intelligent edge computing in Internet of Vehicles: A joint computation offloading and caching solution," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2212–2225, Apr. 2021.
[46] Z. Ning et al., "Joint computing and caching in 5G-envisioned Internet of Vehicles: A deep reinforcement learning-based traffic control system," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5201–5212, Aug. 2021.
[47] T. Karunathilake, M. Zongo, D. Amarawardana, and A. Förster, "CN : Vehicular dataset at traffic light regulated intersection in bremen, Germany," *Sci. Data*, vol. 11, no. 1, 2024, Art. no. 665, doi: 10.1038/s41597-024-03498-4.
[48] Y. Peng et al., "Computing and communication cost-aware service migration enabled by transfer reinforcement learning for dynamic vehicular edgecomputing networks," *IEEE Trans. Mobile Comput.*, vol. 23, no. 1, pp. 257–269, 2024.
[49] B. Qiu, Y. Wang, H. Xiao, and Z. Zhang, "Deep reinforcement learning-based adaptivecomputation offloading and power allocation in vehicular edge computing networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 10, pp. 13339–13339, 2024.
[50] J. Chi, X. Zhou, F. Xiao, Y. Lim, and T. Qiu, "Task offloading via prioritized experience-based double dueling DQN in edge-assisted IIoT," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 14575–14591, Dec. 2024.
[51] B. Wu and C. S. Suh, "Deep reinforcement learning for decentralized multi-robot control: A DQN approach to robustness and information integration," 2024. [Online]. Available: https://arxiv.org/abs/2408.11339
[52] Z. Xu, X. Chen, and G. Hua, "Applying DDQN for advanced model tuning: An evaluation of optimization methods and results," in *Proc. 7th Int. Conf. Inf. Commun. Signal Process.*, 2024, pp. 1079–1083.
[53] C. Sun, T. Li, M. Wu, and S. Chu, "Research on adaptive circuit structure optimization in electronic design based on asynchronous advantage actor critic (A3C) algorithm," *J. Intell. Fuzzy Syst.*, vol. 47, pp. 1–12, 2024.

Zhu Xiao (Senior Member, IEEE) received the MS and PhD degrees in communication and information systems from Xidian University, China, in 2007 and 2009, respectively. From 2010 to 2012, he was a research fellow with the Department of Computer Science and Technology, University of Bedfordshire, U.K. He is currently a full professor with the College of Computer Science and Electronic Engineering, Hunan University, China. His research interests include mobile computing, Internet of Vehicles, and intelligent signal processing. He is serving as an associate editor for *IEEE Transactions on Intelligent Transportation Systems*.

Yanxun Chen received the BS degree in electronic information engineering from the Hunan University of Science and Technology, Xiangtan, China, in 2020, and the MS degree in communication engineering in 2023 with Hunan University, Changsha, China, where he is currently working toward the PhD degree in communication engineering. His research interests include vehicle positioning, mobile edge computing, and data mining.

Tong Li (Member, IEEE) received the PhD degree in computer science and engineering from The Hong Kong University of Science and Technology in 2021 and the second PhD degree in computer science from the University of Helsinki in 2022. He is currently a full professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include mobile computing, wireless network digital twins, network simulation and optimization, and data-driven networking

Shujuan Tian received the PhD degree from Central South University in 2016. From 2019 to 2020, she was a postdoctoral researcher with the Computer Science Department of Illinois Institute of Technology, and cooperated with Prof. Pengjun Wan. She is currently a professor with the School of Computer Science, Xiangtan University. Her research interests include mobile edge computing, VANET, Internet of Things, and network security.

Geyong Min (Member, IEEE) received the BSc degree in computer science from the Huazhong University of Science and Technology, China, in 1995, and the PhD degree in computing science from the University of Glasgow, U.K., in 2003. He is currently a professor of high performance computing and networking with the Department of Computer Science, University of Exeter, U.K. His research interests include computer networks, wireless communications, parallel and distributed computing, ubiquitous computing, multimedia systems, modelling, and performance engineering.

Zhuo Tang (Member, IEEE) received the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2008. He is currently a professor of the College of Computer Science and Electronic Engineering, Hunan University. He is also the chief engineer of the National Supercomputing Center, Changsha, China. He has authored or coauthored almost 230 journal articles and book chapters. His research interests include distributed computing system, cloud computing, parallel processing for Big Data, distributed machine learning, security model, parallel algorithms, and resources scheduling and management in these areas. He is a member of IEEE/ACM and CCF.

Kenli Li (Senior Member, IEEE) received the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2008. He is currently a professor of the College of Computer Science and Electronic Engineering, Hunan University. He is also the chief engineer of the National Supercomputing Center, Changsha, China. He has authored or coauthored almost 230 journal articles and book chapters. His research interests include distributed computing system, cloud computing, parallel processing for Big Data, distributed machine learning, security model, parallel algorithms, and resources scheduling and management in these areas. He is a member of IEEE/ACM and CCF.

Keqin Li (Fellow, IEEE) received the BS degree in computer science from Tsinghua University in 1985 and the PhD degree in computer science from the University of Houston in 1990. He is currently a SUNY distinguished professor with the State University of New York and national distinguished professor with Hunan University, China. He has authored or coauthored more than 1290 journal articles, book chapters, and refereed conference papers. He holds nearly 80 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top few most influential scientists in parallel and distributed computing, regarding single-year impact (ranked #2) and career-long impact (ranked #3) based on a composite indicator of the Scopus citation database. He is listed in Scilit Top Cited Scholars and is among the top 0.02% out of over 20 million scholars worldwide based on top-cited publications in the last ten years. He is listed in ScholarGPS Highly Ranked Scholars and is among the top 0.002% out of over 30 million scholars worldwide based on a composite score of three ranking metrics for research productivity, impact, and quality in the recent five years. He is a member of Academia Europaea (MAE), member of the European Academy of Sciences and Arts (EASA), fellow of the American Association for the Advancement of Science (AAAS), member of the USA National Academy of Artificial Intelligence (NAAI), fellow of the International Academy of Artificial Intelligence Sciences (AAIS), fellow of the International Artificial Intelligence Industry Alliance (AIIA), member of the World Academy of Sciences (WAS), fellow of the World Academy of Engineering & Technology (WAET), fellow of the Institute of Electrical and Electronics Engineers (IEEE), fellow of the Asia Computational Intelligence Society (ACIS), and member of the SUNY Distinguished Academy.